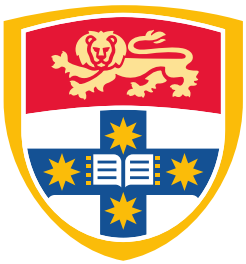
# Slot filling

*Glen Pink*

*Supervisor: James R. Curran*



THE UNIVERSITY OF SYDNEY

A thesis submitted

in fulfilment of the requirements

for the degree of Doctor of Philosophy

The University of Sydney

School of Information Technologies

2017

# Abstract

*Slot filling* (SF) is the task of automatically extracting facts about particular entities from unstructured text, and populating a knowledge base (KB) with these facts. These structured KBs enable applications such as structured web queries and question answering.

SF is typically framed as a query-oriented setting of the related task of relation extraction. Throughout this thesis, we reflect on how SF is a task with many distinct problems. We demonstrate that recall is a major limiter on SF system performance. We contribute an analysis of typical SF recall loss, and find a substantial amount of loss occurs early in the SF pipeline. We confirm that accurate NER and coreference resolution are required for high-recall SF. We measure upper bounds using a naïve graph-based semi-supervised bootstrapping technique, and find that only 39% of results are reachable using a typical feature space.

We expect that this graph-based technique will be directly useful for extraction, and this leads us to frame SF as a label propagation task. We focus on a detailed graph representation of the task which reflects the behaviour and assumptions we want to model based on our analysis, including modifying the label propagation process to model multiple types of label interaction. Analysing the graph, we find that a large number of errors occur in very close proximity to training data, and identify that this is of major concern for propagation. While there are some conflicts caused by a lack of sufficient disambiguating context—we explore adding additional contextual features to address this—many of these conflicts are caused by subtle annotation problems.

We find that lack of a standard for how *explicit* expressions of relations must be in text makes consistent annotation difficult. Using a strict definition of explicitness results in 20% of correct annotations being removed from a standard dataset. We contribute several annotation-driven analyses of this problem, exploring the definition of slots and the effect of the lack of a concrete definition of explicitness:

annotation schema do not detail how explicit expressions of relations need to be, and there is large scope for disagreement between annotators. Additionally, applications may require relatively strict or relaxed evidence for extractions, but this is not considered in annotation tasks. We demonstrate that annotators frequently disagree on instances, dependent on differences in annotator world knowledge and thresholds on making probabilistic inference.

SF is fundamental to enabling many knowledge-based applications, and this work motivates modelling and evaluating SF to better target these tasks.

# Acknowledgements

Thanks go to James Curran: every supervision meeting has ideas and sparked motivation even in the least significant of results. I hope that I have gained even a tiny amount of James' enthusiasm and thoughtfulness (even when presented with yet another boundless "summary" spreadsheet). Thanks to three examiners for their thoughtful feedback to this work.

Thanks to Will Radford, who did more than his fair share in showing me the ropes of KBP and day-to-day research. The same thanks goes to Ben Hachey for the end of my PhD, who (through coffees and lunch) has helped shape an avalanche of barely constructed idea into a thesis. Thanks to Joel Nothman who is an endless source of academic and technical knowledge and rigour.

Thanks to Kellie Webster, Dominick Ng, Tim Dawborn and Nicky Ringland, for making the lab a great place to work. Going through the PhD journey at the same time as Kellie was a great motivator! Thanks to Tim for being always happy to help with technical disasters big and small. Thanks to Ben, Will, Kellie, Dom and Tim for feedback on parts of this thesis. In my time in ə-lab, many members have helped shape my PhD: thanks to Andrew Naoum, Daniel Tse, Tim O'Keefe Andrew Chisholm, James Constable and Will Cannings.

Thanks to my parents, Alan and Christine for supporting me even when they didn't really understand what it was I was doing. Finally, thanks to Danika, I couldn't done it without you (and I'm sure you're happier than me that this thesis is finally done)!

# Statement of compliance

I certify that:

- I have read and understood the University of Sydney Student Plagiarism: Coursework Policy and Procedure;
- I understand that failure to comply with the Student Plagiarism: Coursework Policy and Procedure can lead to the University commencing proceedings against me for potential student misconduct under Chapter 8 of the University of Sydney By-Law 1999 (as amended);
- this Work is substantially my own, and to the extent that any part of this Work is not my own I have indicated that it is not my own by Acknowledging the Source of that part or those parts of the Work.

Name: *Glen Pink*

Signature: Date: *9th July 2017*

# Contents

# List of Figures

# List of Tables

# 1 Introduction

Knowledge bases (KBs) such as Wikipedia[1] contain a large amount of information. Such KBs are of enormous value as community resources, and have wide community reach: Wikipedia itself has had 5–10 billion pageviews/month since 2009.[2] Such KBs are curated and kept up-to-date by human editors. As well as writing articles, editors extract individual facts from text into *infoboxes*, converting unstructured text into a structured format.

This structured format makes key facts within articles more easily accessible, and provides a consistent presentation of information. Critically, this structured format makes data available for further machine processing, which cannot directly make use of unstructured data. Creating structured KBs—in the form of Wikipedia infoboxes or other KBs in more specific domains—enables applications to make use of an immense amount of data that would not otherwise be available.

These structured KBs support a wide variety of valuable applications. The most straightforward applications include web search specifically for structured information: a web query for Mia Farrow birthday essentially becomes a lookup in a KB. More sophisticated uses include question answering (QA). QA is a notable application for structured data, particularly as QA is quickly becoming a core method of accessing information via systems like IBM's Watson[3] and personal assistants

[1] en.wikipedia.org
[2] reportcard.wmflabs.org
[3] ibm.com/watson

like Siri[4] and Google Now.[5] Structured knowledge allows for direct answers to many potential questions. This includes relatively simple questions which can be mapped to a structured KB, such as a question form of our previous query, What is Mia Farrow's birthday?. It also includes the components of more complex questions, in What is Mia Farrow's first husband's birthday? we need information about both the husband and the date of birth. KBs also allow for fact checking of news articles and other documents, as errors in articles can be automatically identified.

More domain-specific KBs allow for further applications. A core goal of the field of biomedical NLP is to use extracted structured data to identify trends in biological events, such as in protein-gene interactions (Kim et al., 2015), in order to support medical research. Financial applications make use of structured data derived from news articles and company documents to support both rapid decision making processes and identification of broader long-term trends in financial markets (Schumaker and Chen, 2009). On a smaller scale, structured information extracted from emails maybe used to support personal workflow and productivity, such as extracting event details into a calendar[6] or contact details from emails.

KBs need to have huge amounts of data if they are to provide significant knowledge for any particular application. Human curation of unstructured text into a structured KB format is limited by human response times, reading times, and ability to collate large amounts of source data (documents such as news articles and web pages) into a single set of facts. With huge amounts of additional available data generated daily, humans cannot possibly curate structured facts from more than a relatively small number of documents, particularly when response times are critical (as for financial applications) or where large amounts of data are required (as for general purpose QA). Most information, except for facts mentioned very frequently, will likely be missed. This may be due to facts being mentioned too

[4] apple.com/ios/siri
[5] google.com/search/about/learn-more/talk
[6] *Events from Gmail*, https://support.google.com/calendar/answer/6084018

rarely for curators to find in the limited time they have available for curation. Alternatively, many facts may be too apparently insignificant for curators to prioritise. However, automatic processing does not have these limitations, and is used to process these large amounts of unstructured data into a structured format, whether fully automatically or as support to human curators who make final adjudications on what should be extracted.

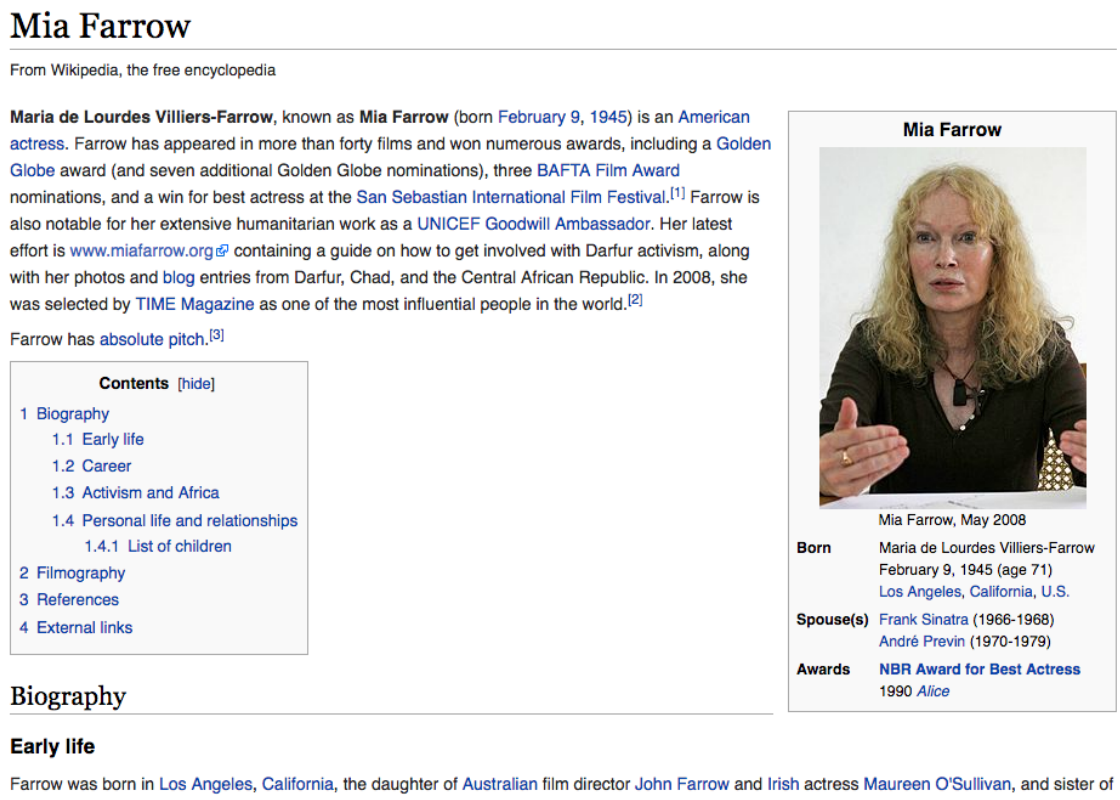
Automatic processing is difficult. Humans can extract many facts with ease. Example 1 shows the first sentence of the Wikipedia article for Mia Farrow.

(1) María de Lourdes "Mia" Villiers Farrow (born February 9, 1945) is an American actress, activist and former fashion model.

This sentence contains a number of facts about Mia Farrow that are immediately obvious: a full name, date of birth, nationality, and three job titles. However, extracting these facts automatically requires a substantial amount of processing and knowledge. Systems need to identify spans of text that are valid to extract as facts; extract context that associates these facts with Mia Farrow and allows us to determine the type of fact (*born* February 9); and make use of knowledge such as that the fact that *American* is a nationality.

Machines have difficulty incorporating this linguistic and world knowledge in order to extract facts. Much work has been done on this problem: this task is often framed as the sentence-level extraction of relations. Facts such as (Mia Farrow, `born`, February 9) are extracted from individual sentences, with little regard to completing a larger KB. The task of slot filling positions this extraction in a more real-world setting, and is a key task for the automatic completion of KBs.

Slot filling (SF) involves extracting facts about particular entities from multiple sources, and merging these facts into an infobox. As an example, Figure 1.1 contains the Wikipedia article for Mia Farrow (from 2008). While there are some facts about this entity available in the infobox which can be leveraged by downstream



# Mia Farrow

From Wikipedia, the free encyclopedia

**Maria de Lourdes Villiers-Farrow**, known as **Mia Farrow** (born February 9, 1945) is an American actress. Farrow has appeared in more than forty films and won numerous awards, including a Golden Globe award (and seven additional Golden Globe nominations), three BAFTA Film Award nominations, and a win for best actress at the San Sebastian International Film Festival.[1] Farrow is also notable for her extensive humanitarian work as a UNICEF Goodwill Ambassador. Her latest effort is www.miafarrow.org containing a guide on how to get involved with Darfur activism, along with her photos and blog entries from Darfur, Chad, and the Central African Republic. In 2008, she was selected by TIME Magazine as one of the most influential people in the world.[2]

Farrow has absolute pitch.[3]

**Contents** [hide]

1 Biography
- 1.1 Early life
- 1.2 Career
- 1.3 Activism and Africa
- 1.4 Personal life and relationships
  - 1.4.1 List of children

2 Filmography
3 References
4 External links

**Mia Farrow**

Mia Farrow, May 2008

| | |
|---|---|
| **Born** | Maria de Lourdes Villiers-Farrow<br>February 9, 1945 (age 71)<br>Los Angeles, California, U.S. |
| **Spouse(s)** | Frank Sinatra (1966-1968)<br>André Previn (1970-1979) |
| **Awards** | **NBR Award for Best Actress**<br>1990 *Alice* |

## Biography

### Early life

Farrow was born in Los Angeles, California, the daughter of Australian film director John Farrow and Irish actress Maureen O'Sullivan, and sister of

Figure 1.1: Except of Wikipedia article for Mia Farrow (October 2008).

applications—such as Mia Farrow's data of birth—there are a great number of facts that are not present. The article itself contains facts that are not present in the infobox. For example, an excerpt from Biography is in Example 2:

(2) . . . the daughter of Australian film director John Farrow and Irish Actress Maureen O'Sullivan . . .

From this text we can extract the `parents` of Mia Farrow. Note that some infoboxes do have `parents`, and in this case the infobox is inconsistent with the article text. Additionally, this is simply from the Wikipedia article itself. From the rest of the web, and other sources of text, we can potentially extract a great deal more facts about Mia Farrow and other entities. For `parents`, only 2% of people in Wikipedia have this slot listed in their infoboxes, despite this attribute being common (the majority of people have known parents). SF uses available unstructured data to

complete these structured KBs, in turn enabling the wide array of downstream applications which require large structured KBs. Traditional sentence-level extraction requires the extraction of relations between all pairs of entities in sentences: every mention of Mia Farrow's parents must be extracted. SF requires only requires one of these relations to be extracted—the goal is that the fact is either extracted or it isn't, and redundant relations are unimportant. SF itself is a difficult task, for many reasons which we will analyse in this work.

## 1.1 Contributions

This thesis considers *slot filling* (SF), the task of extracting values (*filling*) of named attributes (*slots*) of entities from text. In Chapter 2, we review the task of TAC Knowledge Base Population (KBP) slot filling, the primary shared task driving work in SF. TAC SF has its own specific traits as a setting of SF, and we detail the specific requirements of this task, as well as related tasks. We consider the implications of the TAC setting of the task. SF is often described as query-oriented relation extraction (RE): the extraction of relations between pairs of entities in sentences. We will further discuss the relationship between the two tasks in Chapter 2. While RE is a core component of SF, in Chapter 2 and through this thesis we reflect on how SF is a task with many distinct problems. Continuing this, Chapter 3 details approaches to relation extraction (RE), considering how relations are represented by contextual features. Defining a useful representation that is discriminative enough for complex relations is of particular concern to SF. We survey literature for RE, how this task has been defined, and how the definition of this space has influenced SF. Additionally, we survey approaches to SF and how these approaches incorporate RE techniques. We explore some of the general difficulties of the task.

Analysis of these difficulties and consideration of how SF differs from RE continues in Chapter 4, where we contribute a detailed analysis of recall loss in SF systems.

We argue that recall specifically is a major limiter on SF system performance. We precisely analyse where typical SF systems lose recall, and find a substantial amount of loss occurs early in the SF pipeline before RE approaches are applied. We provide guidance to designers of systems in accounting for this loss. In this chapter, we also measure upper bounds on recall using a naïve graph-based semi-supervised reachability approach.

This technique is potentially directly useful as an extraction approach, if sensible constraints are applied, and in Chapter 5 we explore this idea and frame SF as a label propagation task. We focus on creating a detailed graph representation of the task, which reflects the behaviour and assumptions based on results from earlier chapters and experimentation with the graph structure. We apply a label propagation algorithm to a number of configurations of the graph. This baseline label propagation approach has several issues, and we explore a number of ways for improving the process for the task of SF. We identify that allowing all slots to compete in the graph as is standard for label propagation limits performance in the task, and contribute a modification to label propagation to model these types of label interaction in the graph.

One of the larger issues with the graph representation—relating back to the problem of defining a useful, discriminative representation—involves the sparsity of the graph. In Chapter 6 we provide a detailed analysis of the distribution of data used for training and evaluation, and how the distribution of this data affects system performance. We consider that substantial recall is lost due to the construction of our graph, and in particular, we find that a sizable portion of the graph is disconnected. We add more general features to increase connectivity in the graph. Analysis of precision errors reveal that there are instances which require more context to be identified as correct and incorrect, to account for this we add more syntactic information to dependency path features. We also consider sparsity in training data. Overall, this label propagation approach does not achieve state-

of-the-art performance, but modelling SF in this way that enables us to identify a particular set of problems regarding the structure of the task.

We find that lack of a standard across all of RE for how *explicit* expressions of relations must be in text makes consistent annotation difficult. In Chapter 7 we contribute several annotation-driven analyses of this problem, exploring the definition of slots (and relations more generally) and how a concrete definition of explicitness for different applications has not been considered. In particular, explicitness requirements have been treated as the same regardless of the downstream application, but different applications will have different requirements: a system that supports email authoring will likely require less explicit relations than a financial decision-support or legal document retrieval system. We provide a set of considerations about explicitness for future task designers and system implementers to support these applications, as well as important considerations for evaluation of RE as a whole. Finally, we conclude this work in Chapter 8, reiterating that SF is fundamental to enabling many knowledge-based applications, and that this work motivates modelling and evaluating SF to better target these applications.

### 1.1.1 Publications based on this thesis

Parts of this thesis have been reported in conference and workshop proceedings. The recall upper bound analysis making up Chapter 4 appears in:

> Glen Pink, Joel Nothman, and James R. Curran. 2014. Analysing recall loss in named entity slot filling. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 820–830.

The naïve Slot Filling pipeline in Chapter 5 provides the basic structure of the system description in:

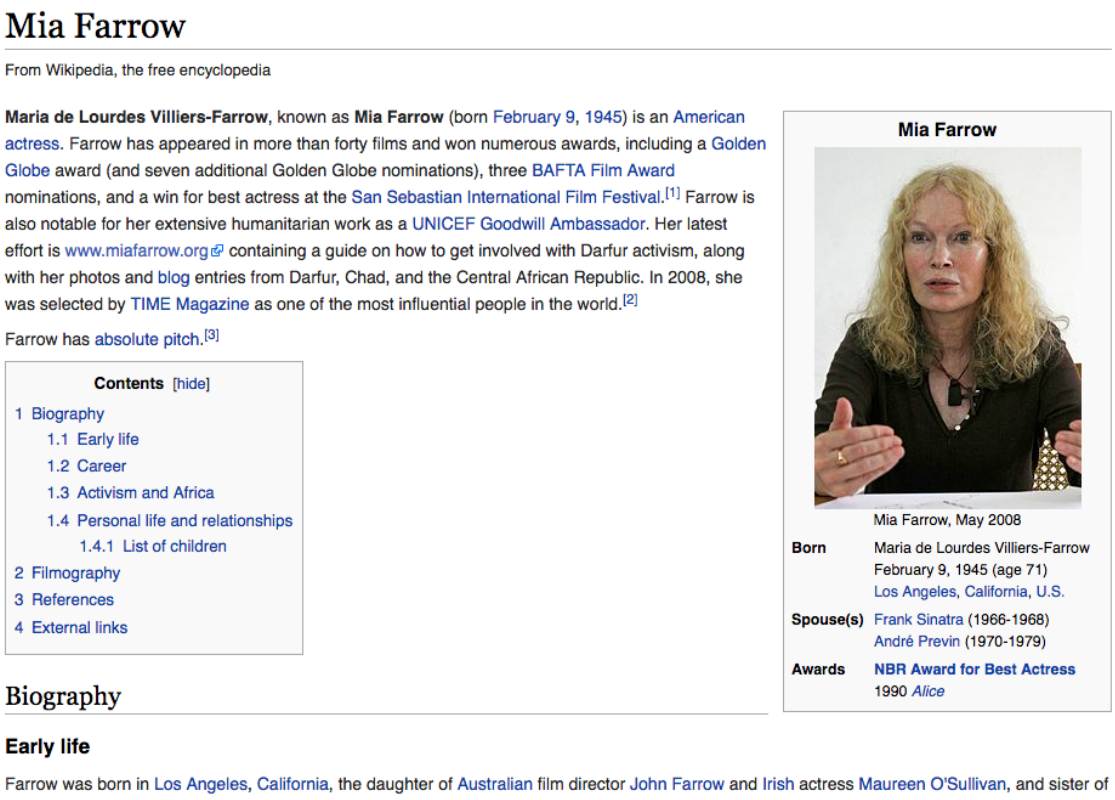
> Glen Pink and James R. Curran. 2014. SYDNEY at TAC 2014. In *Proceedings of the 2014 Text Analysis Conference*.

# 2 TAC KBP slot filling

This work focuses on the Text Analysis Conference Knowledge Base Population (TAC KBP) slot filling (SF) shared task. A *slot* is a named attribute, such as `per:city of birth`. A *fill* is a value of an attribute for a given entity, e.g. for the entity Mia Farrow the fill for `per:city of birth` is Los Angeles. *Slot filling* involves extraction of these attributes of entities from a large corpus of documents, for the purpose of creating or expanding a knowledge base (KB) such as Wikipedia (McNamee and Dang, 2009). The Text Analysis Conference Knowledge Base Population (TAC KBP) is the most explored setting of SF, and we use it as our primary task definition and evaluation in this work. In this chapter, we unpack this high-level description; detail the KB, entities and attributes used in SF; profile the data used for evaluation and the evaluation process; and highlight the impact of these elements on the task. We will identify the difficult aspects of the task, particularly in relation to the well-studied task of relation extraction (RE). We will explore approaches to the task to Chapter 3.

## 2.1 Knowledge Base Population

As SF concerns knowledge base population, we begin with a knowledge base (KB). A key motivator for KBP is to add structured data to a KB such as Wikipedia. The TAC KB is derived from an October 2008 snapshot of English Wikipedia. Wikipedia is comprised of articles such as the Mia Farrow example from Chapter 1, shown again



# Mia Farrow

From Wikipedia, the free encyclopedia

**Maria de Lourdes Villiers-Farrow**, known as **Mia Farrow** (born February 9, 1945) is an American actress. Farrow has appeared in more than forty films and won numerous awards, including a Golden Globe award (and seven additional Golden Globe nominations), three BAFTA Film Award nominations, and a win for best actress at the San Sebastian International Film Festival.[1] Farrow is also notable for her extensive humanitarian work as a UNICEF Goodwill Ambassador. Her latest effort is www.miafarrow.org containing a guide on how to get involved with Darfur activism, along with her photos and blog entries from Darfur, Chad, and the Central African Republic. In 2008, she was selected by TIME Magazine as one of the most influential people in the world.[2]

Farrow has absolute pitch.[3]

**Contents** [hide]

1 Biography
- 1.1 Early life
- 1.2 Career
- 1.3 Activism and Africa
- 1.4 Personal life and relationships
  - 1.4.1 List of children

2 Filmography
3 References
4 External links

**Mia Farrow**

Mia Farrow, May 2008

| | |
|---|---|
| **Born** | Maria de Lourdes Villiers-Farrow<br>February 9, 1945 (age 71)<br>Los Angeles, California, U.S. |
| **Spouse(s)** | Frank Sinatra (1966-1968)<br>André Previn (1970-1979) |
| **Awards** | **NBR Award for Best Actress**<br>1990 *Alice* |

## Biography

### Early life

Farrow was born in Los Angeles, California, the daughter of Australian film director John Farrow and Irish actress Maureen O'Sullivan, and sister of

Figure 2.1: Except of Wikipedia article for Mia Farrow (October 2008).

in Figure 2.1. Extracting additional data for Wikipedia infoboxes about people and organisations such as this example is the focus of TAC. For the purpose of TAC, the article is divided into three primary components: the title, the infobox on the right, and the text (the remainder of the content). The infobox is comprised of key facts about the entity that may, or may not, be present in the text. In this example, the NBR Board of Review Award for Best Actress is not mentioned anywhere in the article, and in this case the infobox is a source of new information from another source of content. Figure 2.1 shows the rendered version[1] of the page—the page is actually represented as MediaWiki markup[2]. Figure 2.2 shows the markup for the infobox, which contains a number of key facts about the entity (dependent on what Wikipedia authors consider to be key for the entity). A large number of

[1]The actual October 2008 version of the article contains a error in the markup which affects rendering, we use the corrected version here.

[2]en.wikipedia.org/wiki/Help:Wiki_markup

```
{{Infobox actor
| image = Mia Farrow.jpg
| name = Mia Farrow
| imagesize =
| caption = Mia Farrow, May 2008
| birthdate = {{birth date and age|1945|2|9}}
| birthplace = [[Los Angeles]], [[California]], [[U.S.]]
| birthname = Maria de Lourdes Villiers-Farrow
| spouse = [[Frank Sinatra]] (1966-1968)<br>[[André Previn]] (1970-1979)
| domesticpartner =
| goldenglobeawards = '''[[Golden Globe Award for New Star Of The Year - Actress
    |Most Promising Newcomer - Female]]'''<br>1965
| awards = '''[[National Board of Review Award for Best Actress
    |NBR Award for Best Actress]]'''<br>1990 ''[[Alice (film)|Alice]]''
}}
```

Figure 2.2: Infobox markup for Mia Farrow (October 2008).

infobox templates are available for editors, who select an appropriate template for an article. These templates are loosely defined and editors can add or remove facts as needed—typically only a few facts will initially be added by editors and these will be added to over time. The Mia Farrow article uses the *actor* infobox, and this infobox includes facts that could apply to any person—such as *birthplace* and *birthdate*—along with more specific actor facts like *awards* won.

For TAC, this MediaWiki markup is converted into XML, as in Figure 2.3 (note that the text is also included). This conversion is intended to reflect the rendered version: elements such as dates are expanded into human-readable formats, and list items are converted to distinct elements. In the case of the Mia Farrow article, `birthdate` is converted from `{birth date and age|1945|2|9}` to `February 9, 1945 (1945-02-09) (age 64)`, and two `spouses` are separated into different items. Two important additions are made to entries as part of this conversion

process. Firstly, entities are mapped to one of four TAC KB types—PER (person), ORG (organisation), GPE (geopolitical entity) or UKN (unknown)—based on the class of the infobox (e.g. *actor* is mapped to PER). Secondly, entities are assigned a TAC KB *id*, and where links to other entities are present, these links are maintained with the new ids.

Only the 818,741 articles in the Wikipedia dump that contained parseable infoboxes are used, and these processed articles make up the TAC KB. At the time of this dump, Wikipedia contained roughly 2.6 million articles, making TAC KB roughly 30% of Wikipedia. Importantly, this means the majority of Wikipedia articles for named entities do not have easily accessible structured facts. Being able to populate these infoboxes with knowledge derived from Wikipedia text or external sources is a key motivation for this task. This also applies to entities beyond those that already have Wikipedia pages. Additionally, even the infoboxes that do make up the KB are incomplete. 83% of people infoboxes have a `place of birth` listed (this includes where place of birth is explicitly Unknown), 9% have a location of `residence`, and 2% have `parents` despite these being common attributes. This does not just apply to people: e.g. only 9% of organisations have a `founder` listed. Clearly, a large amount of structured information is still missing from even those articles that have infoboxes! Automatic KBP can help to solve this problem.

### 2.1.1 Slots

There are a wide variety of Wikipedia infobox templates in the TAC KB, and many define a very specific set of fact types. In the TAC KB, there are 43 PER, 124 ORG, and 53 GPE templates. These templates often contain the same fact type: the *actor* template has a `birth_date` field, as does the *Archbishop of Canterbury* template and 19 other templates. Additionally, different templates contain fields that are named differently but otherwise the same, e.g. 11 templates have the field `children`, 6 royalty templates have the field `issue`, and 1 template has the field `offspring`, but

```
<entity wiki_title="Mia_Farrow" type="PER" id="E0091791" name="Mia Farrow">
<facts class="Infobox actor">
    <fact name="name">Mia Farrow</fact>
    <fact name="birthdate">February 9, 1945 (1945-02-09) (age 64)</fact>
    <fact name="birthplace">
        <link>Los Angeles</link>,
        <link entity_id="E0739132">California</link>,
        <link>U.S.</link>
    </fact>
    <fact name="birthname">Maria de Lourdes Villiers-Farrow</fact>
    <fact name="spouse">
        <link entity_id="E0173926">Frank Sinatra</link> (1966-1968)
        <link>André Previn</link> (1970-1979)
    </fact>
    <fact name="goldenglobeawards">
        <link>Most Promising Newcomer - Female</link> 1965
    </fact>
    <fact name="awards">
        <link>NBR Award for Best Actress</link> 1990 <link>Alice</link>
    </fact>
</facts>
<wiki_text><![CDATA[Mia Farrow

Maria de Lourdes Villiers-Farrow, known as Mia Farrow (born February 9,
1945) is an American actress.
...
  ]]></wiki_text>
</entity>
```

Figure 2.3: TAC KB entry excerpt for Mia Farrow.

**3.6 PER: City of Birth**

***Content:*** Name
***Quantity:*** Single
***Description:*** The geopolitical entity at the municipality level (city, town, or village) in which the assigned person was born. This slot must be filled with the name of a city, town, or village.

- Hong Kong, Macau, Gaza, and Jewish settlements should be classified as cities.
- Capitol Districts (e.g. Washington D.C.) should **NOT** be classified at the city level, rather they should be classified at the state or province level.
- GPEs below the city level (e.g. 5 boroughs of New York City) are **NOT** valid fillers.

| Entity | Document Context | Correct Filler |
|---|---|---|
| Hank Williams | Williams lived in Georgiana in the mid 1930's with his mother, Lillie, and his sister, Irene, after his birth in Mount Olive West | Mount Olive West |
| Tom Lehman | Lehman was born in Austin, Minnesota but … | Austin |

Figure 2.4: Definition for `per:city of birth`.

these all refer to children. These differences are typically created by a lack of consistency in defining templates: ultimately, Wikipedia editors can define infoboxes as they see fit. Finally, some infobox fields are far too rare to be interesting for evaluation: `time in space` is an important fact for an *astronaut*, but not interesting for all other people. Fields this rare are unlikely to appear in evaluation data, and are unlikely be learnable due to lack of possible training data.

In order to make infoboxes more consistent and to provide a more structured KB for evaluation, TAC defines three generic infobox templates, one each for the PER, ORG and GPE entity types. These templates are made up of the key fields for each high-level entity type. The fields which make up these generic infoboxes are *slots*. TAC KB infoboxes are mapped via a rule-based mapping to these slots, e.g. `actor:birthdate` is mapped to `per:date of birth`. The value for a slot, e.g. 1945-02-09 in the case of Mia Farrow's `per:date of birth`, is that slot's *fill*. In this work, we will refer to slot fills as a tuple (entity, slot, fill), e.g. (Mia Farrow, `per:date of birth`, 1945-02-09), or as (slot, fill), e.g. (`per:date of birth`, 1945-02-09) when discussing fills for a particular entity. Overly specific rare fields are not mapped to slots. Where a slot has no fill, it is considered to be filled by a NIL value.

The official definition from Ellis et al. (2012b) for `per:city of birth` is shown in Figure 2.4. As in this example, slots have three parts to their definition: a *content*,

*quantity* and *description*. A slot's *content* is defined as either *name* for slots filled by named entities (NES), *value* for numbers and dates, or *string* for more open-ended responses. Its *quantity* is whether they are filled by a single value (e.g. `per:city of birth`), or a list of values (e.g. `per:children`). Finally, a slot has a *description*, which defines what the slot actually represents, providing a guide for additional annotation and meaningful analysis. The content of this description varies per slot, and often consists of a simple definition and an exploration of edge cases. For `per:city of birth`, most of the specifications relate to the definition of *city* rather than any uncertainty about *birth*. Birth itself is a reasonably well-defined event, both in terms of how much coverage the slot definition has of edge cases, and how consistently birth is expressed in text. For example, in our Mia Farrow article, the date of birth is found in the text in Example 1: dates of birth are frequently expressed in similar, typically unambiguous formats.

(1) Mia Farrow (born February 9, 1945) ...

Other slots are less well-defined, both in edge cases identified in the definition and number of ways in which a fill can be expressed. In the case of `per:cities of residence` in Figure 2.5, there is also uncertainty about what constitutes a city, but what constitutes *residence* is more vague. There is no standard duration, being in prison is counted as residence, and vacation homes must be specifically owned by the entity (and not, say, by a family member) whereas non-vacation homes do not. This set of somewhat detailed (and at times arbitrary) criteria still allows for a substantial amount of uncertainty in some cases: we expect that the cause for this quite specific set of edge case definitions is due to the schema being updated year-to-year based on specific edge cases that arise during the evaluation process (which will discussed in Section 2.1.4). We will further address this uncertainty in slot definition in Chapter 7. Finally and importantly, for this setting of SF, slots have no temporal aspect, and any previously true value is considered to be a valid

**3.15 PER: Cities of Residence**

***Content:*** Name
***Quantity:*** List
***Description:*** Geopolitical entities at the level of city, town, or village in which the assigned person has lived. This slot must be filled with the name of a city, town, or village.

- Former cities of residence are correct responses.
- Residence must be lexically supported in source documents (e.g., "home", "house", "resides", "grew up", etc.) but there is no duration standard to define residence.
- Prison stays and similar legal holdings can support residence fillers.
- Vacation homes can support residence fillers as long as it is clear the residence is owned by the entity and is not just a rental.
- Hong Kong, Macau, Gaza, and Jewish settlements should be classified as cities.
- Capitol Districts (e.g. Washington D.C.) should **NOT** be classified at the city level, rather they should be classified at the state or province level.
- Note that proof of employment in a city does **NOT** justify a residence filler.
- Without other supporting language, birthplace is not sufficient to justify residence. (see Roy Scheider example below for an exception based on other supporting language).
- Unlike countries and states of residence, top-level government employees of cities can **NOT** be inferred to reside in their respective GPEs.
- GPEs below the city level (e.g. 5 boroughs of New York City) are **NOT** acceptable answers.

| Entity | Document Context | Correct Filler |
|---|---|---|
| Abdurrahman Wahid | Abdurrahman returned to his house in Cilandak, Indonesia | N/A |
| Al Gore | The Gore family resides in Nashville, Tennessee | Nashville |
| George W. Bush | US President George W. Bush will meet his French counterpart Nicolas Sarkozy on Saturday at the Bush family's summer home in Kennebunkport, Maine, | Kennebunkport |
| Rudy Giuliani | Former New York City mayor Rudy Giuliani said... | N/A |
| Roy Scheider | Born into a working class family in Orange, New Jersey, Roy Scheider... | Orange |
| Roy Scheider | Scheider lived in Sag Harbor, NY... | Sag Harbor |

Figure 2.5: Definition for `per:cities of residence`.

fill: (Mia Farrow, `per:spouse`, Frank Sinatra) and (Mia Farrow, `per:spouse`, André Previn) are both valid fills despite no longer being true.

The full list of TAC slots as of 2011[3] is given in Table 2.1. There are 42 slots, and while some of these are filled by values that are quite different from the others (e.g. `per:charges` for criminal charges, and `org:website`), many concern facts that characterise particular events or entity states. For example, four slots are of the form `per:* of birth`, and the slots `per:city of birth`, `per:stateorprovince of birth` and `per:country of birth` are particularly close and only distinguished

[3]The only major change to slots after 2011 was the merging of `per:employee of` and `per:member of` into `per:employee or member of` in 2013.

| slot | content | slot | content |
|---|---|---|---|
| per:alternate names | name+ | org:alternate names | name+ |
| per:age | value | org:political/religious affiliation | name+ |
| per:date of birth | value | org:top members/employees | name+ |
| per:city of birth | name | org:number of employees/members | value |
| per:stateorprovince of birth | name | org:members | name+ |
| per:country of birth | name | org:member of | name+ |
| per:date of death | value | org:subsidiaries | name+ |
| per:city of death | name | org:parents | name+ |
| per:stateorprovince of death | name | org:founded by | name+ |
| per:country of death | name | org:date founded | value |
| per:cause of death | string | org:date dissolved | value |
| per:origin | name+ | org:city of headquarters | name |
| per:cities of residence | name+ | org:stateorprovince of headquarters | name |
| per:statesorprovinces of residence | name+ | org:country of headquarters | name |
| per:countries of residence | name+ | org:shareholders | name+ |
| per:spouse | name+ | org:website | string |
| per:children | name+ | | |
| per:parents | name+ | | |
| per:siblings | name+ | | |
| per:other family | name+ | | |
| per:schools attended | name+ | | |
| per:title | string+ | | |
| per:employee of | name+ | | |
| per:member of | name+ | | |
| per:religion | string+ | | |
| per:charges | string+ | | |

Table 2.1: The TAC KBP slots as of 2011. + indicates a list slot that can be filled by multiple values, otherwise a slot is filled by a single value.

| slot | fill |
|---|---|
| per:date of birth | 1945-02-09 |
| per:age | 64 |
| per:city of birth | Los Angeles |
| per:stateorprovince of birth | California |
| per:country of birth | U.S. |
| per:alternate names | Maria de Lourdes Villiers-Farrow |
| per:spouse | Frank Sinatra |
| per:spouse | André Previn |

Table 2.2: TAC KB entry mapped to slots for Mia Farrow.

by geographic granularity. These slots, along with per:* of death, per:* of residence and per:origin (nationality or ethnicity) make up 13 of the person slots, meaning that half of PER slots concern aspects of a person's birth, death, and where they lived. Five more PER slots concern family relationships, and so overall PER slots are about a smaller number of generic events or states than may first appear. ORG slots follow a similar pattern, but to a lesser degree, primarily because there are simply fewer slots: org:date dissolved is the only slot relating to an organisation's "death".

Table 2.2 maps Mia Farrow's TAC KB entry to slots. Note that not all facts from the infobox are represented: facts that are associated with particular entity subtypes, such as awards won, are not included as they are not represented in slots.

We have defined *slots* and *fills*, and can now describe the SF task. For SF, systems are provided with a collection of source documents and a set of query entities, and have to retrieve fills for the slots of those entities. We now detail these components.

### 2.1.2 Source documents

The TAC source documents defined by the task and are mostly newswire and web documents. An excerpt from a newswire document is shown in Figure 2.6, and an excerpt from a web document is included in Figure 2.7. These documents

<DOC> <DOCID> LTW_ENG_20070504.0046.LDC2009T13 </DOCID>
<DOCTYPE SOURCE="newswire"> NEWS STORY </DOCTYPE> <DATETIME> 2007-05-04 </DATETIME>
<BODY> <HEADLINE> Farrow Puts Spotlight on Darfur </HEADLINE> <TEXT>
<P> WASHINGTON DEPOT, Conn. </P>
<P> She's 62 now, as charmingly elfin as ever, and still making
movies. But don't ask Mia Farrow about Hollywood, her famous
marriages or how she manages to stay so trim. </P>
...
Farrow, who lives in nearby Bridgewater, is best known for her
acting career and her various marriages and liaisons with Frank
Sinatra, Andre Previn and Woody Allen, but her trajectory toward
global activism is clear. The daughter of director John Farrow
and actress Maureen O'Sullivan, Farrow had polio as a child and
later adopted a boy...
</TEXT> </BODY> </DOC>

Figure 2.6: Except from document LTW_ENG_20070504.0046.LDC2009T13. For Mia Farrow, (per:age, 62), (per:parents, John Farrow), (per:parents, Maureen O'Sullivan), and (per:city of residence, Bridgewater) are underlined.

```
<DOC> <DOCID> eng-NG-31-142091-9996808 </DOCID>
<DOCTYPE SOURCE="usenet"> USENET TEXT </DOCTYPE>
<DATETIME> 2008-02-08T09:36:00 </DATETIME>
<BODY> <HEADLINE> Maharishi Mahesh Yogi, Spiritual Leader, Dies </HEADLINE>
<TEXT> <POST> <POSTER> ... </POSTER>
<POSTDATE> 2008-02-08T09:36:00 </POSTDATE>
...
The visibility and popularity of the organization can largely be
attributed to the Beatles. In 1968, the band, with great publicity,
began studying with the Maharishi at his Himalayan retreat, or ashram,
in Rishikesh, in northern India. They went with their wives, the folk
singer Donovan, the singer Mike Love, of the Beach Boys, the actress
Mia Farrow and Ms. Farrow's sister Prudence.
... </POST> <POST> ... </TEXT> </BODY> </DOC>
```

Figure 2.7: Except from document `eng-NG-31-142091-9996808`. For Mia Farrow, (`per:sibling`, Prudence) and (`per:title`, actress) are underlined.

have metadata fields which provide a `docid`, document type `doctype`, and time of publication `datetime`. `headlines` are separated from body `text`, but bylines and datelines are not separated from the body text. Note that content has been stripped of most HTML markup (aside from paragraph tags) and unrelated content: web documents are not raw HTML but are the main text context of the web pages from which they are extracted. HTML tags for tables and lists have been removed, but the actual content remains (separated by whitespace). Where web content contained multiple posts (as in discussion forums), this is maintained by `<post>` tags as in Example 2.7.

Reading these example documents, we can immediately identify fills for Mia Farrow: we can extract (`per:sibling`, Prudence) from Example 2 and (`per:city of residence`, Bridgewater) from Example 3, among other fills.

(2) ...the actress Mia Farrow and Ms. Farrow's sister *Prudence*.

| | 2009 | 2010/2011 | 2012 | 2013/2014 |
|---|---|---|---|---|
| broadcast conversation | 17 | 17 | 17 | - |
| broadcast news | 665 | 665 | 665 | - |
| conversation telephone speech | 1 | 1 | 1 | - |
| newswire | 1,286,609 | 1,286,609 | 2,286,866 | 1,000,257 |
| web text | 1,795 | 490,596 | 1,490,595 | 999,999 |
| discussion forums | - | - | - | 99,063 |

Table 2.3: TAC source data statistics.

(3) Farrow, who lives in nearby *Bridgewater*, ...

TAC source document distributions are detailed in Table 2.3. Note that 2009 is a subset of 2010/2011 which is in turn a subset of 2012. This 2009–2012 corpus was retired in 2013, and a new corpus was developed. As TAC has progressed, the ratio of web documents to newswire has substantially increased, and top-performing systems need to account for the web domain, whereas in earlier years a system could focus on newswire and still expect good performance. Discussion forums, added in 2013, provide a more semi-structured set of documents (with forum posts and threads). However, this kind of structure was already present in some web documents, just not as an explicitly distinct domain. The number of documents is large, and techniques for processing the text need to be efficient to be practical.

### 2.1.3 Queries

SF is a query-oriented task. Systems are provided with a set of entities, referred to as query entities. Systems must retrieve fills for the slots of these entities, and are evaluated on the fills they return for these specific query entities. We now detail the structure of these queries. Figure 2.8 shows a query for Mia Farrow. Each query consists of the text of a mention of the entity (`name`); the document in which that mention appears (`docid`); the entity type (PER or ORG), a `nodeid` if the entity is in

```
<query id="SF589">
    <name>Mia Farrow</name>
    <docid>eng-NG-31-141971-9990568</docid>
    <enttype>PER</enttype>
    <nodeid>E0091791</nodeid>
    <ignore>
        per:age
        per:city_of_birth
        per:country_of_birth
        per:date_of_birth
        per:stateorprovince_of_birth
    </ignore>
</query>
```

Figure 2.8: Mia Farrow query from TAC 2011.

the TAC KB (or NIL if not in the KB); and `ignore`, mapped slots which already exist for the entity in the TAC KB and shouldn't be filled (e.g. we already have (Mia Farrow, `per:age`, 64) in the KB).

Note that in this example, Mia Farrow exists in the KB, and so the entity being queried is not ambiguous. In cases where the entity is not in the KB, and the entity is potentially ambiguous (other entities may have the same name as the query entity), systems can use the `docid` document to retrieve an example unambiguous mention. The distribution of queries per year is in Table 2.4, with TAC generally using an even split of PER and ORG entities.[4]

### Query selection

Queries are selected for productivity: at least 2–3 fills for a given entity should be found by manual search of the KB and source corpus. This query selection

[4] GPEs were dropped after 2009, as GPEs had most slots already filled in the KB, and so they did not make for interesting queries. GPEs were added back to the cold start task discussed in Section 2.3 in 2015.

| | PER | ORG | GPE |
|---|---|---|---|
| 2009 train | 24 | 10 | 0 |
| 2009 eval | 16 | 30 | 5 |
| 2010 train | 25 | 25 | - |
| 2010 eval | 50 | 50 | - |
| 2011 eval | 50 | 50 | - |
| 2012 eval | 40 | 40 | - |
| 2013 eval | 50 | 50 | - |
| 2014 eval | 50 | 50 | - |

Table 2.4: TAC query statistics.

process is carried out prior to any other annotation or evaluation. In addition, queries are controlled for ambiguity: `name` strings must be contained in at most a small number of KB entities so as to be broadly unambiguous (Li et al., 2011). This criteria for query selection has varied slightly year-to-year, and will be discussed in Section 2.4.

### 2.1.4 Evaluation

As the source corpus is very large, it is not feasible for annotators to construct an exhaustive KB for evaluation. Instead, the evaluation process is as follows. For a given set of queries, an initial time-limited (2 hour) human annotation is performed by LDC annotators, once the queries have been selected but prior to the official evaluation. This annotation is primarily intended to capture responses that are difficult for systems to find. The results of this annotation and the results of all systems participating in the shared task are then pooled together for evaluation.

Results are then manually marked. While correctness is critical, fills have the additional requirement of being novel with respect to the KB, and with respect to each other. In our Mia Farrow example, we already have the fill (`per:spouse`, Frank Sinatra) in the KB, and so any equivalent fills are redundant. Equivalent fills could be other mentions of Frank Sinatra that exactly match, or mentions like Sinatra

and Frank Albert Sinatra that refer to the same entity. These redundant results are marked separately in evaluation, and are incorrect for the purposes of scoring, IE systems are penalised for failing to normalise and merge extractions. If this initial (per:spouse, Frank Sinatra) fill was not present in the KB, then a system could return any one of these mentions as a correct fill, but any additional mentions would be marked as redundant.

Given these constraints, all pooled results are manually marked as one of:

- **correct**
- **inexact**: result includes part of the correct answer, or contains the correct answer with extraneous text.
- **redundant**: result already exists within the KB, or is redundant with another answer from the same system.
- **wrong**: otherwise incorrect, including a non-NIL result for a NIL.

Correct and redundant answers are assigned to *equivalence classes*, which group different mentions of the same fill, as in our Frank Sinatra example. Equivalence classes are the check for redundancy: systems only get credit for one correct result per equivalence class, additional otherwise correct results are counted as redundant. Note that this result annotation is separate from the original LDC annotation, and the LDC annotation often has some precision errors.

For our Mia Farrow example, we provide sample results from the pooled results from TAC 2011 in Table 2.5. These include all results for our example documents in Figure 2.6 and Figure 2.7. Note that these include correct results from our earlier examples, wrong results where systems have made incorrect extractions, and one redundant result as Frank Sinatra already occurs in the KB. There are also several correct fills that were extracted by no system (including the time-limited human annotation), e.g. there is no extraction of (per:parents, John Farrow) from

| slot | docid | assessment | eqv | fill |
|---|---|---|---|---|
| per:alternate names | † | wrong | - | director |
| per:children | † | wrong | - | John Farrow |
| per:children | † | wrong | - | example |
| per:children | † | wrong | - | music |
| per:children | † | wrong | - | soundtrack |
| per:cities of residence | † | correct | 743 | Bridgewater |
| per:parents | † | correct | 1089 | Maureen O'Sullivan |
| per:parents | † | wrong | - | Darfur |
| per:parents | † | wrong | - | interviewer |
| per:parents | † | wrong | - | shows |
| per:spouse | † | redundant | 1209 | Frank Sinatra |
| per:siblings | ∗ | correct | 1090 | Prudence |
| per:title | ‡ | correct | 1093 | ambassador |
| per:title | ‡ | inexact | - | goodwill ambassador |
| per:member of | ‡ | wrong | - | UNICEF |

Table 2.5: Sample results for Mia Farrow query. *eqv* is equivalence class. *docid*s are removed for presentation: † indicates LTW_ENG_20070504.0046.LDC2009T13 (Figure 2.6), ∗ indicates eng-NG-31-142091-9996808 (Figure 2.7), and ‡ indicates APW_ENG_20080921.0003.LDC2009T13 (Example 4).

Figure 2.6 nor (per:title, actress) from Figure 2.7. We also include three results from an additional document with the relevant excerpt in Example 4:

(4) Farrow, a UNICEF goodwill ambassador, will also visit sites in Port-au-Prince and the hard-hit town of Cabaret during a five-day tour.

This gives us an example of an inexact result, as the per:title fill extracted from this context should be ambassador instead of goodwill ambassador.[5]

These results are pooled from the results of 14 teams and the human annotation. For this particular Mia Farrow query, 433 total fills were submitted. 46 of these fills are marked as correct, over 18 equivalence classes. After marking, recall, precision and F1 are calculated for each system, using the following equations.

$$\text{Correct} = \text{number of correct non-NIL system fills} \tag{2.1}$$

$$\text{System} = \text{number of non-NIL fills} \tag{2.2}$$

$$\text{Reference} = \text{number of correct equivalence classes} \tag{2.3}$$

$$\text{Recall} = \frac{\text{Correct}}{\text{Reference}} \tag{2.4}$$

$$\text{Precision} = \frac{\text{Correct}}{\text{System}} \tag{2.5}$$

$$\text{F1} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \tag{2.6}$$

[5]We note that this fill is consistently annotated for this query, and ambassador is considered to be the correct fill here and given other spans like UNICEF global ambassador. Organisations are not part of per:title fills, so not including UNICEF as part of the fill is clear. However, discarding goodwill is a questionable annotation. The definition for per:title states that *in selecting modifiers to include in title fillers, a general rule is to include them if they describe positions within organizations (e.g., "record producer", "executive producer") and exclude them if they do not (e.g., text excerpts "meteorology professor", and "LGBT activist" would produce the fillers "professor", and "activist")*. Where goodwill falls in the context of UNICEF as an organisation is unclear.

F1 is the official metric for system evaluation. As an example, if the results in Table 2.5 were the pooled reference results (with three correct equivalence classes), and a system returned all of the `per:parents` rows as its only results, its recall would be $0.33$, precision $0.25$, and F1 $0.28$.

## 2.2 Implications of TAC KBP for slot filling

TAC KBP SF is the most explored setting of SF. However, it is only one version of the task, and several aspects will not be the same for other possible versions of SF. At a high level, the goal of SF is to produce a complete KB. The makeup of a complete KB depends on the actual domain, and entities and slots will vary depending on the task. TAC attempts to capture a variety of domains and slots, and approaches to the task are mostly agnostic to the underlying slots: we expect that different slots would be treated in similar ways to the existing slots, but changing domains may introduce different challenges.

In practice, SF systems will not produce complete KBs, and so the choice of evaluation metric is important for comparing approaches to the task. TAC makes use of a micro-averaged F1. This has two important implications. The first is the micro-averaged aspect: a small number of frequently occurring slot types dominate the set of fills. Systems can focus on performing well on these slots—mostly ignoring the infrequent slots—and achieve relatively high results. However, extracting fills in the long tail of infrequent slots may require a different set of techniques than for frequent slots, and the choice of micro-averaged score directs attention away from these.

The second implication is caused by the use of F1 as an evaluation metric. This metric encourages system designers to balance between precision and recall. However, there is no requirement that a definition of SF includes this balance. In a real-world setting, it may be the case that a SF system should prioritise 100% recall,

e.g. if all output is going to be passed to human curators that will validate output. Other settings could require 100% precision. These different settings may require different techniques than those focused on a balanced F1.

With the caveats in mind, we still see TAC SF as an very informative evaluation, that chooses reasonable parameters for motivating work on the broader SF task. We will focus on the TAC setting for most of this work, and return to considering potential issues in TAC in Chapter 7.

## 2.3 Related tasks

We have now detailed the core task of SF. To complete this description, we need to discuss the changes made to the task year-to-year. To motivate these changes, we first need to describe three tasks closely related to SF: relation extraction, named entity linking, and cold start.

SF is often described as query-oriented *relation extraction* (RE). RE is the task of extracting semantic relations between entities in text, typically in sentences. From our early text, repeated in Example 5, a RE system might extract the relation (Mia Farrow, `sibling`, Prudence). This extraction is the core component of SF, and almost all SF approaches are built around RE techniques. We will provide a comparison between the two tasks in Section 2.5, and survey RE in detail in Chapter 3.

(5) They went with their wives, the folk singer Donovan, the singer Mike Love, of the Beach Boys, the actress *Mia Farrow* and Ms. Farrow's sister *Prudence*.

*Named entity linking* (NEL) is the task of disambiguating entity mentions by linking to a KB: in the context of TAC KBP, NEL is the other main sub-task alongside SF. An NEL system must link mentions of entities to specific KB entities. Examples 6 and 7 are the first sentences of Wikipedia articles for entities called John Howard. Given

a mention of John Howard in text, an NEL system must link that mention to the correct entity (or a different John Howard altogether).

(6) John Winston Howard, OM, AC (born 26 July 1939), was the 25th Prime Minister of Australia, serving from 11 March 1996 to 3 December 2007.

(7) John Howard (born 22 October 1952) is an Australian stage and screen actor.

Identifying fills for the correct entity is a fundamental concern for SF—using Example 7 to fill slots for the entity in Example 6 is incorrect. However, the TAC KBP SF query selection process has typically targeted entities that are unambiguous. In the case of our Mia Farrow example, other "Mia Farrows" are rarely (if ever) mentioned in text.

Both TAC KBP entity linking and slot filling assume a pre-existing KB. TAC KBP *cold start* removes this assumption, and requires the use of both sub-tasks to build a KB from scratch from a collection of documents.

## 2.4 Changes to TAC KBP slot filling

For completeness as a survey of TAC SF, we detail the remaining key details of the SF task, and we now detail changes to the SF over the years that it has been run. Except for the changes from 2009 to 2010, many of these changes are subtle, but we include these changes for completeness. In this thesis, we will focus on the setting from 2010 to 2012 as we have described—the only difference from 2012 to 2014 that affects later chapters is the merging of `per:employee of` and `per:member of`, and we will discuss this where relevant. In 2015, SF officially became a sub-task of cold start. The task of SF within cold start remained almost the same (aside from a substantially smaller document collection), but in this work will focus on SF as distinct from cold start.

### Changes from 2009 to 2010

The task has been run once a year since 2009, and has existed in a similar format each year. The original setting for the task was most substantially different to the following years. The key changes from 2009 to 2010 were:

- In 2009, SF queries were selected as a subset of NEL queries, which were deliberately selected as ambiguous names. Since 2010, SF query selection has been separate, minimising the significance of name disambiguation for SF.
- The evaluation metric was changed from accuracy over all slots to F1 over filled slots. As NIL fills are very frequent, returning NIL for every fill as a baseline outperformed all systems in 2009 under the accuracy metric.
- The size of the corpus was expanded (particular the web portion), introducing a wider variety of text formats and topics to the task.
- GPEs were removed as a query type.
- Optional NEL of slot fills was removed from the task: from 2010 fills are all returned as text.
- Each location slot was split into three granularities: e.g. in 2009 there was only a `org:headquarters` slot, this was replaced with `org:city of headquarters`, `org:stateorprovince of headquarters`, and `org:country of headquarters`.

### Changes to the KB

From 2014, TAC KB was no longer used as the reference KB (Surdeanu and Ji, 2014). Live (at time of query creation) Wikipedia was used in its place as SF input formats were brought into line with cold start input formats. Use of an updated version of Wikipedia primarily changed the entities and facts in the KB, requiring little change to techniques for handling redundancy as changes to the schema were minimal.

### Changes to slots

Most changes to the slot schema have been relatively minor, and are as follows:

- In 2010, `per:origin` was switched from a single slot to a list slot. The same change was made to `per:religion` in 2013.

- In 2013, `per:employee of` and `per:member of` were merged into the slot `per:employee or member of` due to their similarity. Distinguishing between these slots is difficult, but potentially useful for some applications. Nevertheless, they were merged, as this distinction was determined to be not important enough to warrant penalising systems for making the incorrect decision. We note that this decision is often about subtyping the ORG entity, often not-for-profit entities will have members and for-profit entities will have employees (but this is not a rule).

- From 2013, `per:title` fills at different organisations are considered to be different fills even if they are the same title: i.e. for Mitt Romney, CEO at Bain Company, CEO and Bain & Company, and CEO at 2002 Winter Olympics are three different CEO fillers. Note that the fill text itself is still just CEO and does not include the company names. Prior to 2013, they would be considered to be one single CEO equivalence class, in line with other slots.

- Several slots have had minor name changes, e.g. `per:stateorprovinces of residence` was changed to `per:statesorprovinces of residence` (`state` became `states`).

- From 2015, as the task officially became a sub-task of cold start, a number of slots were added as inverses of existing slots, such as `gpe:births in city` as an inverse for `per:city of birth`. However, this only affects SF when framed as a sub-task of cold start.

### Changes to queries

Offsets for the entity mention (`name`) were added to queries in 2012 to distinguish between identical mention texts in the same document. The `nodeid` and `ignore` fields were removed in 2014 to reduce overhead for new TAC entrants in handling the TAC KB (Surdeanu and Ji, 2014): as TAC KBP moved focus towards cold start, detecting redundancy with the TAC KB became less of a focus.

### Changes to query selection

From 2012 entities were preferred if potential fills included infrequently filled slots, such as `per:charges` (Ellis et al., 2012a). NEL requirements were also increased: from 2013, the requirement for queries to be relatively unique in the TAC KB was removed (Ellis et al., 2013), and in 2014, ten deliberately ambiguous names were included as part of the set of queries (Li et al., 2014). It is likely that systems could still perform well without handling ambiguity, particularly as these queries only made up a small portion of the total set. However, this did make ambiguity handling an explicit requirement for systems.

### Changes to evaluation

The requirement of offsets for fill justification—a span of text which provides contextual evidence for a fill, such as the sentence it occurs in—was added in 2012. However, these justifications were not actually used for evaluation until 2013, at which point, to be considered correct, a fill had to be both correct and have a valid justification. Basic inference of slots, as distinct for direct extraction, was allowed from 2014, and slots could be inferred provided that they could be justified in the corpus. For example, from the texts Bob was born in Paris and Paris is the capital of France (Bob, `per:country of birth`, France) could be extracted. However, these fills would have to *both* be found in the corpus: a system could not use world

knowledge to know that Paris is the capital of France. From 2014, redundant results are ignored for the purposes of evaluation (as the TAC KB is no longer used).

## 2.5 SF and RE

We have now detailed the task of TAC KBP slot filling. So far, we have mostly discussed examples which have focused on extracting slots or relations between entities in sentences: effectively this is RE. However, SF has several differences from RE, some additional complications, and some compensations for these complications. Before we consider approaches to both tasks in Chapter 3, we first identify these differences.

SF is often described as query-oriented RE. Traditional RE requires the extraction of relations between all pairs of entities in sentences, independent of a larger KB and regardless of whether relations are redundant. For a set of equivalent, redundant relations in text, RE requires that all are extracted. SF only requires one of these relations to be extracted (for a given query). A system can still get full credit for a fill by only extracting the most obvious expression, and not other more difficult to extract expressions. However, extracting with respect to a KB adds normalisation, merging and redundancy identification requirements, adding additional complexity to SF, and we will discuss these requirements below.

In addition, in some cases the the RE component of SF is more complex than typical RE tasks. RE has been primarily studied using pairs of mentions in a single, isolated sentence. SF changes this focus to extracting facts from a collection of documents, and extending a KB with these facts. From 2014, inference across documents is allowed, making multi-document extractions possible. We expect that these document-level or multi-document-level extractions may make for a more difficult task than traditional RE, particularly where beyond sentence-level fills would not be considered candidates for extraction in RE.

#### RE in SF

SF systems use RE to extract facts for KB population. In Example 8, we have two entity mentions Mia Farrow and Prudence in a sentence, and can identify `sibling` relation between the two. This is equivalent to extracting (Mia Farrow, `per:sibling`, Prudence): a (query, fill) pair is just an entity pair for the purposes of RE. We note that SF is query-based and this fill is directional: (Prudence, `per:sibling`, Mia Farrow) is considered to be a separate fill for the purposes of evaluation. This separation is typical for RE evaluations, as many relations are inherently directional, such as `child of` or `located in`.

(8) They went with their wives, the folk singer Donovan, the singer Mike Love, of the Beach Boys, the actress *Mia Farrow* and Ms. Farrow's sister *Prudence*.

#### Extra-sentential relations

Document-level and multi-document-level extraction is required for SF. This requires both NEL and coreference resolution. For the above example, extracting `per:sibling` for a query about Prudence Farrow would be substantially more difficult than Mia Farrow: resolving Prudence to Prudence Farrow is first required. In a large corpus this may be difficult, as a system would need to resolve every possible Prudence to the KB to identify that this mention refers to a Prudence Farrow query. Indeed, in this particular example, there is no guarantee that Prudence is Prudence Farrow except by the relationship with Mia Farrow itself. In RE evaluations this disambiguation is not required. All that is required is that the relationship between the mentions of Prudence and Mia Farrow is extracted, given the sentence context.

Many of these cases are potentially resolvable by coreference resolution (coref). Example 9 requires that we identify that Farrow refers to an earlier Mia Farrow before we can extract (Mia Farrow, `per:city of residence`, Bridgewater). Pipeline

error from use of coref (or the missing of fills if coref is not used) directly affects SF evaluation, but not sentence-level RE.

(9) But don't ask *Mia Farrow* about Hollywood, her famous marriages or how she manages to stay so trim. ...
*Farrow*, who lives in nearby *Bridgewater*, ...

Document-level requirements extend even beyond what is typically captured by NEL and coref. Example 10 contains a particularly difficult extraction of fills for `per:children` for the query John Negroponte. None of the children appear in the same sentence as a mention of Negroponte. These type of instances are not considered as RE candidates.

(10) "Who's that?" *Negroponte* said, calling out to a creak on the stairs. "Hellllllo?"
He paused. Another creaking sound. "Alejandra?"
A young woman peeked into the living room. "Where's George?" said *Alejandra*, 23. *George*, 17, appeared. Then *Sophia*, 13, and *John*, 19.
Four of the five *Negroponte* children were at home. They drifted in and out of the living room, onto the couch, and into the conversation.

**Semi-structured and unstructured data**

Slots can be filled from unstructured, semi-structured and structured data, where these different forms exist in documents. These can include list or table structures, like the `per:date of birth`, `per:city of birth` and `per:country of birth` fills for Lewis Hamilton in Example 11 (Min and Grishman, 2012).[6] This is not unique to SF, as other RE evaluations are over different data formats. Additionally, semi-structured and structured data may make for easier fact extraction, if this formatting can actually be identified in documents that have been stripped of most HTML

[6]We note that this text does not contain explicit marked-up table structure, however this is likely the result of HTML tag stripping. The text itself is more structured than typical sentences.

markup. However, the key difference is that SF explicitly requires analysis of different text formats: this is not often the case for RE.

(11) *Lewis Hamilton*
BORN: *Jan. 7, 1985*, *Stevenage*, *England*.
F1 DEBUT: 2007 Australian GP with McLaren (finished third).

**Grounded entities**

For SF, entities are to be grounded in a KB, and mentions must be resolved to the KB for accurate extraction, as we have discussed in our above comparison with sentence-level extraction. Only relations that are novel with respect to the KB should be extracted. Handling redundant extractions particularly affects evaluation: an RE system may correctly extract ten relations, but if these all refer to the same fact, this is only one correct slot fill. Systems which perform very well on extracting sentence-level relations may not necessarily score well, if they have a poor normalisation and merging processes. If a poorer RE system extracts more novel instances, or has better normalisation and merging processes, it may score higher, even with fewer correct sentence-level extractions overall. Nevertheless, we expect that good sentence-level results are a necessary but not sufficient condition for good SF performance.

**Cross-extraction inference**

SF systems must account for contradictory extractions, and other forms of interaction between extractions and the KB, such as location inference and event causality (Min and Grishman, 2012). Min and Grishman identify several cases where complex inference is required, such as (National Christmas Tree Association, `org:members`, River Ridge Tree Farms) in Example 12.

(12) First lady Laura Bush kicked off the holiday season Sunday by standing out in the rain to receive this year's White House Christmas tree.

> Jessie Davis and Russell Estes, owners of *River Ridge Tree Farms* in Crumpler, North Carolina, where the tree was grown, joined the first lady, along with their families.
>
> ...
>
> The *National Christmas Tree Association* has presented the official White House tree since 1966.
>
> Members of the association compete in state and regional competitions to become eligible to take a tree to the national contest. River Ridge was named grand champion in the National Christmas Tree Contest in August.

**Automatic NLP pipeline** SF requires pipeline components that can create a substantial amount of pipeline error, such as coreference resolution (which itself often requires parsing), and errors in this pipeline compound upon SF errors. Errors in basic components such as sentence splitting also contribute to pipeline error, but this is a general problem across RE.

**General technical considerations** SF techniques must also address practical concerns for evaluation: provenance and justification of extracted facts must be maintained, and systems must be able to scale to process an entire large corpus.

## 2.6 Slot distribution

Fills are not evenly distributed across slots and several slots, mostly employment related, dominate the fills. Systems optimising for these few slots could potentially perform better than a more general-purpose system. The distribution of fills for each year are given in Table 2.6. Note that we include `per:member of` as part of `per:employee of` for the 2013–2014 and the aggregate columns. We see that fills are particularly skewed to a small number of slots, and a system focusing on this group can outperform approaches that may do better on the tail. The slots

per:title and org:top members/employees make up 25% of fills alone. Note that these particular slots are often expressed in formulaic fashion, e.g. in apposition. With per:employee of, per:member of, org:alternate names, per:cities of residence, per:children and per:age, these eight slots make up 51% of fills, with the remaining 49% distributed across 34 slots.

It is not clear what correlation there is between slots that are potentially easier to extract and slots that are more frequently represented in TAC. Fills for slots such as per:title and org:top members/employees are often found in very close context to entities and are relatively easy to consider as candidates. In this particular case, the often occur together: consider U.S. president Barack Obama. While these may still be difficult for a system to extract if it cannot identify the types of the entities (or that president indicates employment), a human annotator can easily resolve this ambiguity, and so such fills are likely to be included in at least the human annotation. As results are compiled from what systems can extract, it is possible that other fills in more difficult contexts may exist but are never extracted. The human annotation goes some way to addressing this, but this is a time-limited annotation and it is not entirely clear what the distribution of fills is in a larger setting. Additionally, annotators and systems discard fills that are redundant but otherwise correct, limiting the number of results that are evaluated and made available as additional data.

Some slots are so infrequent that they are particularly difficult to learn by any approach: the most extreme example, org:date dissolved only occurs 9 times in five years. This may be easy to learn if there was no variation in the expression of this slot, but this is not the case: as an example, Example 13 contains the single TAC 11 org:date dissolved fill (Badr Organization, org:date dissolved, 2004), which is not a trivial extract even for humans.

| | 2010 | 2011 | 2012 | 2013 | 2014 | total | % | Σ % |
|---|---|---|---|---|---|---|---|---|
| per:title | 149 | 201 | 224 | 142 | 139 | 855 | 15 | 15 |
| org:top members/employees | 124 | 118 | 177 | 116 | 83 | 618 | 11 | 25 |
| per:employee of | 71 | 71 | 66 | 72 | 66 | 538 | 9 | 35 |
| per:member of | 44 | 47 | 101 | - | - | - | | |
| org:alternate names | 58 | 98 | 91 | 82 | 34 | 363 | 6 | 41 |
| per:cities of residence | 43 | 17 | 61 | 51 | 41 | 213 | 4 | 44 |
| per:children | 13 | 17 | 100 | 52 | 27 | 209 | 4 | 48 |
| per:age | 49 | 16 | 27 | 51 | 52 | 195 | 3 | 51 |
| per:alternate names | 38 | 46 | 44 | 45 | 14 | 187 | 3 | 55 |
| org:subsidiaries | 53 | 32 | 59 | 25 | 11 | 180 | 3 | 58 |
| per:statesorprovinces of residence | 24 | 11 | 46 | 28 | 32 | 141 | 2 | 60 |
| per:origin | 28 | 23 | 27 | 32 | 28 | 138 | 2 | 62 |
| org:country of headquarters | 23 | 22 | 25 | 34 | 32 | 136 | 2 | 65 |
| per:schools attended | 16 | 16 | 53 | 27 | 13 | 125 | 2 | 67 |
| per:countries of residence | 14 | 20 | 20 | 36 | 29 | 119 | 2 | 69 |
| per:cause of death | 3 | 3 | 40 | 47 | 18 | 111 | 2 | 71 |
| per:charges | 11 | 15 | 9 | 45 | 31 | 111 | 2 | 73 |
| org:members | 17 | 8 | 38 | 22 | 24 | 109 | 2 | 75 |
| org:city of headquarters | 30 | 19 | 13 | 24 | 21 | 107 | 2 | 76 |
| per:parents | 23 | 3 | 30 | 25 | 23 | 104 | 2 | 78 |
| per:spouse | 15 | 8 | 34 | 28 | 16 | 101 | 2 | 80 |
| org:stateorprovince of headquarters | 26 | 17 | 16 | 20 | 21 | 100 | 2 | 82 |
| org:website | 16 | 14 | 18 | 32 | 15 | 95 | 2 | 83 |
| per:date of death | 1 | 4 | 23 | 48 | 14 | 90 | 2 | 85 |
| org:founded by | 13 | 7 | 14 | 21 | 23 | 78 | 1 | 86 |
| org:date founded | 14 | 6 | 14 | 13 | 30 | 77 | 1 | 88 |
| per:siblings | 22 | 6 | 27 | 11 | 10 | 76 | 1 | 89 |
| per:other family | 28 | 6 | 14 | 15 | 9 | 72 | 1 | 90 |
| org:parents | 15 | 24 | 11 | 13 | 5 | 68 | 1 | 91 |
| per:city of death | 1 | 1 | 20 | 35 | 11 | 68 | 1 | 92 |
| org:shareholders | 5 | 18 | 13 | 17 | 3 | 56 | 1 | 93 |
| per:date of birth | 11 | 3 | 21 | 16 | 5 | 56 | 1 | 94 |
| org:number of employees/members | 15 | 6 | 11 | 12 | 9 | 53 | 1 | 95 |
| per:city of birth | 9 | 6 | 16 | 12 | 5 | 48 | 1 | 96 |
| per:stateorprovince of death | 1 | 0 | 17 | 18 | 8 | 44 | 1 | 97 |
| per:stateorprovince of birth | 8 | 1 | 13 | 10 | 5 | 37 | 1 | 97 |
| per:religion | 4 | 5 | 7 | 9 | 9 | 34 | 1 | 98 |
| org:political/religious affiliation | 8 | 2 | 13 | 1 | 6 | 30 | 1 | 99 |
| per:country of birth | 9 | 3 | 3 | 5 | 7 | 27 | 0 | 99 |
| org:member of | 2 | 11 | 9 | 4 | 0 | 26 | 0 | 99 |
| per:country of death | 0 | 1 | 1 | 10 | 12 | 24 | 0 | 100 |
| org:date dissolved | 3 | 1 | 3 | 0 | 2 | 9 | 0 | 100 |
| total | 1057 | 953 | 1569 | 1306 | 943 | 5828 | 100 | |

Table 2.6: Distribution of fills over slots from 2010-2014.

(13) A parliamentarian from the SIIC denied that the *Badr Organization* was a militia. The lawmaker, Layla Kafaji, said it had once existed as one but had been dissolved in *2004* in accordance with ...

Other infrequent examples tend to fluctuate in number substantially, even in later years where entities with fills for infrequent slots were preferred: `per:city of death` occurs from 1 to 35 times depending on the year. Infrequency of particular slots is somewhat self-reinforcing. Poor recall on infrequent slots does not particularly affect overall system performance. Hence, these slots are not targeted by system developers, and substantial additional results do not become part of the pooled evaluation. This is particularly influenced by the choice of micro-averaged F1 as the evaluation metric. A micro-averaged metric would influence the focus on infrequent slots. In turn, these slots are not made available as training data for future years.

We will further explore the impact of these factors in Chapter 4.

## 2.7 Summary

TAC KBP slot filling is the core task of this thesis. In this chapter, we have detailed SF, establishing the foundation for the rest of this work. SF requires that a system use a large collection of documents to fills a pre-determined schema of slots for a set of query entities. These query entities may or may not exist in the TAC KB, but whatever the case, systems must merge and filter redundant results extracted from text. We have discussed the challenges that the task introduces, and contributed a high-level discussion of difficult aspects of SF. In particular, we have addressed why SF is a perhaps a more difficult setting of standard RE, with document-level extraction and query entity resolution requirements.

We will now review approaches to SF in Chapter 3, starting with a broad review of RE before surveying approaches to TAC.

# 3 Background

Successful slot filling (SF) systems have made use of a wide variety of machine learning and data acquisition techniques. Much of their implementation has focused on the relation extraction (RE) component of SF. Indeed, SF itself is often described as query-oriented RE. Before we can analyse and improve upon existing SF pipelines, we first need to have an understanding of the broader background of RE. In this chapter, we explore this background, the impact that RE tasks, data and annotation have had on SF, and the techniques applied to both tasks.

As we introduced in Chapter 2, RE is the task of extracting semantic relations between arguments mentioned in text. In Example 1 we can identify that Sean Ross in an `employee of` Edison Media Research:

(1) "There are a lot more Internet stations. There are a lot more former professionals doing Internet stations," said *Sean Ross*, the vice president of music and programming for *Edison Media Research*.

RE has been significantly driven by shared tasks. TAC itself inherits aspects from its predecessor, the Automatic Content Extraction (ACE) program, which in turn followed from the Message Understanding Conferences (MUC). Outside these shared tasks, different definitions of relations and arguments has resulted in a number of distinct RE tasks. In this chapter, we will discuss approaches to RE across these different tasks.

## 3.1 Representation

We first introduce some definitions. We refer to the the above tuple (Sean Ross, `employee of`, Edison Media Research) as a *relation tuple*, of the form $(e1, r, e2)$. $r$ is a *relation type*, also referred to as a *label* (essentially equivalent to a *slot*). $e1$ and $e2$ are the arguments that exist in a *relation*: they are either *entities* if the relation is a *fact* in a knowledge base (KB), or *mentions* of those entities if in text (in the above example, the mention of Edison Media Research refers to a real-world Edison Media Research entity). We refer to this argument pair as an *entity pair* or *mention pair* respectively. The *context* of a relation tuple is the text from that relation tuple is extracted, and this context provides justification for the extraction. It is important to note that while TAC relies on the use of named entities (and many of our examples will be relations of this type), entities do not have to be named in many settings. For example, in Example 2, CEO is not a named entity.[1]

(2) **relation:** (CEO, `employee of`, Microsoft)
**context:** . . . the *CEO* of *Microsoft* . . .

We now discuss the representation of relation context, using Example 1 as a motivator. In this case, our relation type is `employee of`. However, the representation of a relation context is typically independent of the relations being targeted. We will cover various definitions for `employee of` in Section 3.3, for now we will use a loose dictionary-style definition for `employee`: a person paid to do work.

For clarity, we will discuss only the portion of Example 1 relevant for the relation, as the direct quote in the sentence is not relevant. This gives us Example 3.

[1] For TAC, CEO may resolve to a query entity, and this would be a valid extraction in this case, but for many RE tasks this resolution is not required.

(3) **relation:** (Sean Ross, `employee of`, Edison Media Research)
**context:** . . . said *Sean Ross*, the vice president of music and programming for *Edison Media Research*.

A human reading this context automatically identifies Sean Ross as a person and Edison Media Research as an organisation. They identify that Sean Ross is a vice president for the organisation and, from a general knowledge understanding of roles in organisations, know that a vice president is an employed position. Hence, a reader can extract the relation (Sean Ross, `employee of`, Edison Media Research). Representation of this relation context, and the relation itself, requires the encoding and labelling of this understanding.

In a typical setting, entity spans are labelled prior to RE. This may be done by automatic named entity recognition (NER) or mention detection, and may include coreference resolution of mentions or named entity linking (NEL) of entities to a KB. This labelling may provide mention types, which are often critical for RE. A typical labelling will include PER, ORG and LOC (location, including GPEs) types, in our example Sean Ross (PER) and Edison Media Research (ORG). Mentions of entities that to not fall into PER, ORG or LOC may also be included (depending on the entity schema), such as labelling vice president of music and programming as a ROLE.

For extraction, RE will then consider potential relations between all pairs of entities: in this case we have potential relations for (Sean Ross, Edison Media Research), (vice president of music and programming, Edison Media Research) and (Sean Ross, vice president of music and programming). For the purpose of `employee of`, we can discard the pairs that are not of the form (PER, ORG) or (PER, LOC): a ROLE (vice president of music and programming) cannot be an employee or employer.[2]

We now have a mention pair (Sean Ross, Edison Media Research). The relation itself is represented by syntactic and semantic features of the context. Designing a

[2]In practice, it may be better to not immediately discard these cases if error in entity typing is a major concern, but this filtering is almost always used. We will explore this in detail in Chapter 4.

| type | pattern |
|---|---|
| sequence | PER , vice president of music and programming for ORG |
| | PER , NN+ of NN and NN for ORG |
| | PER * vice president * ORG |
| | PER * vice president * for ORG |
| | PER * president * ORG |
| syntactic (constituency) | PER $\leftarrow$ NP $\rightarrow$ NP $\leftarrow$ PP $\leftarrow$ NP ORG |
| syntactic (dependency) | PER $\overset{\text{nn}}{\longleftarrow}$ president $\overset{\text{prep for}}{\longrightarrow}$ ORG |
| | PER $\overset{\text{nn}}{\longleftarrow}$ NN $\overset{\text{prep for}}{\longrightarrow}$ ORG |

Table 3.1: Examples of sequence and syntactic patterns.

discriminative but statistically reliable representation of relations is of key importance to RE, and this representation can differ depending on the types of relations that are targeted. However, there is a broad space of features that approaches use to represent relations. We detail those now.

## 3.2 Features

One of the most straightforward methods for representing relations is to define contextual patterns. They are based on lexical or syntactic features of the text. Lexical features can simply be the contextual words themselves, but syntactic features require more sophisticated analysis. A constituency parse of Example 3 is given in Figure 3.1 and a dependency parse (using Stanford dependencies (de Marneffe and Manning, 2008)) is given in Figure 3.2. Both forms provide syntactic trees which can be used as context and both provide POS tags.

Table 3.1 lists a number of example patterns for Example 3. Sequence paths are presented with and without POS tags, with POS tags providing a high-level syntactic abstraction over individual tokens. This higher level of abstraction does not necessarily have to be provided by POS tags: any abstraction that clusters tokens under a label can be used. Wildcards (*) allow for patterns to generalise over more

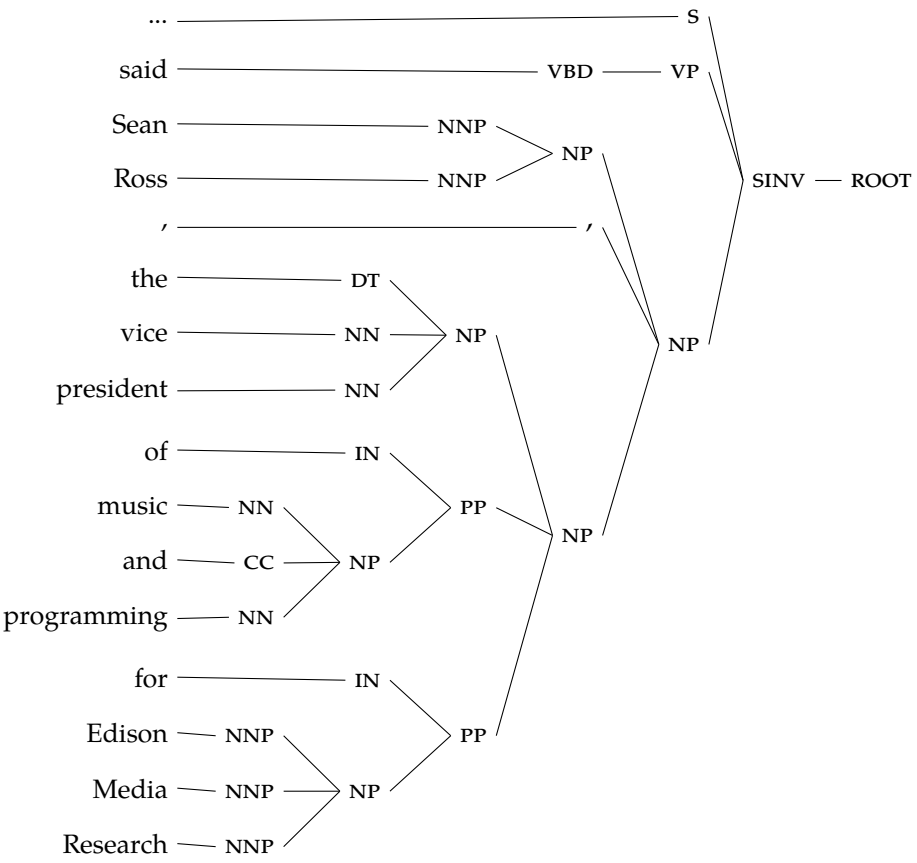



Figure 3.1: Constituency parse for Example 3.

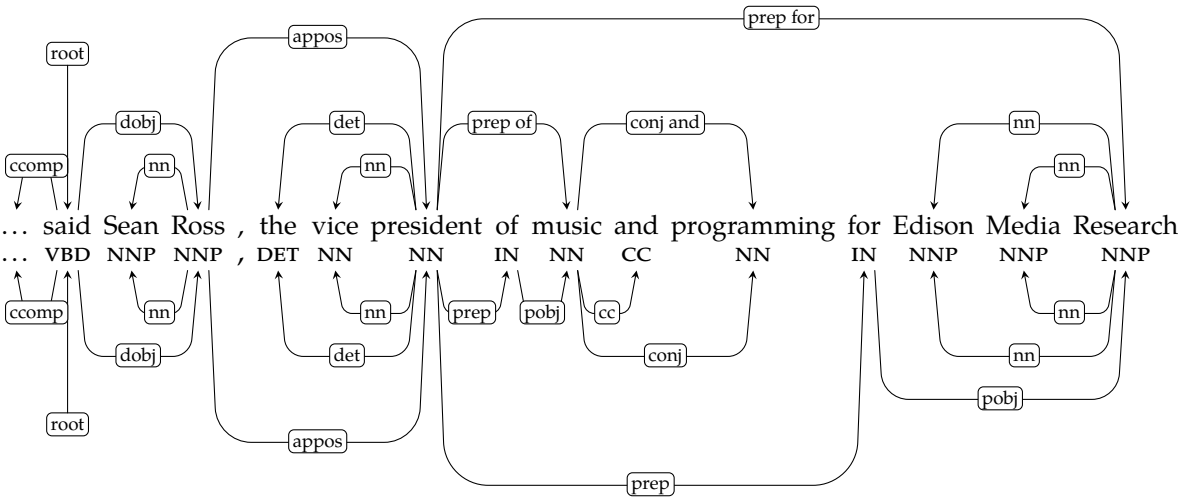



Figure 3.2: Dependency parse for Example 3. Bottom parse uses Stanford basic dependencies, top parse uses Stanford collapsed and propagated dependencies.

instances, providing a simpler representation, but with lower discriminability. Representing Example 3 with the pattern PER * vice president * for ORG will make it equivalent to useful contexts (e.g. PER is the vice president of engineering for ORG), but also to erroneous contexts (e.g. PER is not the vice president for ORG). However, these less specific features create a denser feature sparse, and a smaller number of patterns will be useful for representing a larger amount of data. Syntactic patterns are typically shortest paths from one entity to the other in a constituency or dependency parse tree, potentially with extra token context on either end of the path and wildcards. Syntactic structures provide a level of abstraction over a sequence that may be more useful than patterns of the sequence itself. In our example, [ PER $\xleftarrow{\text{nn}}$ president $\xrightarrow{\text{prep for}}$ ORG ] is a useful abstraction for representing any president of an organisation. However, parsing is relatively computationally expensive, and parse errors may contribute to an inaccurate representation.

While these patterns provide a straightforward representation, a combination of simple features is often used to improve generality. Features can include patterns; components of these patterns such as bag-of-words and $n$-gram features in the context; and argument features such as the types of the entities. A potential feature set for Example 3 is in Table 3.2. These features are of varying levels of discriminability. Some features are not very discriminative (e.g. the token music), and it is the combination of these features that is required to represent this relation.

Finally, different patterns and features are in practice derived from automatic processes. Pipeline errors can cause representations of particular instanced to be erroneous. Deeper parse features may be more discriminative, but parse errors will be propagated to these representations. Limiting features that are used to those derived from higher accuracy, but less sophisticated, analysis—such as POS tagging—may result in a more accurate (albeit potentially less useful) representation. Making these trade-offs is a key concern for system designers. As we will see

| type | features |
|---|---|
| sequence | , vice president of music and programming for |
| sequence bag-of-words | ,, and, for, music, of, president, programming, vice |
| sequence bigrams | , and, and programming, music and, of music, president of, programming for, vice president |
| left of $e1$ token | said |
| right of $e2$ token | . |
| dependency path | $\xleftarrow{\text{nn}}$ president $\xrightarrow{\text{prep for}}$ |
| $e1$ dependent | $\xrightarrow{\text{said}}$ |

Table 3.2: Example a potential feature space for Example 3.

in later sections and throughout the rest of this thesis, defining a representation that balances discriminability and generality is a difficult and still open problem.

We have now discussed representation of relations. Before we can detail systems for RE, we need to discuss data used for evaluation, and, importantly, how relation types are defined.

## 3.3 Data and annotation

Traditional RE defines a relation schema: an explicitly defined set of relations. In this form, RE has been primarily driven by shared tasks. These shared tasks have defined much about the representation of relations and presentation of data for RE. RE was introduced in MUC-7.[3] Three organisation-based relation types were considered: `employee of`, `product of`, and `location of` (Chinchor and Marsh, 1998). For given organisations in text, systems must mark all entities that are in a well-defined relation with their respective organisations. For example, in Example 4 a system would mark (Dennis Gillespie, `employee of`, Navy):

[3]In the 7th Message Understanding Conference, 1997, RE was introduced as *Template Relations*.

(4) The officer said the decision to reassign Kilian to the Pacific headquarters of the *Navy*'s Fighter Wing was made Saturday by the commander of Carrier Air Wing 11, Capt. *Dennis Gillespie.*

The `employee of` relation in MUC-7 is defined with the following key points (directly from the MUC-7 guidelines (Chinchor and Marsh, 1998)), with similar constraints for the other two relations:

- Definition: The person is an employee of the organisation, that is, the person works for the organisation in return for financial compensation.
- If a person dies while in the employment of an organisation, the `employee of` relationship is to be reported.
- Only current employment is to be reported.
- Implication by form is allowed: the writer of an article can be labelled as a relation with the their newspaper, even if that is not explicit in text.
- World knowledge cannot be used for inference, unless that knowledge is present in text. However, if a human reader can infer that a person is a member of professional team from the content, the relation must be reported.
- MUC allows for smaller "organisations" that are actually part of a larger organisation, the Globe Staff as part of Boston Globe. An employee can be related to both organisations, but the smaller-level organisation is optional.
- Club and committee membership, as well as organisation ownership does not constitute employment.

From the one-off MUC RE evaluation, the RE task was carried over to the Automatic Content Extraction (ACE) evaluations, which has RE components from 2002-2008. The ACE relation task was introduced in 2002, and refined in 2004, 2005 and

2008, with ACE 2004 used extensively for benchmarking approaches to RE. Each year's task defined a set of relation types to extract from a set of documents.

Five types are defined in ACE 2002 (LDC, 2002): `Role`, `Part`, `Located`, `Near` and `Social`, each of these types additionally provide subtypes similar to the 2004 subtypes shown in Table 3.3. Additionally, ACE 2002 defines relations as being *explicit* or *implicit*. Explicit relations are relations expressed directly in syntax, and implicit are those which depend on contextual inference. Types of explicit relations include copular predicate modifier (Example 5), and prepositional phrase (Example 6).

(5) **relation:** (Clinton, `Located`, Washington)
**context:** President *Clinton* was in *Washington* today ...

(6) **relation:** (officials, `Located`, California)
**context:** *Officials* in *California* are warning residents ...

Implicit relations are those conveyed as part of the natural understanding of a document. In Example 7, the relation (Israeli policeman, `Located`, a major West Bank road) is implicit. As with MUC, relations that rely on outside world knowledge should not be labelled.

(7) *Israeli policemen* fired live rounds in the air Thursday to disperse hundreds of young Palestinians who blocked *a major West Bank road* to show their support for Saddam Hussein.

The ACE 2004 relations are listed in Table 3.3. We note a similar version of MUC's `employee of` appears as `employment` (as part of `employment / membership / subsidiary`). In ACE, `employment` applies only between PERs and the ORG or GPE by which are employed, explicitly constrained by these named entity types. The

| relation type | subtypes |
|---|---|
| physical | located, near, part-whole |
| personal-social | business, family, other |
| employment/membership/subsidiary | employ-executive, employ-staff, employ-undetermined, member-of-group, partner, subsidiary, other |
| agent-artifact | user-or-owner, inventor-or-manufacturer, other |
| person-org affiliation | ethnic, ideology, other |
| GPE affliation | citizen-or-resident, based-in, other |
| discourse | - |

Table 3.3: ACE 2004 relation types.

employ-* subtypes capture where the person is executive (Example 8) or staff (Example 9), or if this is undetermined (Example 10).

(8) (George Bush, employee-executive, US)
George Bush, the US president, ...

(9) (a senior programmer at Microsoft, employee-staff, Microsoft)
Mr. Smith, *a senior programmer at Microsoft* ...

(10) (Microsoft spokesman, employee-undetermined, Microsoft)
*Microsoft* spokesman, Bob Jones ...

All of the other subtypes are catch-all for relations which are not otherwise covered in a given type. A discourse relation indicates a part-whole or membership relation that isn't between real-world entities, such as (Many of these people, discourse, these people) in Many of these people.

The *explicit* and *implicit* relations are no longer distinct classes in ACE 2004 (LDC, 2004), this concept is replaced by set of relation justification classes. Relations are annotated as *possessive*, *preposition*, *premodifier*, *formulaic* and *verbal* (note that these labels are not used as part of the evaluation metric). ACE 2005 (LDC, 2005) adds

three more classes: *coordination*, *participial* and a broader *other* class. Note that these classes require relations to be extracted at a sentence level only. A *reasonable reader rule* is also defined, relations are only annotated where there is no reasonable interpretation of the sentence under which the relation does not hold. Additionally, relations are annotated with modality *asserted* or *other*, where *other* indicates a hypothetical relation, such as in Example 11. While this modality is annotated, ACE is directly concerned with the evaluation of asserted relations. Tense (past, present, future and unspecified) is also annotated.

(11) **relation:** (Al-Qaeda, `located`, Baghdad)
**context:** We are afraid *Al-Qaeda* terrorists will be in *Baghdad*.

ACE does not allow relations through entities tagged as part of ACE entity annotation: this is of particular importance for annotation and evaluation. For example, because hotel is an entity in ACE, Example 12 is not a valid labelling. However, conference is not a taggable entity in ACE, and so Example 13 is a valid labelling. This is somewhat arbitrary, as conference could be be taggable in a different entity schema.

(12) **wrong relation:** (Smith, `located`, Brazil)
**context:** *Smith* went to a hotel in *Brazil*.

(13) **relation:** (Smith, `located`, Brazil)
**context:** *Smith* went to a conference in *Brazil*.

The raw document format for ACE is essentially the same as for TAC (TAC follows from ACE in this regard), as presented in Chapter 2. ACE 2005, in particular, is a slightly more cross-domain task than TAC, due to the better balance between data sources. As can be seen in Table 3.4, newswire and web documents do not dominate the corpus. However, due to many relations in ACE being very short-range, it is unclear whether this distribution has a large impact on the difficultly of

| | 2002 | 2004 | 2005 | 2009 | 2010 | 2012 | 2013 |
|---|---|---|---|---|---|---|---|
| broadcast conversation | - | - | 60 | 17 | 17 | 17 | - |
| broadcast news | 51 | 220 | 226 | 665 | 665 | 665 | - |
| conversation telephone speech | - | 8 | 39 | 1 | 1 | 1 | - |
| newswire | 29 | 128 | 106 | 1,286,609 | 1,286,609 | 2,286,866 | 1,000,257 |
| web text | - | - | 119 | 1795 | 490,596 | 1,490,595 | 999,999 |
| discussion forums | - | - | 49 | - | - | - | 99,063 |
| newspaper | 17 | - | - | - | - | - | - |
| Chinese Treebank translation | - | 37 | - | - | - | - | - |
| Arabic Treebank translation | - | 58 | - | - | - | - | - |

Table 3.4: ACE 2002-2005 and TAC 2009-2014 corpus statistics.

the task. Overall, the number of documents is far smaller, reducing the number of possible contexts for extraction and making for a more targeted task.

ACE uses an entity-based markup for relations, an example of a relation from the ACE 2005 annotation is in Figure 3.3. This is a particularly long-range relation for ACE, as can be seen by the lexical condition `Other`. The relevant text is contained in `extent`, and this sample contains two mentions (`relation_mention_argument`) of entities that are not named.

In moving to TAC KBP, RE loses these some of these task-specific restrictions (particularly those which result in a focus on short-range relations), as TAC is intended to provide a more "real-world" setting. Relation specifications become more about the understanding of a relation rather than syntactic constraints. While TAC's `per:employee of` has a large number of caveats, these concern the definition of employee, rather than the form and location of fills in text.

```
<relation ID="AFP_ENG_20030319.0879-R31"
TYPE="PART-WHOLE" SUBTYPE="Subsidiary" TENSE="Unspecified" MODALITY="Asserted">
  <relation_argument REFID="AFP_ENG_20030319.0879-E17" ROLE="Arg-1"/>
  <relation_argument REFID="AFP_ENG_20030319.0879-E13" ROLE="Arg-2"/>
  <relation_mention ID="AFP_ENG_20030319.0879-R31-1" LEXICALCONDITION="Other">
    <extent><charseq START="2024" END="2264">Faced with debt of 35 billion euros
    (37 billion dollars) in June last year -- including 19 billion from the media
    and telecoms division alone -- the sprawling conglomerate has set a target
    of reducing its debt by 16 billion euros by end-2004</charseq></extent>
    <relation_mention_argument REFID="AFP_ENG_20030319.0879-E17-42" ROLE="Arg-1">
      <extent><charseq START="2128" END="2158">
          the media and telecoms division</charseq></extent>
    </relation_mention_argument>
    <relation_mention_argument REFID="AFP_ENG_20030319.0879-E13-43" ROLE="Arg-2">
      <extent><charseq START="2169" END="2194">
          the sprawling conglomerate</charseq></extent>
    </relation_mention_argument>
  </relation_mention>
</relation>
```

Figure 3.3: ACE relation annotation example. Arguments are underlined.

## 3.4 Machine learning approaches to RE

RE techniques underlie SF systems, and so we now cover these techniques before we turn our attention to SF. Techniques for traditional RE can be broadly divided into one of five categories: *rule-based* systems which explicitly use patterns for extraction; *similarity-based semi-supervised* approaches which directly leverage similarity with training instances; *supervised classification* (feature-based and kernel-based); *distant supervision-based classification*[4]; and approaches which incorporate *open schema*.

### 3.4.1 Rule-based

Rule-based approaches directly use patterns to extract relation instances. These will typically be hand-coded, or curated from examples found in a sample of text. Just as we can represent an example `employee of` relation as [ PER $\xleftarrow{\text{nn}}$ president $\xrightarrow{\text{prep for}}$ ORG ], we can use that pattern to extract instances of `employee of`. Early approaches to RE use this approach (Aone et al., 1998; Yangarber and Grishman, 1998); it continues to be a part of some successful TAC systems (Sun et al., 2011b).

Typically, these patterns are high precision but have very low coverage. The distribution tends to be very focused on straightforward expressions of relations. This is similar to ACE's focus on syntactic justification: short-range patterns are easier to identify in text.

Purely rule-based systems are typically outperformed by learning-based approaches which can better generalise over training instances than manual rules. Rule-based approaches remain popular in industrial applications (Chiticariu et al., 2013), and have seen use in TAC (as we will see in Section 3.5), as they can be useful for quickly incorporating domain knowledge and fixing high-impact errors. Much

[4]Distant supervision is often discussed as a semi-supervised method of acquiring training data. It is also not a distinct learning approach. However, there is a distinct body of classifier-based work that we can categorise in this way.

of the work in pattern-based extraction has been expanded by semi-supervised approaches, which can expand a small set of initial rules.

### 3.4.2 Semi-supervised

Semi-supervised learning approaches are supervised techniques that make use of unlabelled data for training. Many semi-supervised approaches, known as self-training, use a small seed set of labelled data to train an initial model, and then use this model on unlabelled data to find new instances to label. These new examples are used to retrain the model, and this process is repeated. This is potentially most useful when available training data is limited, as is the case in RE. Much of this space has been explored outside of the shared tasks.

Semi-supervised approaches can be considered as extension of pattern-based approaches: initially a small set of patterns are defined or extracted. This initial step is effectively a pattern-based system, which is then used to find more patterns. Agichtein and Gravano (2000) describe *Snowball*, a basic bootstrapping approach for RE, based on the earlier work of Brin (1998) extracting `book author` pairs from the web. Seed relation tuples (Example 14) are located in text (Example 15), and the text patterns between them are extracted (Example 16).

(14) (Microsoft, `location of headquarters`, Redmond)

(15) *Microsoft*'s headquarters in *Redmond*

(16) ORG's headquarters in LOC

These patterns are used to locate further mention pairs based on the text pattern itself and the types of the pair (Example 17), extracting new tuples (Example 18).

(17) *Exxon*'s headquarters in *Irving*

(18) (Exxon, `location of headquarters`, Irving)

Confidence scores are calculated for patterns and tuples, where confidence for a pattern is its precision over tuples, and confidence for a tuple is derived from the confidence of paths. This process is then repeated using high confidence tuples. Low confidence tuples are discarded. They find that a small number of iterations, in some configurations only two iterations, are required to converge.

Agichtein and Gravano find that many errors from Snowball are due to the extraction of non-existent organisations, and using a high confidence threshold $\tau = 0.8$ prunes most precision errors. From an initial set of 5 seeds in the North American News Text Corpus (NANC) (LDC, 1995), they find a random sample of 100 tuples has 7 incorrect as opposed to 48 incorrect for $\tau = 0$. However, recall falls quickly as precision is increased by raising the confidence threshold, with the recall dropping from 88 for $\tau = 0$ to 18 for $\tau = 0.8$. Nevertheless, they find their approach can be used to generate high quality tuples.

Bootstrapping approaches have been successful for a range of tasks which require classes to be acquired for instances, including word sense disambiguation (Yarowsky, 1995); document classification (Blum and Mitchell, 1998); and named entity classification (Collins and Singer, 1999). One of the biggest issue with bootstrapping is the cascading precision loss between iterations. This is termed *semantic drift* and occurs when errors accumulate more errors (Curran et al., 2007). This problem is addressed by adding constraints to the bootstrapping process, such as weighted mutual exclusion (McIntosh and Curran, 2008, 2009), which aim to minimise unchecked growth.

Carlson et al. (2010) base their system, Never-Ending Language Learning (NELL), on the approach of Agichtein and Gravano (2000), with the addition of a range of constraints. They couple together a range of extractors for sentences, semi-structured elements such as lists, and structured HTML elements of documents. As well as relation tuples, they also jointly extract instances of NE types using the same bootstrapping process. Targeting high-precision extraction, they add several

constraints to extractors: mutual exclusion for types that are mutually exclusive, complex argument type checking, and separate patterns to extract from unstructured and semi-structured text features. These manually-defined constraints can be reasonably complex, such as that a person cannot have a `date of death` before a company they are `employed by` is `founded`.

They add these constraints as filters in the bootstrapping process, providing a small set of seeds derived from patterns for hyponymy extraction (Hearst, 1992). 55 relation and 123 NE types are initially defined. Bootstrapping is done over lightweight linguistic features, using only word sequence and POS tag patterns. NELL has been continuously run since January 2010, with some human input to validate/correct categories and relations. At high confidence, precision is very high but recall appears to be low, accumulating 2 million high confidence beliefs over 500 million documents in ClueWeb09. This approach has not been evaluated on ACE, and it is difficult to determine how well this linguistically lightweight approach would perform on a small scale task that requires high yield extraction.

NELL has been substantially extended beyond the base approach, focusing on acquiring relation tuples by inference over the graph of relation tuples, and the discovery of new relation types outside those defined by initial seeds. Lao et al. (2011) explore inferring relations from existing relations and categories using random walks over the relation graph. Mohamed et al. (2011) propose an approach to automatically discover relations by clustering text contexts for pairs of categories, and then using a classifier trained on existing relations to determine which clusters are valid relations.

Building upon bootstrapping, label propagation (Zhu and Ghahramani, 2002) in a RE setting allows for the learning process to take the distribution of data into account. In label propagation, labelled seed nodes in a graph are iteratively propagated to unlabelled nodes. Edges in the graph are the similarity of nodes. Chen et al. (2006) apply this technique to RE. Labelled and unlabelled mention pairs are

represented at nodes on a connected graph, then propagate label information to nearby nodes through weighted edges on the graph. Their system achieves an F1 of 54.6% on ACE 2002 subtypes, slightly worse than the state-of-the-art classifier approach at the time (these approaches will be discussed shortly). They hypothesise that this may be due to their limited feature set (lexical and chunk features) or less sophisticated similarity measure—they make use of cosine similarity and Jensen-Shannon divergence (Lin, 1991). Wang et al. (2011) expand on this with their PRAVDA system, making use of the Modified Absorption algorithm (Talukdar et al., 2008) to extract relations in the soccer and celebrity domains with high precision. Modified Adsorption, based on the Absorption (Baluja et al., 2008) label propagation algorithm, takes into account the impact of very frequent uninformative features and incorrect seed labels by including a measure of entropy over random walks in the graph. We will describe and use this algorithm in Chapter 5.

Other work in semi-supervised RE focuses on aspects of these approaches. Gabbard et al. (2011) consider the use of coreference resolution in the bootstrapping process: evaluating on ACE 2007, they find that recall is improved with the use of coreference, seeing a significant increase in recall on all relations, but results on precision are mixed. They only test on a small set of ten relations, but double the F-score for half of these. All are improved by the use of coreference in the bootstrapping process.

### 3.4.3 Supervised classification

Supervised classification approaches treat RE as a mention pair classification task. This is framed in one of two ways (Sun et al., 2011a). Given $n$ relation types, typically a $(n + 1)$-way classifier is trained, adding `no relation` as a relation type. Alternatively, two classifiers are trained: a binary classifier to determine if any relation in a schema exists, and then a $n$-way classifier to identify the actual relation.

| corpus | system | types | | | subtypes | | |
|---|---|---|---|---|---|---|---|
| | | P | R | F1 | P | R | F1 |
| ACE 2002 | Kambhatla (2004) | - | - | - | 63.5 | 45.2 | 52.8 |
| | GuoDong et al. (2005) | 77.2 | 60.7 | 68.0 | 63.1 | 49.5 | 55.5 |
| ACE 2004 | Jiang and Zhai (2007) | 72.4 | 70.2 | 71.3 | - | - | - |
| | Sun et al. (2011a) | - | - | 71.5 | - | - | - |

Table 3.5: Results on ACE for feature-based classifiers.

Supervised approaches broadly fall into two major categories: feature-based and kernel-based. Feature-based approaches learn individual weights for features that represent an instance, whereas kernel-based methods use a kernel function to measure the similarity of instances.

**Feature-based** No MUC systems made use of classifiers.[5] Kambhatla (2004) were the first to apply this approach to ACE, using a range of lexical, syntactic and entity type features with a maximum entropy classifier. Results for this approach, and other feature-based classifier approaches in this section are shown in Table 3.5. A larger set of features was used to build an SVM model in GuoDong et al. (2005), increasing performance over other systems. Features added in this system were primarily finer-grained features (e.g. instead of bag of words as the only lexical feature, they include a range of lexical features such as single words before and after mentions), and incorporation of gazetteers as features.

Jiang and Zhai (2007) expand on this by carrying out a study of the feature space for RE. They classify a range of features: entity attributes, bag of words, bigrams, constituency parse and dependency parse features . They evaluate on ACE 2004 and conclude that using a set of basic features from each feature subspace can achieve near state-of-the-art performance, and that addition of overly complex features may hurt performance. Their best configuration uses a maximum entropy classifier with

[5]Only one MUC-7 system used a learning-based approach: Miller et al. (1998) treated RE as a parsing problem augmented constituency parses with relation attributes, estimating a PCFG that learned these attributes. This allowed for extraction of relations directly from a parsed sentence.

sequence and constituency parse bigrams and trigrams, but dependency parse feature results are similar, suggesting redundancy between the feature spaces. These feature-based classification approaches perform at slightly lower levels compared to kernel methods on ACE, and these methods will be discussed next.

Sun et al. (2011a) note performance of supervised RE is hampered by the sparsity of lexical features in the training data. For example, person roles that exist in relations in the evaluation data may not appear in the training data, and other features are not informative. They apply Brown word clustering (Brown et al., 1992) to add cluster features to a maximum entropy classifier. Adding these cluster features gives them a small but statistically significant increase in F1 from 70.4% to 71.5% on the high-level types.

Due to this substantial sparsity of features in training, coupled with the relative lack of available training data, work in RE that uses a learned classifier has mostly been carried over to distantly supervised approaches, and will continue to be discussed in Section 3.4.4.

**Kernel-based** Kernel methods define a kernel function that provides a measure of similarity between two instances. This allows features that have a complex similarity to be useful, as opposed to just computing cosine similarity over a space of binary features. In Figure 3.4 from Bunescu and Mooney (2005), dependency paths are expressed as the features generated by each token on the path: the word, POS tag, generic POS tag and entity tag. The kernel function is define over two paths as the product of the number of elements that match in each position in the path. In Figure 3.4, the first position has three matching elements. The kernel score in this case, for these paths, is $3 \times 1 \times 1 \times 1 \times 2 \times 1 \times 3 = 18$. Other kernel functions define different measures of similarity. For example, a similarity measure between word sequences may be implemented as the edit distance between sequences.

$$\begin{bmatrix} \text{his} \\ \text{PRP} \\ \text{PER} \end{bmatrix} \times \begin{bmatrix} \rightarrow \end{bmatrix} \times \begin{bmatrix} \text{actions} \\ \text{NNS} \\ \text{NOUN} \end{bmatrix} \times \begin{bmatrix} \leftarrow \end{bmatrix} \times \begin{bmatrix} \text{in} \\ \text{IN} \end{bmatrix} \times \begin{bmatrix} \leftarrow \end{bmatrix} \times \begin{bmatrix} \text{Brcko} \\ \text{NNP} \\ \text{NOUN} \\ \text{LOC} \end{bmatrix}$$

$$\begin{bmatrix} \text{his} \\ \text{PRP} \\ \text{PER} \end{bmatrix} \times \begin{bmatrix} \rightarrow \end{bmatrix} \times \begin{bmatrix} \text{arrival} \\ \text{NN} \\ \text{NOUN} \end{bmatrix} \times \begin{bmatrix} \leftarrow \end{bmatrix} \times \begin{bmatrix} \text{in} \\ \text{IN} \end{bmatrix} \times \begin{bmatrix} \leftarrow \end{bmatrix} \times \begin{bmatrix} \text{Beijing} \\ \text{NNP} \\ \text{NOUN} \\ \text{LOC} \end{bmatrix}$$

Figure 3.4: Example dependency paths used in Bunescu and Mooney (2005).

| | | types | | | subtypes | | |
|---|---|---|---|---|---|---|---|
| corpus | system | P | R | F1 | P | R | F1 |
| ACE 2002 | Culotta and Sorensen (2004) | 67.1 | 35.0 | 45.8 | - | - | - |
| | Bunescu and Mooney (2005) | 65.5 | 43.8 | 52.5 | - | - | - |
| | Zhang et al. (2006) | 77.3 | 65.6 | 70.9 | 64.9 | 65.6 | 70.9 |
| | Zhou et al. (2007) | 80.3 | 68.4 | 74.1 | 65.2 | 54.9 | 59.6 |
| ACE 2004 | Zhao and Grishman (2005) | 69.2 | 70.5 | 70.4 | - | - | - |
| | Zhang et al. (2006) | 76.1 | 68.4 | 72.1 | 68.6 | 59.3 | 63.6 |
| | Zhou et al. (2007) | 82.2 | 70.2 | 75.8 | 70.3 | 62.2 | 66.0 |
| | Nguyen et al. (2009) | 76.6 | 67.0 | 71.5 | - | - | - |

Table 3.6: Results on ACE for kernel-based classifiers.

While some of these kernels are sequence-based (Mooney and Bunescu, 2006), the majority are tree kernels, which measure the similarity between parse trees.

Zelenko et al. (2003) introduce the use of kernel methods for RE. The define a kernel function for shallow parse subtrees that contain mention pairs, incorporating POS tags, entity types, phrase heads and the relation between the pair of entities. The kernel function uses a set of rules to score similarity. They evaluate on `person-affiliation` and `organization-location` on a corpus of 200 news documents. They find a kernel-based SVM outperforms a feature-based SVM and perceptron on those two types. This has possibly limited generality to other types.

This approach is extended with dependency trees by Culotta and Sorensen (2004), who use a similar tree matching function to Zelenko et al., making additional use of dependency features. They demonstrate that the subtree kernels have better performance than a bag-of-words kernel, and the combination of both performs best. Their best result uses a cascaded setup, with a relation detection classifier (i.e. an any-vs-none classifier) followed by a full relation classifier. Results for this system, along with other kernel-based classifier results, are in Table 3.6.

Bunescu and Mooney (2005) propose that the shortest dependency path be used for relation extraction, hypothesising that the useful context is concentrated in this path. Their kernel is defined for short paths of the same length, comparing word, POS tag and entity types for tokens, and unlabelled arc direction for arcs. This approach substantially improves results on ACE over Culotta and Sorensen, again using a cascaded setup.

ACE 2003 and 2004 added further levels of provided annotation, including entity headwords and entity subtypes, and systems incorporated this data. Zhao and Grishman (2005) built upon earlier approaches by proposing a number of kernels over different feature spaces, combining these together in composite kernels. They find that each level of syntactic processing is informative for the task. Zhang et al. (2006) propose a composite kernel which combines a convolution parse tree kernel (Collins and Duffy, 2002) with an entity feature kernel. Their combination kernel has better performance than a single syntactic kernel. Zhou et al. (2007) expand on this by adding arcs outside the shortest path in constituency parse to include more context about certain relations. For example, in the text John and Mary got married, the shortest path between John and Mary does not contain married. This can be added as additional context. They extend convolution tree kernel to use this additional context, and achieve state-of-the-art performance on ACE. Nguyen et al. (2009) explore the use of convolution tree kernels on dependency trees, and combine this approach with constituency parse and sequence kernels. Plank and

Moschitti (2013) propose exploiting lexical semantic similarity features for kernels, showing that a combination of these features is promising for adapting a system to different domains.

**Deep learning** Deep learning approaches have been applied to RE, although uptake for SF has been slower than for other NLP tasks. Zeng et al. (2014) and dos Santos et al. (2015) apply convolutional neural networks (CNNs) to semantic relation classification (identifying relations between words, instead of between NEs as in RE), and find that CNNs perform better than other models.

Malon et al. (2012) implement a CNN for SF, and their CNN outperforms their classifier-based approached, but there overall system performance is low. Adel et al. (2016) experiment with a range of models for SF, including a CNN. Their CNN-based approach outperforms SVM and pattern-based approaches, and gets better results than other supervised approaches on a new dataset derived from SF data. Vu et al. (2016) continue work on semantic relation classification, finding that a combination of a CNN and a recurrent neural network outperforms both approaches individually.

### 3.4.4 Distant supervision

The lack of available training data for RE, in conjunction with the domain dependence of this data and the cost of creating a large set of manual annotations has prompted substantial work in the area of *distant supervision*. Originally proposed in biomedical NLP for learning from *weakly labelled data* (Craven and Kumlien, 1999; Morgan et al., 2004), distant supervision takes relation tuples from a KB, locates mentions of the pair of entities in text (typically in the same sentence), and uses these examples as training data. In this chapter, we categorise this as distinct from semi-supervised learning, which may use training tuples from a KB, but uses an iterative process to expand the example set.

Craven and Kumlien extract instances of protein location in subcellular structures: for example, the (collagen, `subcellular localization`, plasma-membranes) tuple from a KB can be aligned to Example 19:

(19) Immunoprecipitation of biotinylated type XIII collagen from surface-labeled HT-1080 cells, subcellular fractionation, and immunofluorescence staining were used to demonstrate that type XIII *collagen* molecules *are indeed located* in the *plasma membranes* of these cells.

In this way, large KBs can be used to create a substantial amount of training data.

Following this work, Snow et al. (2004) use WordNet hypernyms to extract hypernym relations between entities in newswire, training a range of classifiers on the newswire. Bunescu and Mooney (2007) apply a similar approach to the task of corporate acquisitions, using a very small set of 6 KB pairs to train an SVM classifier.

Wu and Weld (2007) extract relations from a Wikipedia page using distant supervision from the page's infobox to train CRFs, in order to fill other similar infoboxes, effectively performing in-domain distant supervision. Hoffmann et al. (2010) find that this technique scales to 5,025 relation types in Wikipedia, but some additional techniques including dynamic lexicon learning are required to deal with sparsity despite a large amount of available training data.

Mintz et al. (2009) apply distant supervision to Freebase, seeking to avoid the domain dependence of small hand-labelled corpora. They use 900,000 tuples from Freebase to train a multi-class logistic classifier with high-precision pattern features, using Wikipedia as the evaluation corpus. Freebase is a publicly accessible and previously publicly editable[6] KB of semantic data. Sources of data in Freebase include Wikipedia infoboxes and tabular data, the Notable Names Database (biographical data) and the U.S. Security and Exchange Commission (company data). Mintz et

[6] At time of writing, Freebase is going through a shutdown process with data being transferred to Wikidata. It became read-only on 31/03/2015.

al. extract a dataset of 1.8 million tuples for 102 relation types between 940,000 entities, a much larger source of training and evaluation tuples than previously used for RE. For several thousand extractions manually evaluated, they achieve a precision of 68%. As a frequently followed approach in TAC SF systems, we use this large-scale domain-independent application as the prototypical definition for *distant supervision*.

Mintz et al. primarily uses lexical, syntactic and entity features. Feature engineering is a minor component of distant supervision research, and most systems use the Mintz et al. set of features. These features are standard in the literature for classifier-based systems, and are composed of the types of both entities and the sequence of words or dependency path between them, as well as the words and modifiers to the left and right of the entities.

The alignment of entity pairs into sentences relies on the *distant supervision assumption*: the assumption that if two entities participate in a KB relation, then all sentences that mention those two entities express that relation. For example, tuple (Virginia, `location contains`, Richmond) aligns to Example 20, and features can be extracted for `location contains`.

(20) *Richmond*, the capital of *Virginia*.

However, this assumption is flawed: most given pair of entities do not exist in a single, unambiguous relationship, e.g. (Obama, U.S.) has several relations that apply, such as `employment`, `place of birth` and `residence`.

Additionally, even if every pair had an unambiguous relationship, every occurrence of the pair of entities would have to express that relation for the assumption to be correct. This is not the case, e.g. the labelling (Obama, `per:employee of`, U.S.) should not be applied to Example 21.

(21) *Obama* was born in the *U.S.*

However, this assumption is still useful for creating training data. Mintz et al. use these alignments as training instances for a multiclass logistic regression classifier, combining features for identical tuples. Mintz et al. align the Freebase tuples to Wikipedia sentences. Negative data for training is constructed by randomly selecting 1% of entity pairs that do not appear in any relation. They note that 98.7% of entity pairs in Wikipedia are not in any relation in Freebase.

### Sources of error

While distant supervision provides a large amount of training data, there are two key sources of error that it introduces:

1) Invalid tuple alignment. As discussed, the distant supervision assumption is flawed: a mention of a pair in text may or may not entail a particular relation.

2) KB noise. Noise can occur in the form of factual errors in the KB, but also due to ambiguous entities or a lack of specificity. For example, a `spouse` relation between every Bill and Hillary in text is likely to create more incorrect extractions than for Bill Clinton and Hillary Clinton.

Work on improving distant supervision has focused on these two areas.

### The distant supervision assumption

One of the areas of RE research since Mintz et al. (2009) has been in both relaxing and better modelling the distant supervision assumption. Riedel et al. (2010) map Freebase to the New York Times corpus (NYTC) for `nationality`, `place of birth` and `contains`, and find the assumption is violated 31% of the time. They propose an *expressed-at-least-once* assumption that at least one sentence that mentions a pair of entities might express their KB relation. They frame the problem as a form of multi-instance learning, substantially improving extraction performance. This assumes that pairs do not exist in multiple relations: as in the earlier (Barack Obama,

United States) example, this assumption is often violated. Hoffmann et al. (2011) note that 18.3% of facts in Freebase that match a sentence in the NYTC voliate this, and propose a graphical model of multi-instance learning which handles overlapping relations (MULTIR). Surdeanu et al. (2012) further this by proposing a multi-instance multi-label graphical model (MIML-RE), which jointly trains an entity-pair level classifier of relations with a classifier that assigns relations to mention instances and can capture dependencies between labels. The multi-instance aspect of this treats all sentence-level relation instances as a single 'bag' of instances to be classified (as opposed to being classified individually); the multi-label aspect allows for multiple labels to be assigned to these bags.

Nguyen and Moschitti (2011a) extend distant supervision by adding semantic relations for other external sources, such as YAGO (Suchanek et al., 2007), a semantic KB, and increasing the size of training data. They also find that both dependency and constituency parse features are useful when used together in a SVM. They extend this work in Nguyen and Moschitti (2011b), evaluating on ACE 2004 and experimenting with NER as part of an end-to-end system. Angeli et al. (2014) release a large amount of additional data specifically for TAC SF, with the intention of improving the coverage of training data for the task. To these end they use an active-learning style approach over distantly-supervised training instances from Freebase to target informative instances for annotating. They use Mechanical Turk workers to annotate a total of 33,748 new training examples. Making use of this data gives them a 3.9% gain in F1 for their SF system. This large source of data is particularly useful for our work, and we will make use of this data later.

**Reducing noise**

Roth et al. (2013a) categorise different noise reduction methods as using one of three basic principles. The first are the approaches to distant supervision modelling discussed above, based on the *expressed-at-least-once* assumption, which seek to

model noise as part of the learning process. The second is the approach used in Alfonseca et al. (2012), where a hierarchical topic model is used to identify useful patterns. The third is the approach of Takamatsu et al. (2012), which leverages a generative model to identify which patterns are do not express relations or express incorrect relations. These patterns are then used to filter out tuples.

### 3.4.5 Unsupervised RE

As we have explored, traditional RE approaches require a relation schema to be pre-defined. Other approaches seek to identify relations beyond a pre-defined schema, and without training data.

The general technique for these approaches is to identify potential relations and then cluster these into concrete semantic relation types. Early work in the question answering domain considered clustering by distributional similarity of entities and relation phrases (Lin and Pantel, 2001). Shinyama and Sekine (2006) cluster articles that reference the same event, and then cluster relations across entities within and across these event clusters. For the purpose of TAC SF, entirely unsupervised approaches are not directly useful, as mapping back to a TAC slot is required. Typically, unsupervised methods are only used for components of extraction, such as clustering features. Some systems map unsupervised clusters to labels in a rule-based fashion, with limited success.

### 3.4.6 Open information extraction

Open Information Extraction (OpenIE), introduced with TextRunner (Banko et al., 2007), uses lightweight linguistic techniques to scale unsupervised RE to a web-scale corpus, with the motivation that sparsely occurring relations may be identifiable given sufficient redundancy over enough documents. TextRunner extracts 1 million distinct relation types. TextRunner trains an extractor using parse tree features but uses only abstracted chunking features for fast extraction. ReVerb (Fader et al.,

2011) uses a small set of verb-based patterns and lexical constraints to extract relation phrases, and then identifies the relevant relation arguments. A classifier is trained using textual features to score relation phrase confidence. Ollie (Mausam et al., 2012) uses a single bootstrapping step on relations extracted by ReVerb to generate general dependency patterns, which are then used to extract further relations. These systems produce a low yield of high-confidence patterns.

These key systems do not canonicalise relation phrases or arguments to relations or entities, and have the same limitations as unsupervised approaches for the purpose of TAC. A substantial amount of work on OpenIE has tried to deal with this problem. Yates and Etzioni (2009) cluster relation arguments into synonym groups. Lin et al. (2012a) and Lin et al. (2012b) both considering linking entities to a knowledge base (Freebase) and typing these entities. Min et al. (2012b) propose an algorithm for clustering relation phrases based on their entity types as linked to Freebase, allowing relations to belong to multiple relation clusters.

In addition to canonicalisation, inference of constraints for relations is a key component of unsupervised RE. Lin et al. (2010) consider learning relation properties based on the distribution of relation arguments. As OpenIE systems aggregate across relations that appear many times, they are high precision, but remain low recall for individual documents (Banko and Etzioni, 2008).

*Universal schema* (Riedel et al., 2013) approaches seek to bridge the gap between schema-based and schema-less approaches. Universal schema takes the union of multiple schema (using surface forms in the case of schema-less approaches), and treats RE as a matrix factorisation problem, learning latent feature vectors for relation tuples, relation types and entity pairs, and weights that capture direct correlations between relations. In this way universal schema approaches focus on predicting relations rather than modelling semantic equivalence.

## 3.5 Approaches to TAC SF

We have now covered RE approaches to SF, and can move on to discussing SF approaches themselves. SF systems typically consist of several pipelined stages (Ji et al., 2011), with each stage potentially being comprised of several ensembled components. The basic pipeline consists of four stages (Ji and Grishman, 2011a): *document retrieval*, *candidate generation*, *answer extraction*, and *answer post-processing*. We note that these stages do not explicitly need to be pipelined: for example, a system could generate candidates across a corpus without first isolating relevant documents; or may jointly extract and perform inference over answers. However, this pipeline applies to most systems and we will use this as a framework for analysis. Answer post-processing primarily refers to systems merging and ranking answers for output. However, post-processing can be arbitrarily complex, and some systems apply sophisticated inference over extracted answers. As there are few differences in approaches to document retrieval, we discuss those first before discussing the core candidate generation and answer extraction components.

### 3.5.1 Document retrieval

Most documents in a corpus will not contain information about a query entity, and so systems typically first reduce the search space for a particular query by retrieving only relevant documents from the corpus. This is almost always implemented in two components: the first is query expansion, which take a query entity and converts it into a search query. The second component is search which takes that query and returns relevant documents. This separate step usually occurs due to the size of the corpus: using an information retrieval approach is much more efficient, as opposed to applying named entity linking or extracting relations for every entity pair in the corpus.

### Query expansion

Our analysis in Pink et al. (2014), discussed in Chapter 4, shows that for TAC 2011 most documents could be retrieved without query expansion. However, for complete coverage of relevant documents systems must ideally find all different mentions of an entity across a corpus. Systems expand queries in several ways.

**No expansion** Some approaches simply use the query name directly to search for documents. However, no top performing systems use this straightforward approach. Systems may pass the full name to the search engine, or allow a partial match (Varma et al., 2010).

**KB aliases** If a KB node id is provided with a query, extract aliases directly from the KB node, e.g. Tahs is listed as a nickname in the infobox for NEW SOUTH WALES WARATAHS (Castelli et al., 2010). Kisiel et al. (2013) lookup entity aliases in FACC1,[7] an annotation of Freebase tuples into the web-scale ClueWeb corpus.

**KB redirects** If a KB node id is provided with a query, use Wikipedia redirects to the KB node to extract aliases (Chen et al., 2010; Jian et al., 2011; Min et al., 2012a).

**Rule-based aliases** Apply a set of transformation rules to the query name (Chen et al., 2010; Xu et al., 2013). These are typically: addition or removal of organisation suffixes such as Corp.; initialisation of person names; removing initials and middle names from person names; and use of acronyms of organisations. Ageno et al. (2013) use a grammar of person names to extract the structure of a name, primarily to maintain the family name so that first name variants can be generated, e.g. Robert to Bob, Bobby. Roth et al. (2013b) also calculate the pointwise mutual information (PMI) between an alias and a query name, an alias must have a high PMI above a threshold before it is considered to be a valid alias. Angeli et al. (2013) make use

[7] Freebase Annotations of the ClueWeb Corpora, http://lemurproject.org/clueweb12/FACC1/

of a backoff IR, using progressively less precise queries until a certain number of documents (50 in their case) are retrieved.

**Coreferential mentions** Mentions coreferential with the query entity can be used as aliases (Angeli et al., 2013).

`alternate names` **fills** If a query has fills for the appropriate `alternate names` slot, identified in the answer extraction stage, these names can be used as aliases to retrieve more documents (Ageno et al., 2013).

**KB links** Given New South Wales Waratahs in a document, NEL would link that mention to the TAC KB node for NEW SOUTH WALES WARATAHS. Other mentions of this entity in the corpus, such as NSW Waratahs, Waratahs and 'Tahs would also link to this node, and their corresponding documents could be retrieved. If such a node does not exist in the KB, then another similar approach to clustering these entities, such as cross-document coreference, resolution is required. However, it is not typically feasible to apply these approaches to a large corpus.

### 3.5.2 TAC training data

SF approaches typically make use of some source of training data for answer extraction, whether that data is used in a supervised or semi-supervised fashion, or as a source of information for hand-written rule-based systems. Several sources of training data are available for SF. Official SF training data was released for the 2009 and 2010 tasks. This data was limited in size, and counts of annotations of this data are given in Table 3.7. Additionally, each set of evaluation assessments is released each year, these can be used as additional training or evaluation data. While this data is reasonable to use for training a traditional RE-based system, there are a few issues to consider.

| slot | count | slot | count |
|---|---|---|---|
| org:top members/employees | 183 | per:schools attended | 17 |
| org:members | 92 | org:website | 15 |
| per:title | 88 | per:siblings | 13 |
| org:alternate names | 81 | per:origin | 13 |
| org:subsidiaries | 68 | per:spouse | 12 |
| per:employee of | 62 | per:other family | 11 |
| per:cities of residence | 42 | per:alternate names | 11 |
| per:age | 37 | org:member of | 11 |
| per:member of | 34 | per:children | 10 |
| org:country of headquarters | 34 | per:date of birth | 9 |
| org:city of headquarters | 33 | per:country of birth | 7 |
| org:stateorprovince of headquarters | 27 | per:city of birth | 7 |
| per:stateorprovinces of residence | 26 | per:stateorprovince of birth | 5 |
| per:parents | 25 | per:charges | 5 |
| org:parents | 24 | per:religion | 4 |
| org:founded by | 23 | org:dissolved | 3 |
| org:founded | 22 | per:date of death | 2 |
| org:number of employees/members | 21 | per:country of death | 1 |
| org:shareholders | 19 | per:city of death | 1 |
| per:countries of residence | 18 | per:cause of death | 1 |
| org:political/religious affiliation | 18 | | |

Table 3.7: Counts of annotations in the 2009 and 2010 TAC training data.

**Training data is limited** Even with several years of assessments available, training data is limited, particularly for infrequently filled slots.

**Query-driven incompleteness** As TAC is query-driven, sentences are not annotated completely, they are only annotated relative to a given set of entities, and sentences are not fully annotated with slot fills for every entity. Negative data is also dependent on the errors that other systems have made.

**Reliance on automatic pipeline** System-generated output is reliant on an automatic pipeline such as NER, hence assessment data will be biased towards output from pipelines commonly used by SF systems. For example, as many systems use

CoreNLP NER (Finkel et al., 2005), many fills will have NER bounds as determined by CoreNLP. If a system uses a pipeline which results in different NER bounds, this may result in invalid or usable training instances.

In addition, negative instances are made up of fills that other automatic systems output as correct, which may influence the training process as compared to a more general selection of negative instances.

**Changes in schema** Changes in schema between years potentially invalidate some training data. However, the effects of this are minor: the location granularity change from 2009 to 2010 means that either the coarse 2009 training data can be ignored, or filtered and used as training data for all appropriate slots (e.g. `org:headquarters` filtered by a gazetteer of countries used for `org:county of headquarters`). The 2013 change to `per:title` is more subtle, and it is not clear what the effect of using earlier training data is, except that systems typically ignore redundant responses when training, and these would have a similar effect of training as the incompleteness issue.

**Relation scope** SF is primarily a document-level task: there is no explicit requirement that entities or entity mentions occur in the same sentence. However, with few exceptions (Swampillai and Stevenson, 2011) RE systems require entities to be mentioned in the same sentence. Furthermore, they require the evidence for a relation to be also contained in the same sentence, which may or may not be the case. When coreference resolution is required to resolve a particular annotated fill, differences in automatic coreference can result in a correct document-level annotation being incorrect at the sentence level. Alternatively, coreference resolution may be required to locate the evidence for a fill in the document, and this requirement is not part of most assessment data. 2014 and later evaluations marked justification spans, so these can now potentially be used to better align training data. However,

this is still not a foolproof process as the interaction between justification spans is inferred by a human assessor and not part of system output.

**Annotation errors** A very small number of annotations are arguably errors, such as in Example 22 where Alvin Hilaire does not appear to be a top employee of the International Monetary Fund:

(22) **wrong annotation:** (Alvin Hilaire, `org:top members/employees`, International Monetary Fund)
**context:** "The situation remains complicated," said *Alvin Hilaire*, the *International Monetary Fund* representative in Guinea.

Most of this annotations were not strictly errors at the time, but are invalid annotations for later years due to schema clarification, the senior official role in Example 23 is too generic a justification for top employee:

(23) **invalid annotation:** (John Lipskey, `org:top members/employees`, International Monetary Fund)
**context:** The reform program "won't be sufficient by itself to steer the economy on a viable financial path," said *John Lipskey*, a senior official with the International Monetary Fund.

**Scope of issues in training data**

To measure the scale of these issues in a typical SF pipeline, we perform some analysis of the 2009 and 2010 slot filling training annotations. For this analysis, we selected only sentences which have both the query and fill NEs and have a dependency path between them that is not unique in the corpus (we will detail this filtering further in Section 4.7). Combined, these two sources provide 1137 annotations. Of these, 903 are *correct* (i.e. true positive) fills. We then take the

corresponding documents and process them using CoreNLP NER, and extract sentences that contain both the query name and fill name explicitly in the same sentence. We do not use coreference resolution to resolve mentions outside the same sentence. As we are using standard 4-class NER (PER, ORG, LOC, MISC), we only are able to align fills of these types.

It is only possible to align 210 annotations into sentences in this way. We re-annotate these annotations with one of three labels: *correct*, *ambiguous* or *uncertain*, and perform this re-annotation relative to only the sentence that the two entities are mentioned in. We identify 147 (70% of the 210 annotations) as correct, 37 (18%) as ambiguous and 26 (12%) as uncertain.

Ambiguous cases are primarily where background knowledge or inference is required beyond what information is in the sentence. These include simple cases that appear to be incorrect, such as Example 24, and more potentially ambiguous cases such as Example 25.

(24) **ambiguous relation:** (Spencer Pratt, `per:siblings`, Stephanie Pratt)
**context:** *Spencer Pratt* and *Stephanie Pratt* on Heidi's New Puppy

(25) **ambiguous relation:** (David Banda, `per:country of birth`, Malawi)
**context:** The granting of an 18-month interim custody order, which enabled Madonna to take young *David Banda* out of *Malawi* last year, sparked heated debate about adoption laws in a country where the number of orphans is surging as a result of AIDS.

The other notable category of ambiguous instances are where some kind of NE unpacking is required. In Example 26, the fact that Project Islamic Hope is `affiliated` with Islam comes more from Islamic, rather than the separate mention of Islam.

(26) **ambiguous relation:** (Project Islamic Hope, `org:political/religious affiliation`, Islam)

**context:** He was a gang member before converting to Islam and founding the group Project Islamic Hope, which describes its mission as fighting poverty and social injustice.

We include *uncertain* to categorise borderline cases, which are entirely related to `org:top members/employees` and `per:employee of`, and typically involve a question about a person's place in an organisation (some of these instances are clarified in later years of the task). For example, in Example 27, a senior official is likely not a top employee.

(27) The reform program "won't be sufficient by itself to steer the economy on a viable financial path," said John Lipskey, a *senior official* with the International Monetary Fund.

**Other sources** While not directly applicable to SF, many ACE relations can be mapped directly to slots. Xu et al. (2013) search Google for training samples, Angeli et al. (2013) also make use of similar web snippets.

We will further discuss annotation issues in Chapter 7.

### 3.5.3 Answer extraction

We now discuss answer extraction approaches. Many systems, particularly high-performing systems, use a number of components ensembled together for extraction, sometimes with a large amount of engineering work and parameter tuning. Hence, clearly separating these systems into different approaches is difficult. We will discuss the top performing systems for each year, as well as notable other systems, and identify general trends.

| system | score |
|---|---|
| NIL baseline | 79.4 |
| Bikel et al. (2009) | 77.9 |
| Varma et al. (2009) | 77.9 |
| McNamee et al. (2009) | 64.1 |
| Li et al. (2009) | 46.0 |
| Agirre et al. (2009) | 37.3 |

Table 3.8: Key results for TAC 2009.

#### TAC 2009

These approaches to TAC mostly focused on either adapting work on ACE to SF, or setting up baseline rule-based system. Results for 2009 are listed in Table 3.8. The metric for this year was an accuracy-based score and not particularly informative, if a system simply returned NIL for all fills it would out-perform all systems. Nevertheless, we include these results as some measure of official performance. The best performing team, IBM (Bikel et al., 2009), took their ACE-based KLUE system and mapped the relations from that system to TAC slots. KLUE uses a classifier-based RE approach following the work of Kambhatla (2004)—using a cascade of maximum entropy classifiers over all candidate mention pairs. The classification pipeline is made up of separate models for existence, relation type, argument order (these three models being key for SF) as well as for tense, modality and specificity (all elements of ACE annotation). Features are broad and cover structural, lexical, syntactic (constituency parse-based) and semantic role label features, as well as relations existing with other entities in the sentence. Stanford-UBC (Agirre et al., 2009) also followed a classifier-based approach, training a binary classifier for each slot using context $n$-grams as features.

Most other systems used pattern-based approaches. McNamee et al. (2009) generate patterns from training data, and select the most frequent to use as patterns for extraction. Siel_09 (Varma et al., 2009) and PRIS (Li et al., 2009) use simple hand-

| system | P | R | F1 |
|---|---|---|---|
| Byrne and Dunnion (2010) | 66.6 | 18.7 | 29.2 |
| Chen et al. (2010) | 28.7 | 27.9 | 28.3 |
| Castelli et al. (2010) | 31.0 | 25.9 | 28.2 |
| Lehmann et al. (2010) | 44.9 | 19.4 | 27.1 |
| Grishman and Min (2010) | 28.0 | 26.0 | 27.0 |
| Gao et al. (2010) | 14.0 | 14.4 | 14.2 |
| Surdeanu et al. (2010) | 24.1 | 8.2 | 12.3 |
| Varma et al. (2010) | 36.3 | 5.4 | 9.4 |

Table 3.9: Key results for TAC 2010.

coded rules, where extractions are made based on the co-occurrence of entity pairs and trigger words for each slot.

#### TAC 2010

As we discussed in Chapter 2, TAC 2010 established the basic format of the SF task. Chada et al. (2010), was the top team for 2010, but they used an additional annotated 2 million documents to source fills, and so we choose to disregard their results for this year.

Despite using mostly distinct approaches, the other top systems all had similar performance. Results for these systems are shown in Table 3.9. IIRG (Byrne and Dunnion, 2010) generate token and POS patterns from training data, and map slots to questions for use in an existing pattern-based QA system. This system has lower recall than the other top-performing systems but, as with other rule-based approaches, it is relatively precise, and has a substantially higher precision than the other systems. CUNY-BLENDER (Chen et al., 2010) ensemble a number of components: a pattern matching component that uses distant supervision from TAC KB to extract precise patterns for extraction; an ACE classifier-based system mapped to TAC slots; a QA system enhanced with heuristic cross-slot inference rules; and a Freebase and Wikipedia lookup of fills in the source documents.

| system | P | R | F1 |
|---|---|---|---|
| Sun et al. (2011b) | 35.0 | 25.5 | 29.5 |
| Byrne and Dunnion (2011) | 25.0 | 12.3 | 17.1 |
| Xu et al. (2011) | 21.0 | 13.0 | 16.0 |
| Surdeanu et al. (2011) | 14.1 | 13.0 | 13.5 |

Table 3.10: Key results for TAC 2011.

Results are filtered with a suite of rules based on features such as NE types and confidence, similarity between the query and fill, and dependency path constraints. A maximum entropy-based reranker is use to combine the components. IBM's Castelli et al. (2010) approach is similar to their 2009 system, with the addition of a set of manually-coded inference rules. Like CUNY-BLENDER, LCC (Lehmann et al., 2010) implement an ensembled system, with are an active learning classifier approach, template-based system and a set of hand-coded rules.

Other systems use a range of techniques, some using approaches similar to the top systems. NYU (Grishman and Min, 2010) make use of hand-coded token and dependency patterns. From a set of 34 patterns, they follow Agichtein and Gravano (2000) in bootstrapping over entity pairs to generate further patterns, bootstrapping only over high precision pairs. Generated patterns are manually inspected, resulting in 970 patterns for extraction. PRIS (Gao et al., 2010) use regular expression patterns based on keywords for each slot. Siel_10 (Varma et al., 2010) use hand-coded rules, and have low performance.

Several systems follow the work of Mintz et al. (2009) in implementing a distantly supervised classifier (Surdeanu et al., 2010).

#### TAC 2011

NYU (Sun et al., 2011b) with an F1 of 29.5% was the top performing team in TAC 2011 with a margin of 10 points over the next best team. They follow a similar approach to their 2010 system, with a small set of ensembled approaches primarily

| system | P | R | F1 |
|---|---|---|---|
| Li et al. (2012) | 67.5 | 41.2 | 51.7 |
| Min et al. (2012a) | 45.0 | 22.0 | 29.6 |
| Roth et al. (2012) | 22.0 | 25.0 | 23.4 |

Table 3.11: Key results for TAC 2012.

based on patterns. They add a distantly supervised maximum extropy classifier component, trained on Freebase relations, with a precision-based tuple refinement filter. They also use this component to calculate precision of patterns extracted by bootstrapping. Performing analysis of each component on the system, they find that the combination of hand-coded patterns and the distantly supervised classifier were the best overall.

The next best performing teams, IIRG (Byrne and Dunnion, 2011) and LSV (Xu et al., 2011), follow their respective approaches from 2010, with results in Table 3.10. Stanford (Surdeanu et al., 2011) make some additions to their 2010 system. They convert their inference model to one based on Hoffmann et al. (2010) to allow for multi-label predictions. They incorporate model combination: as opposed to using the whole training set to train a single model, separate models each with distinct subsets of the training data are trained. These different models then have a plurality vote for the extracted label. They find that this combination approach provides a beneficial regularisation effect.

#### TAC 2012

PRIS (Li et al., 2012) scored highly in 2012 with an F1 of 52%. They use a purely pattern-based approach, bootstrapping from pairs from previous TAC evaluations. They use lexical and dependency patterns; it is unclear how they filter final patterns. Use of their released patterns in later years has not provided benefit to systems (Roth et al., 2013b), suggesting that the 2012 patterns may have been overfit to that particular year.

| system | P | R | F1 |
|---|---|---|---|
| Roth et al. (2013b) | 42.5 | 33.2 | 37.3 |
| Rawal et al. (2013) | 50.4 | 27.5 | 35.5 |
| Yu et al. (2013) | 40.7 | 29.0 | 33.9 |
| Xu et al. (2013) | 61.4 | 21.7 | 32.1 |
| Angeli et al. (2013) | 35.9 | 28.4 | 31.7 |
| Grishman (2013) | 53.8 | 16.8 | 25.6 |
| Singh et al. (2013) | 10.9 | 18.7 | 13.7 |

Table 3.12: Key results for TAC 2013.

The next best teams, NYU (Min et al., 2012a) and LSV (Roth et al., 2012) with F1s of 29.6% and 23.4% respectively (see Table 3.11), both use a combination of hand-coded patterns and distantly supervised classification. LSV derive training data using extractions from hand-coded patterns and by mapping Freebase to the source documents. A set of binary SVM classifiers are trained and tuned on development data. NYU's system is similar to their 2011 system, with improvements to query expansion and distant supervision. For distant supervision, they calculate the sentence-level PMI of entity pairs and only use pairs for training if they are above a threshold. They also train a set of maximum entropy models over the full training set, and use this to relabel the training set itself, the idea being that redundancy in the data allows a classifier to select instances more likely to express relations. Hand-coded and bootstrapped patterns are also used to correct distant supervised labels, as these patterns should have higher precision. These approaches are effective, with a 8-point F1 gain over their baseline.

#### TAC 2013

TAC 2013 saw a broader range of approaches to the task than in previous years, including sophisticated distant supervision models, OpenIE, and approaches incorporating unsupervised learning such as a universal schema model. Results for these differing approaches are shown in Table 3.12.

LSV's system (Roth et al., 2013b) was similar to their 2012 system, but with a focus on shallow techniques and features, which resulted in the top performance of 37.3% F1. They change their SVM feature set to use only shallow features: $n$-grams and skip $n$-grams. They make use of the Min et al. (2012a) aggregate training method, training a classifier on the distantly supervised training data, using this classifier to relabel the training data. Lexical token patterns extracted from distant supervision are scored according to frequency, and filtered with a noise-reduction approach, using a combination of a generative topic model and a discriminatively trained perception. Rawal et al. (2013) use hand-coded POS and word patterns, and achieve an F1 of 35.5%.

RPI_BLENDER (Yu et al., 2013) implement a large truth-finding ensemble system, focussing on a novel multi-dimensional truth-finding model, which constructs a knowledge graph based on the output of their system and evidence from a range of sources including multiple KBs. They apply a range of hard constraints, e.g. dependency path length, and soft features, such as voting across sources. Xu et al. (2013) use a pattern-based approach. They use hand-coded dependency patterns, and expand these by replacing tokens with synonyms, resulting in approximately 20,000 patterns. Stanford (Angeli et al., 2013) implement a MIML-RE based system, but primarily focus on pipeline components, particularly in the post filtering validation, using an extensive set of rules as a constraint satisfaction problem.

NYU (Grishman, 2013) continue their approach from 2012, with no substantial changes. UWash are the first team to use an OpenIE approach for SF. They use OpenIE to extract relations, and then manually map the relations that appear in development data to slots. Singh et al. (2013) make use of the Universal Schema approach. Using this approach they combine Freebase, the TAC KB and a subset of the TAC source to generate a KB. They then extract fills from this generated KB. Performance is low for this year, but sets up a baseline for this approach.

Yu and Ji (2016), evaluating on the TAC 2013 data, use an unsupervised approach to identify trigger words for person NE slots. They use an algorithm based on PageRank (Page et al., 1999) to identify the trigger words: the most prominent verbs, nouns or adjectives relating the query and fill entities together in the sentence. This identifies trigger candidates for each sentence, and a clustering algorithm is applied to the candidates for each sentence. The cluster with the highest average score becomes the trigger set for that given instance. For each slot type, they separately build a gazetteer of trigger words that indicate that that particular slot is expressed (e.g. for `per:spouse` this gazetteer would include wife, husband, marry). Extraction takes place by looking up the trigger set of a given instance in these gazetteers. They achieve an F1 of 57.4 on person NE slots, making this a state-of-the-art approach for those slots.

#### TAC 2014

At time of writing, TAC proceedings from 2014 and onwards have not been made publicly available, and so we cannot review those systems here. However, the task overview paper (Surdeanu and Ji, 2014) is publicly available, and so we refer to that. The top-performing Stanford system, based on the 2013 Stanford system achieved an F1 of 36.7% (recall 27.7%, precision 54.4%), largely due to the addition of a large amount of crowdsourced annotation (Angeli et al., 2014). This overview paper identifies some clear trends. The first is the prevalence of distant supervision: 14 of 18 teams made use of the technique. Many systems combine multiple approaches, including rule-based approaches, by simply combining results. Finally, it appears that machine-learning approaches out-perform rule-based. Our rule-based system (Pink and Curran, 2014), scored 0.6-points above the median F1 of 19.8%, was the only such system to score above the median. Overall, distantly-supervised classifier approaches have continued to improve on the state-of-the-art, but performance remains fairly low.

## 3.6 Summary

In this chapter, we have reviewed approaches to relation extraction, and how these techniques have been applied with varying success to slot filling. We identify approaches for acquiring additional training data as being of particular benefit to slot filling, as sparsity of training data is particularly problematic. Many previous approaches for RE had good performance on extracting explicit, short-range relations. However, extracting the more complex relations required for slot filling has proven to be more difficult. Sophisticated distantly supervised models have been used with some success, but we note that performance remains low overall, and it is likely much more training data is required. In addition, we identify that other semi-supervised approaches to acquiring more data have remained mostly unused in SF, except for straightforward bootstrapping techniques. In Chapter 4 we will continue to analyse this poor system performance, particularly in the context of recall. Chapter 5 will explore a graph-based approach to SF, and Chapter 6 will consider the issue of data sparsity.

# 4 Recall bounds

In the previous chapter, we broadly categorised approaches to slot filling. In this chapter, we contribute a detailed analysis of recall loss, as recall appears to be a major limiter on SF performance. SF systems typically use a pipeline of components, and while precision can be improved later in the pipeline, *candidates that are lost cannot be recovered later*. Hence, improving recall is a key concern for slot filling. Consider the official system scores for TAC SF 2013 (TAC13) in Table 4.1. The best TAC13 system scored 37.3% F-score (Roth et al., 2013b), and the median F-score was 16.9% (Surdeanu, 2013). Recall across systems is especially low, with many systems using precise, highly-engineered extractors with low recall. Precision ranges from 9% to 40% *greater than recall* for the top 5 systems in TAC13, and unsurprisingly, best-performing team (lsv) has the highest recall at 33%.

This gap is notable despite systems being tuned for F1: even a slightly higher recall would substantially improve the results of many of the top systems. Some systems do have higher recall than precision, but both the precision and recall of these systems are very low. There appears to be a clear gap between recall and precision across approaches. Closing this recall gap without substantially increasing the search space (and hence the computational cost), and without sacrificing precision, is critical to improving SF results.

We note that this is not as simple as a precision-recall trade-off: as we will discuss, generation of candidate fills sets a hard upper bound on recall. Candidates that are not generated at all can never be recovered by downstream processes.

| team | recall | precision | F-score | precision−recall Δ |
|---|---|---|---|---|
| lsv | 33.17 | 42.53 | 37.28 | +9.36 |
| ARPANI | 27.45 | 50.38 | 35.54 | +22.93 |
| RPI-BLENDER | 29.02 | 40.73 | 33.89 | +11.71 |
| PRIS2013 | 27.59 | 38.87 | 32.27 | +11.23 |
| BIT | 21.73 | 61.35 | 32.09 | +39.62 |
| Stanford | 28.41 | 35.86 | 31.70 | +7.45 |
| NYU | 16.76 | 53.83 | 25.56 | +37.07 |
| UWashington | 10.29 | 63.45 | 17.70 | +53.16 |
| CMUML | 10.69 | 32.30 | 16.07 | +21.61 |
| SAFT KRes | 14.99 | 15.67 | 15.32 | +0.68 |
| UMass_IESL | 18.46 | 10.88 | 13.69 | -7.58 |
| utaustin | 8.11 | 25.16 | 12.26 | +17.05 |
| UNED | 9.33 | 17.59 | 12.19 | +8.26 |
| Compreno | 12.74 | 9.74 | 11.04 | -3.00 |
| TALP_UPC | 9.81 | 7.69 | 8.62 | -2.12 |
| IIRG | 2.86 | 7.72 | 4.17 | +4.86 |
| SINDI | 2.59 | 7.84 | 3.89 | +5.25 |
| CohenCMU | 3.68 | 1.98 | 2.57 | -1.70 |
| LDC (human annotation) | 57.08 | 85.60 | 68.49 | 28.52 |

Table 4.1: All official scores in TAC 2013, with precision-recall differences.

However, precision can always be improved by downstream filtering, provided that the information needed to filter has not also been discarded.

In this chapter, we contribute an analysis framework that models techniques broadly in use across slot filling systems. We implement this framework as a series of filters over all possible candidates. We contribute a systematic recall analysis, pinpointing the cause of every candidate lost in typical pipelines, and estimate upper bounds on recall in existing approaches. Finally, we provide guidelines for system designers seeking to maximise recall, particularly in regards to NLP components used for candidate generation.

## 4.1 Error analysis for slot filling

Several authors have performed high-level analysis of slot filling errors. Ji and Grishman (2011b) and Min and Grishman (2012) identify many of the challenges of SF. They find that slot fills are expressed in a large variety of ways, and substantial analysis and inference is required to identify whether a slot fill is expressed in text. For the 140 TAC 2010 slot fills found by human annotators but not found by any system, Min and Grishman hypothesise sources of error, manually looking for evidence in the reference documents. They find inference, coreference and NER to be the top sources of error, and that the most studied component—sentence-level RE—is not the dominant problem, contributing only 10% of recall loss.

This post-evaluation approach is limited, as it only allows for a hypothesis of the likely source of recall loss for each type of answer, identifying the kind of answers that are lost, but not directly *how* they are lost. For instance, it is impossible to distinguish candidate generation errors from answer merging errors. Expected sources of these errors have often been identified anecdotally (Ji et al., 2011), without quantifying reasons for recall loss. Roth et al. (2014) report missing recall at a high level, identifying a 62.8% recall loss due to queries and fills not being correctly found in sentences by their TAC 2013 system.

In this chapter, we take this high-level analysis much further by performing a systematic recall analysis that allows us to pinpoint the cause of every recall error (candidates lost that can never be recovered) and estimate upper bounds on recall in existing approaches. We implement a collection of naïve SF systems utilising a set of increasingly restrictive filters over documents and named entities (NEs). TAC has three slot types: NE, string and value slots. In this chapter, and throughout this thesis, we consider only those slots filled by NEs as there are widely-used, high accuracy tools available for NER; identifying NE bounds and typing are major factors for accurate SF; and focusing on NEs only allows us to precisely gauge

performance of filters. String slots do not have reliable classifiers, and value slots require more normalisation than directly returning a token span. Otherwise, this evaluation is not specifically dependent on the nature of NEs, and we expect similar results for other slot types.

We focus on systems which first generate candidates and then process them, which is the approach of the majority of TAC systems. Our filters apply hard constraints over NEs commonly used in the literature, accounting for a typical SF candidate generation pipeline—matching the query term, the form of candidate fills and the distance between the query and the candidate—but not performing any further scoring or thresholding. Previous work (Gabbard et al., 2011) has identified the importance of coreference, and we compare several forms of coreference as filters, motivated by the need for efficient coreference resolution when processing large corpora. Complementing these unsupervised experiments, we implement a maximum recall bootstrap to identify which fills are reachable from training data, and the constraints implicitly applied by training data.

## 4.2 Why focus on recall?

While ultimately every system makes precision-recall trade-offs, a system's coarse candidate generation process sets a hard upper bound on recall, as candidates that are not generated at all can never be recovered by downstream processes. SF systems could generate every noun phrase in a corpus as potential candidates, but they apply hard candidate generation constraints for efficiency and precision.

We implement these hard constraints as a series of filters, and return every candidate which passes a filter without further ranking or thresholding. These filters are comprised of generic components, which are representative of SF pipelines, such as NER. We are only interested in precision in so much as it corresponds to the size of the search space (the candidates generated), assuming a small, fixed number

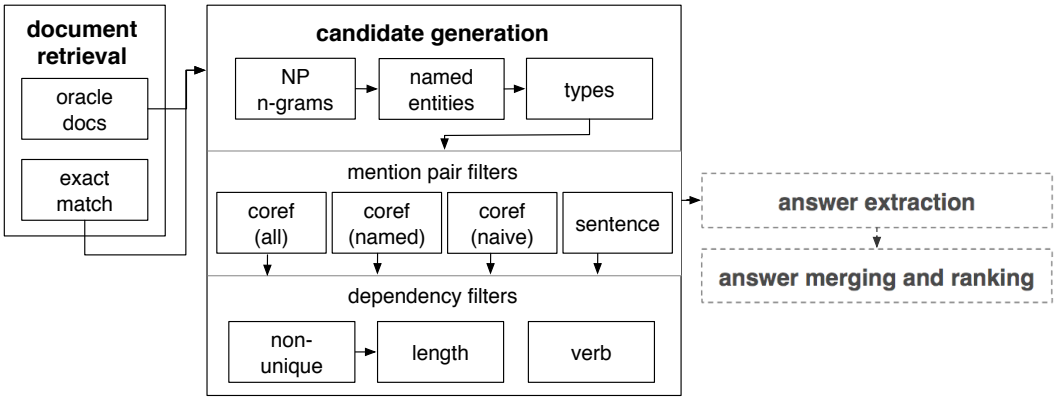



Figure 4.1: Candidate filters within the standard SF pipeline. Arrows indicate a sequence of filters. The components of this pipeline are detailed in Section 4.5.

of answers. The *search space determines the workload* of later stages responsible for extraction, merging and ranking, and effectively sets a lower bound for precision. Precision can be improved by post-processing the candidate set, but recall cannot.

## 4.3 Slot filling pipeline

SF systems typically consist of several pipelined stages (Ji et al., 2011), providing many potential locations for error. The basic pipeline, as described in Chapter 3 and shown in Figure 4.1, consists of four main stages (Ji and Grishman, 2011b): document retrieval, candidate generation, answer extraction, and answer merging and ranking. The output of the second stage is a set of candidates which are then classified as correct or incorrect answers by RE techniques. The components within these first two stages in Figure 4.1 are the filters that we will detail in Section 4.5. In this chapter, we focus on the first two stages, as they typically inadvertently filter correct answers that cannot be recovered, and they determine the size of the search space for later stages. An approach to these later stages is explored in Chapter 5.

In this chapter, we precisely characterise the contribution of these sources of error. We adopt filtering constraints imposed by the RE techniques used in the SF systems discussed in Section 3.5. As we have seen, features for RE typically

encode attributes of the entities; lexical sequences, dependency or constituency parse subtrees; and surrounding tokens (GuoDong et al., 2005; Mintz et al., 2009; Zhang et al., 2013).

## 4.4 Experimental setup

We begin with a set of TAC queries and, for each query, the documents known to contain any valid slot fill, as determined by oracle information retrieval from human annotation and judged system output. Filling every slot for the query with every $n$-gram in every document constitutes a system with nearly perfect recall,[1] representing every fill that exists as a sequence of tokens in the document. This baseline system is our starting point. We then apply a series of increasingly restrictive filters over this system's output. As in Figure 4.1, SF systems in practice must retrieve relevant documents and generate candidates. We propose filters that allow for analysis of recall lost during these stages. We ignore the remaining stages and evaluate the set of candidates directly.

Filters define what documents or NEs are allowed to pass through, based on constraints imposed by query matching, entity form, and sentence and syntactic context. We combine these filters in series in a number of configurations to model the progressively stricter constraints commonly applied by systems. Finally, we experiment with a bootstrapping training process, to reflect these constraints implicitly applied during a training approach, and measure how learnable test instances are from the training instances.

The SF typical system pipeline presented in Section 4.3 applies to most, but not all SF approaches. The following filters directly apply only to systems that use NER as the method of candidate generation, and where candidate generation is distinct from answer extraction. Fifteen of the eighteen teams participating

[1] As we will address later in this chapter, a small number of fills are not expressed verbatim in the text and required more sophisticated normalisation.

in TAC13 submitted system reports (Surdeanu, 2013). These fifteen systems are the key systems we will consider in this chapter, and we will note the portion of systems that each particular constraint reflects. As a starting point, fourteen of these fifteen systems identify NEs with NER and pass these to an answer extraction process. The assumption that fill spans are identified using automatic NER is core to most of our filters. The remaining TAC13 system does not rely on standard NER for candidate generation for `name` slots, opting to use POS tag patterns. We include a high recall baseline based on noun phrases (NPs) to cover this system.

## 4.5 Filters

The first step in a typical pipeline is to find documents which mention a query entity.[2] Here, we do not evaluate IR systems, but provide a setup intended to reflect two reasonable recall upper bounds. The first is the best case of retrieving all relevant target documents (the oracle); the second finds just those that contain the query entity verbatim. We use oracle IR to find target documents, this is ORACLE DOCS in Figure 4.1. To measure the effect of this oracle IR on recall, we implement a naïve EXACT MATCH filter, which allows a document only if an NE matches the query verbatim, reflecting our simpler second case.

We need to find a mention of the query entity in these documents for other filters and downstream stages. Finding entities which exactly match a query name in EXACT MATCH documents is trivial (because that is how they were retrieved). In the remaining oracle documents, an alias of the query entity is used, e.g. where the query Fyffes PLC is only mentioned as Fyffes in a document. In these cases, we manually annotate the longest token span which refers to the query as part of ORACLE DOCS. All of our candidate filtering begins with ORACLE DOCS.

[2]Some systems do not explicitly have this separate document retrieval step. They still need to identify and disambiguate query mentions in the context of a document.

| | entity | | | mention pair | | | | dependency | | |
|---|---|---|---|---|---|---|---|---|---|---|
| team | NP N-GRAMS | NAMED ENTITIES | TYPES | COREF (ALL) | COREF (NAMED) | COREF (NAÏVE) | SENTENCE | LENGTH | VERB | NON-UNIQUE |
| lsv | × | × | × | | | | × | | | |
| ARPANI | × | | | | | | × | × | | |
| RPI-BLENDER | × | × | × | × | | | | † | | † |
| PRIS2013 | × | × | × | × | | | | × | | × |
| BIT | × | × | × | | | × | | † | | † |
| Stanford | × | × | × | × | | | | ∗ | † | ∗ |
| NYU | × | × | × | × | | | | † | | † |
| UWashington | × | × | × | | × | | | ∗ | ∗ | ∗ |
| CMUML | × | × | × | × | | | | ∗ | | ∗ |
| SAFT KRes | × | × | × | × | | | | † | | † |
| UMass_IESL | × | × | × | × | | | | † | | † |
| utaustin | × | × | × | | | | × | | | |
| UNED | × | × | × | × | | | | × | | × |
| TALP_UPC | × | × | × | | | | × | | | |
| IIRG | × | × | | | | | × | | | |

Table 4.2: Candidate generation filters that apply to each TAC 2013 system, for those teams that submitted system reports. × indicates a hard constraint over the full system; † indicates a hard constraint over an ensembled component; and ∗ indicates where a filter is relevant for a system, but is not a hard constraint. Corresponding upper bounds are tight for systems marked with a ×.

The remainder of our filters relate to candidate generation. We identify where filters apply to specific systems in Table 4.2, and will discuss the system coverage of particular filters as they are described. As detailed in Section 3.5, many TAC systems are ensemble approaches, and so some constraints relate to specific components of a system but potentially not to the full system. It is possible that other ensembled components without that constraint may be able to prevent recall loss.

**Entity form filters** Entity form filters are based on the form of the entities extracted from documents. We include examples of both candidates allowed through the following filters and candidates rejected by them in Table 4.3. As high-recall, yet tractable, baseline, we initially use all substrings of all NPs. This NP N-GRAMS filter allows every $n$-gram of every NP, and captures fills that are directly extractable from text: the starting point for every system. NAMED ENTITIES allows NEs only, the typical setting for RE-based approaches. The TYPES filter requires fill NEs to be of an NER type defined by the slot, e.g. for `per:city of birth` only LOC NEs are allowed. As previously mentioned, all but one system makes use of NER, and only IIRG do not use strict NER types: they include NER types as features but do not include any definition-based rule for these slots.

**Mention pair filters** The previous filters allow fills to be returned from anywhere in a document, regardless of a reference to the query. However, RE techniques require arguments to be mentioned in the same sentence. As discussed in Section 2.5 these techniques are core to SF approaches, but are limited to sentence-level extractions. Mention pair filters apply this constraint, requiring the query mention and candidate fill be mentioned together in a sentence, filtering out candidates where this is not the case.

Different mention pair filters are further defined to reflect different types of coreference resolution. While we could apply these filters in sequence, systems do not typically implement multiple types of coreference, and so we will evaluate

| filter | allowed candidate | rejected candidate |
|---|---|---|
| NP N-GRAMS | **query:** Bob Dillinger<br>**candidate:** Pinellas County<br>**context:** Bob Dillinger, the *Pinellas County* public defender, is refusing to extend a contract with St. Petersburg to protest what he calls excessive arrests of homeless people in the city. | **query:** Bob Dillinger<br>**candidate:** Pinellas County<br>**context:** "We run into that quite frequently," said Bob Dillinger, chief public defender in Pinellas and Pasco counties. |
| NAMED ENTITIES* | **query:** John Kerry<br>**candidate:** Senate Small Business and Entrepreneurship Committee<br>**context:** "Long-term recovery for the Gulf Coast requires a whole lot more than 18 months of empty promises," said Democratic Senator John Kerry, chairman of the [*Senate Small Business and Entrepreneurship Committee*]$_{\text{ORG}}$. | **query:** John Kerry<br>**candidate:** Senate Committee on Small Business and Entrepreneurship<br>**context:** He was elected to the Senate in 1984 , and currently serves as chairman of the [*Senate Committee*]$_{\text{ORG}}$ *on* [*Small Business*]$_{\text{MISC}}$ *and* [*Entrepreneurship*]$_{\text{MISC}}$. |
| TYPES* | **query:** Red Sox<br>**slot:** `org:member of` (ORG-ORG)<br>**candidate:** Major League Baseball<br>**context:** Lyons and the *Red Sox* say they aren't aware of any other [*Major League Baseball*]$_{\text{ORG}}$ team with such an arrangement. | **query:** Red Sox<br>**slot:** `org:member of` (ORG-ORG)<br>**candidate:** American League East<br>**context:** The *Red Sox* held on to win the [*American League East*]$_{\text{MISC}}$ for the first time in a dozen years, and they are alive in the postseason, if barely, while the Yankees spend their days debating the fate of Joe Torre. |

Table 4.3: Example candidates for document-level filters. Square brackets indicate automatic NER where relevant. * indicates that the error example is due to automatic NER, gold NER would result in no rejected candidates for these filters. Query and candidate fill are in italics.

these filters independent of each other. We include examples of candidates allowed through these filters at Table 4.4.

The COREF (ALL) filter allows candidate fills only where the query and the fill are mentioned in the same sentence, either as named, nominal or pronominal mentions. This is the setting for the majority of systems (8/15 of TAC13 systems), which use coreference resolution to capture mentions of entities that may not be mentioned canonically in a query or sentence. COREF (NAMED) is a similar filter to COREF (ALL), but only allows named mentions: the query and the fill must have coreferent named mentions in the same sentence. These named mentions are from the full coreference resolution process. UWashington is the only team to use such an approach, in their case resolving entity mentions in OpenIE-style extractions. Despite most systems using full coreference, we consider this configuration to be interesting, particularly as proper noun named coreference is arguably an easier task than full coreference. For comparison, we include the COREF (NAÏVE) filter, which considers naïve rule-based proper noun coreference. Team BIT uses a simple version of this in the form of resolving acronyms, several other teams similarly make use of acronyms and alias generation as part of IR (but not strictly as part of the candidate generation step. We will detail the coreference resolution techniques used to implement these filters in Section 4.7. Finally, SENTENCE models the most straightforward approach, where no coreference is used: the query and the fill must be named verbatim in the same sentence for the candidate fill to be allowed through the filter. Interestingly, both the best and worst performers from TAC13 do not make use of coreference. That the best team does not use coreference suggests that this might be somewhat orthogonal to good performance, at least at this performance level: pipeline error from coreference resolution may be substantial.

**Syntactic filters** Dependency paths are often a key feature for extracting relations. We apply further syntactic filters based on dependency paths between NES

| filter | allowed candidate |
|---|---|
| COREF (ALL) | **query:** John Negroponte<br>**slot:** `per:schools attended`<br>**candidate:** Exeter Academy<br>**context:** [*John Negroponte*]$_{\text{PER}}$, due to be named Friday as US Deputy Secretary of State, is known as a "diplomat's diplomat" and is also intimately identified with the battered US operation in Iraq.<br>Schooled at the elite [*Exeter Academy*]$_{\text{ORG}}$ and then Yale University, he married socialite Diana Villiers, whose father was a former chairman of British Steel. |
| COREF (NAMED) & COREF (NAÏVE) | **query:** Fyffes PLC<br>**slot:** `org:top members/employees`<br>**candidate:** David McCann<br>**context:** Investment and consultancy group DCC PLC agreed to pay the sum principally to banana giant [*Fyffes PLC*]$_{\text{ORG}}$ to compensate for DCC's euro85 million (US$135 million) in profits from the February 2000 sale of Fyffes shares.<br>...<br>The case pitted Flavin, one of Ireland's most successful entrepreneurs who now serves as DCC chairman, against his former close friend and longtime business partner, [Fyffes]$_{\text{ORG}}$ chairman [*David McCann*]$_{\text{PER}}$. |
| SENTENCE | **query:** Sean Ross<br>**slot:** `per:employee of`<br>**candidate:** Edison Media Research<br>**context:** "It has to have some tempo, it has to have some energy," said [*Sean Ross*]$_{\text{PER}}$, vice president of music and programming at [*Edison Media Research*]$_{\text{PER}}$, which surveys the radio industry. |

Table 4.4: Example allowed candidates for mention pairs filters. Each filter rejects all example candidates higher in the table. Square brackets indicate relevant NEs, underline indicates relevant mentions. Query and candidate fill are in italics.

and mentions in sentences. Where we use dependencies, we use the Stanford collapsed and propagated representation (de Marneffe and Manning, 2008), e.g. in the sentence Alice is an employee of Bob and Charlie the collapsed and propagated dependency path between Alice and Charlie is [ $\xrightarrow{\text{nsubj}}$ employee $\xleftarrow{\text{prep of}}$ ]. We always use the lexicalised form of dependency paths, e.g. in the sentence Alice is an employed by Bob the path between Alice and Bob is [ $\xrightarrow{\text{nsubj}}$ employ $\xleftarrow{\text{prep by}}$ ].

Unlike the previous filters, use of dependency paths does not always create strictly hard constraints. Where dependency paths are used as part of rule-based or semi-supervised pattern extractors, it is the case that the following dependency path filters create hard constraints. In Table 4.2, this is the case for a small number of teams which rely heavily on such patterns, most notably PRIS2013. Many teams incorporate a dependency pattern-based component, or use dependency paths as one of a small number of path features: these teams are marked with a †. Teams which instead incorporate dependency paths as features in a much larger feature space are marked with a $*$. The top team for TAC13 did not incorporate dependency paths. However, this appears to be an outlier for TAC: as discussed in Section 3.5, syntactic information in the form of dependency paths is common in top performing systems.

As with the dependency paths themselves, these filters characterise the complexity of the syntax connecting the query and filler. Examples of allowed and rejected candidates are shown in Table 4.5. LENGTH $\leq$ N requires that the query and fill are separated by a dependency path of at most N arcs, e.g. the path [ $\xrightarrow{\text{nsubj}}$ employee $\xleftarrow{\text{prep of}}$ ] is two arcs. Most systems use a length constraint as a sanity check, but some systems make explicit use of length constraints for particular slot: Yu et al. (2013) (RPI-BLENDER) note that syntactically distant candidates are typically incorrect fills. VERB requires a verb to be present in the dependency path between the query and fill mentions or names. While a verb constraint on dependency paths is rare, OpenIE components, particularly those based on ReVerb (Fader et al.,

| filter | allowed candidate | rejected candidate |
|---|---|---|
| LENGTH $\leq$ N | **query:** Konica Minolta<br>**slot:** `org:country of headquarters`<br>**candidate:** Japan<br>**context:** [*Japan*]$_{\text{ORG}}$ 's [*Konica Minolta*]$_{\text{ORG}}$ said Thursday its net profit nearly ...<br>**path:** [ $\xrightarrow{\text{poss}}$ ] | **query:** Sean Ross<br>**slot:** `per:employee of`<br>**candidate:** Edison Media Research<br>**context:** ... says [*Sean Ross*]$_{\text{PER}}$, vice president of music and programming at [*Edison Media Research*]$_{\text{ORG}}$.<br>**path:** [ $\xleftarrow{\text{prep at}}$ president $\xrightarrow{\text{appos}}$ ] |
| VERB | **query:** Badr Organization<br>**slot:** `org:top members / employees`<br>**candidate:** Hadi al-Amiri<br>**context:** ... and [*Hadi al-Amiri*]$_{\text{PER}}$ , who heads the [*Badr Organization*]$_{\text{ORG}}$n , the armed ...<br>**path:** [ $\xleftarrow{\text{dobj}}$ head $\xrightarrow{\text{nsubj}}$ ] | **query:** Badr Organization<br>**slot:** `org:top members / employees`<br>**candidate:** Hadi al-Amiri<br>**context:** ... and [*Hadi al-Amiri*]$_{\text{PER}}$ , the head of the [*Badr Organization*]$_{\text{ORG}}$ , the armed ...<br>**path:** [ $\xleftarrow{\text{prep of}}$ head $\xleftarrow{\text{appos}}$ ] |
| NON-UNIQUE | **query:** Chen Zhu<br>**slot:** `per:member of`<br>**candidate:** Chinese Academy of Sciences<br>**context:** Also on the list is the appointment of [*Chen Zhu*]$_{\text{PER}}$, former vice president of the [*Chinese Academy of Sciences*]$_{\text{ORG}}$ (CAS), as the minister of health by China's top legislature.<br>**path:** [ $\xleftarrow{\text{prep of}}$president $\xrightarrow{\text{appos}}$ ] | **query:** Chen Zhu<br>**slot:** `per:member of`<br>**candidate:** Institute of Medicine<br>**context:** China's Health Minister [*Chen Zhu*]$_{\text{PER}}$ has been elected as foreign associate of the United States [*Institute of Medicine*]$_{\text{ORG}}$ (IOM), according to the IOM website.<br>**path:** [ $\xleftarrow{\text{nsubjpass}}$ elected $\xrightarrow{\text{prep as}}$ associate $\xrightarrow{\text{prep\_of}}$ ] |

Table 4.5: Example candidates for dependency filters. For LENGTH, we use N $= 1$ in this example. Square brackets indicate automatic NER where relevant. Query and candidate fill are in italics.

2011) require a verb in syntactic context. NON-UNIQUE models a slightly more subtle constraint, requiring the dependency path between the query and fill to occur more than twice in a corpus. This models the hard constraint on feature-based learning processes that require a feature to occur in both training and test examples to be useful. Of course, systems may incorporate additional features to account for lost recall. However, given the dependency paths are often good, discriminative features that capture a large amount of context, sparsity in paths is probably indicative of sparsity of a larger feature space over a similar context. This issue of sparsity will be further addressed in Chapter 6.

## 4.6 Bootstrapping reachability

In addition to the upper bound set by these explicit hard constraints, we want to reflect constraints that are implicitly applied by an extraction process. The most straightforward version of this is the NON-UNIQUE constraint, described above, but beyond this we want to more broadly consider if there are fills that are never learnable given a set of features and a set of training data.

We extend our evaluation to include a training process in a basic semi-supervised setting, following the bootstrapping approach described in Section 3.4.2 (Agichtein and Gravano, 2000). Given training pairs of query-fill NE pairs in text, we extract the context of each pair, and find other pairs in the corpus that share that context. A pair is reachable, and hence learnable, if it can be found by iterating this process. An example of this process is shown in Figure 4.2. The first Leslie Walker instance shares features with the Mohamed ElBaradei training instance, and the slot `per:employee of` also applies to it. By iteratively bootstrapping, we can find instances with transitively share features with training instances. The Jim Senn instance is one of these cases, found through the dependency path [ $\xleftarrow{\text{prep for}}$ director $\xleftarrow{\text{appos}}$ ] shared with an instance of Leslie Walker.

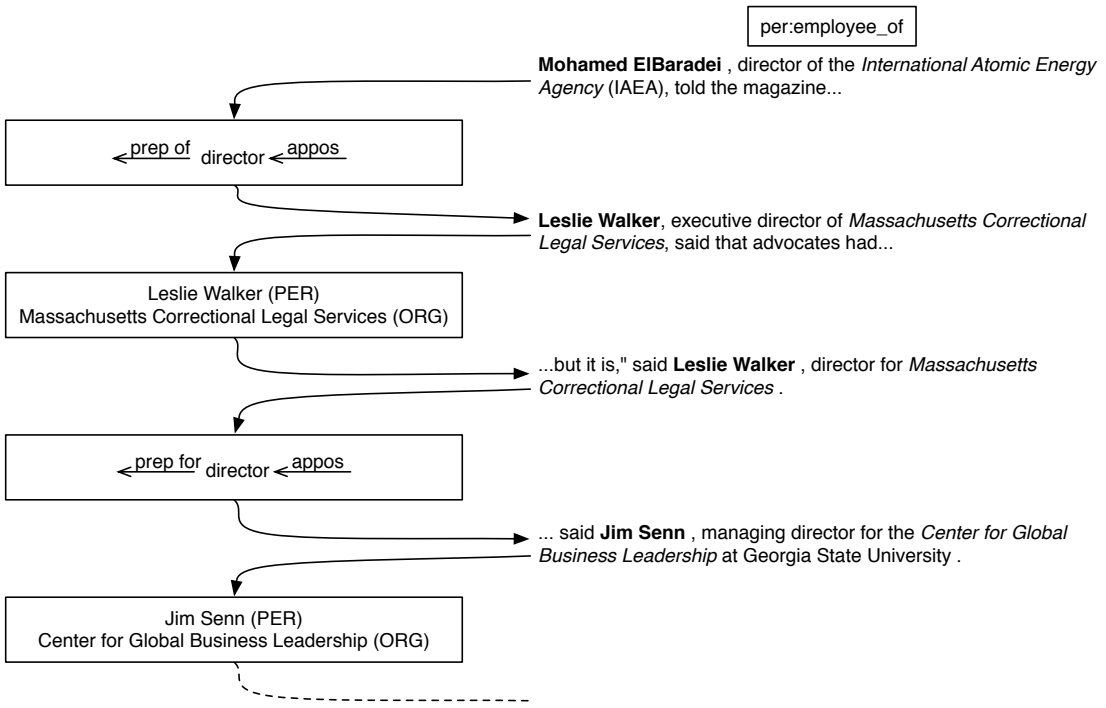



Figure 4.2: Bootstrapping. All nodes are labelled with `per:employee of` after two iterations.

We continue to evaluate maximum recall and do not apply thresholding or ranking that would typically be utilised in a bootstrapping process. This is not a practical approach for actual extraction, as bootstrapping processes will typically apply very strict constraints to find candidate: our process gives a very optimistic maximum recall. We output all possible candidates in order to measure recall loss. As with the hard constraints applied by our other filters, if recall is lost it can never be recovered.

We use lemmatised dependency paths as the context for this process as they are relatively precise and discriminative, compared to other features used for SF. In order to simplify processing, we construct a graph of all pairs and paths in the corpus first, and then bootstrap from training instances over this graph. a We will expand upon this idea in later chapters, but use this basic reachability setting here. This analysis is directly applicable to pattern-based and semi-supervised systems, giving us the maximum recall these systems could achieve, however optimistically.

The graph is constructed as follows. Each node represents a typed pair of NEs that occur in the same sentence in the TAC KBP Source Data (LDC, 2010), collapsing nodes that have equal names and types into a single node. An edge exists between pairs that are connected *at least once by the same dependency path*. The constructed graph is equivalent to the EXACT MATCH + SENTENCE + NON-UNIQUE filter. Constructing a graph for COREF (ALL) (which requires many more edges than SENTENCE) was impractical.

Initially, pairs in the training data are labelled with their corresponding slots. In each bootstrap iteration, the labels of each node are added to its neighbouring nodes. There is no filtering or competition between labels on a node, they are all added. The space complexity of this algorithm is $O(|V|)$ - worst case, the path will be the length of the entire graph, and each visited node will need to be stored in memory. Time complexity is $O(|V| + |E|)$, as algorithmically this is a standard depth-first search (each node is visited once, but worst case every edge but be checked to see if its corresponding node has been visited). We analyse performance after each iteration, evaluating by mapping the labelled graph back to the equivalent SF queries. This enables us to determine what fills are recoverable from the bootstrapping process.

## 4.7 Evaluation

We evaluate our filters on the TAC KBP English Slot Filling 2011 corpus, queries and task specification presented in Chapter 2.

For efficiency when evaluating the filters, we use only the documents from the TAC KBP Source Data (LDC, 2010) that are known to contain at least one correct slot fill in the TAC KBP 2011 English Slot Filling Assessment Results (LDC, 2011): this is appropriate as we target recall upper bounds. Also, we are not evaluating IR systems and this extra workload is a secondary issue. We refer to this set of

documents as TEST. We use the full corpus for the NON-UNIQUE filter and the bootstrapping reachabilty experiments.

We restrict the assessment results and the evaluation process to all slot types that are filled by `name` content types as opposed to `value` or `string`. We also do not evaluate the `per:alternate names` or `org:alternate names` slots, as fills for these slots are rarely in the same sentence as queries. While X also known as Y or similar may appear in text, X and Y are typically mentioned independently.

There are 100 TAC 2011 queries, 50 PER and 50 ORG. There are 535 fills in our reduced evaluation, 1,171 correct responses over these fills: 56% of the original evaluation fills. The distribution of these fills is shown in Table 4.6. These fills are dominated by employment, organisation hierarchy, and location-related slots, skewing the evaluation towards these types. 44% of fills are for `org:top members/employees`, `per:employee of` or `per:member of`; 11% for `org:subsidiaries` or `org:parents`; and 20% for `per:* of residence` or `per:* of headquarters`. The number of fills per query ranges from 0 (one query has no `name` fills) to 71, with a median of 17. TEST is comprised of 1,351 documents. The number of documents per query ranges from 0 to 63, with a median of 15.5. We use TAC 2009 and 2010 results and annotations as training data for bootstrapping, with 4,647 relevant training examples. We evaluate ignoring case and without requiring a specific source document: `nocase` and `anydoc` in the SF evaluation.

We preprocess documents with Stanford CoreNLP: tokenisation, POS tagging (Toutanova et al., 2003), NER (Finkel et al., 2005), parsing (Klein and Manning, 2003), and coreference resolution (Lee et al., 2011), and these annotations form the relevant components of our filters. Where we use dependency paths, we lemmatise tokens on the path to increase generality and recall in further analysis. For example, for Alice employs Bob we extract the path $[\ \xleftarrow{\text{nsubj}}\ \text{employ}\ \xrightarrow{\text{dobj}}\ ]$ between Alice and Bob. We use CoreNLP as it is representative of the NLP pipeline used by SF systems.

| slot | # | % | Σ % |
|---|---|---|---|
| org:top members/employees | 118 | 22 | 22 |
| per:employee of | 71 | 13 | 35 |
| per:member of | 47 | 9 | 44 |
| org:subsidiaries | 32 | 6 | 50 |
| org:parents | 24 | 4 | 55 |
| per:origin | 23 | 4 | 59 |
| org:country of headquarters | 22 | 4 | 63 |
| per:countries of residence | 20 | 4 | 67 |
| org:city of headquarters | 19 | 4 | 70 |
| org:shareholders | 18 | 3 | 74 |
| per:cities of residence | 17 | 3 | 77 |
| per:children | 17 | 3 | 80 |
| org:stateorprovince of headquarters | 17 | 3 | 83 |
| per:schools attended | 16 | 3 | 86 |
| per:stateorprovinces of residence | 11 | 2 | 88 |
| org:member of | 11 | 2 | 90 |
| per:spouse | 8 | 1 | 91 |
| org:members | 8 | 1 | 93 |
| org:founded by | 7 | 1 | 95 |
| per:siblings | 6 | 1 | 96 |
| per:other family | 6 | 1 | 97 |
| per:city of birth | 6 | 1 | 98 |
| per:parents | 3 | 1 | 99 |
| per:country of birth | 3 | 1 | 99 |
| org:political/religious affiliation | 2 | 0 | 99 |
| per:stateorprovince of birth | 1 | 0 | 100 |
| per:country of death | 1 | 0 | 100 |
| per:city of death | 1 | 0 | 100 |

Table 4.6: Number of fills in the evaluation.

Of the fifteen TAC 2013 systems for which we have system descriptions, ten of these make use of CoreNLP, eight of these specifically make use of CoreNLP NER.

The COREF (NAÏVE) filter uses CoreNLP coreference, limited to mentions which are headed by NNPS. For COREF (NAÏVE) we use a naïve rule-based coreference

process (Pink et al., 2013), motivated by reasons of efficiency, as the full CoreNLP requires parsing and a more complex model. These naïve rules do not require deep processing and can run quickly over large volumes of text. The rules are as follows. All NEs from a document are matched by processing in decreasing length order. NEs are normalised for case, and honorifics such as Mr. are removed. Two names are marked coreferent where: they match exactly (e.g. Smith matches Smith; and Smith matches Mr. Smith as honorifics are ignored); they have a matching final word (e.g. Smith matches John Smith); they have a matching initial word (e.g. John matches John Smith); or one is an acronym of the other (e.g. USA matches United States of America). If multiple conditions are matched, the earliest match is used.

The NON-UNIQUE filter requires that a dependency path occurs more than once between NEs in the full TAC KBP Source Data (LDC, 2010), comprised of 1.8M documents and 318M NE pairs. There are 38.6M distinct lemmatised dependency paths, 5M of which occur more than once.

## 4.8 Results

We now analyse where the filters lose recall. Results for non-syntactic filters are listed in Table 4.7. Figure 4.3 illustrates our main pipeline which contains filters that would typically be implemented.

**NP n-grams** We choose all n-grams of NPs at all levels of parse trees (from the CoreNLP constituency parser) to be our highest recall filter, and so our highest baseline has 3% recall loss, with 14 errors. Four of these errors are due to the fill not existing verbatim in text, e.g. Pinellas and Pasco counties does not contain Pinellas County verbatim. Four errors occur where an NP is not correctly identified, which occurs in two different cases: where there is a genuine parser error, or where the "sentence" is actually a structure that cannot be handled by a parser (a list or other semi-structured data as opposed to an actual sentence). Six errors are

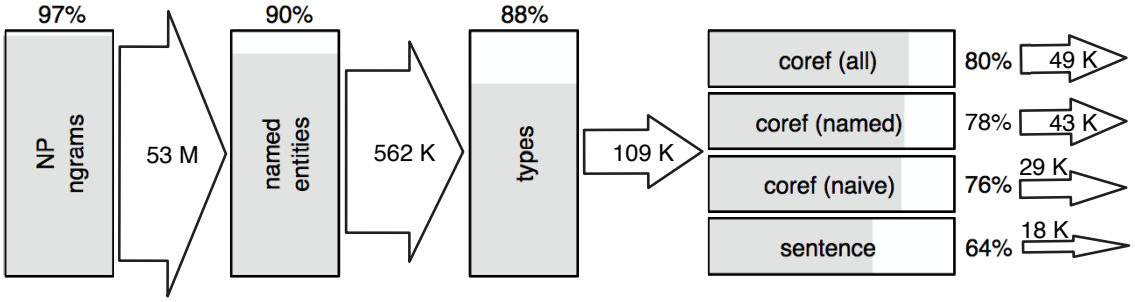



Figure 4.3: Core filter pipeline results: results for NP N-GRAMS + NAMED ENTITIES + TYPES, followed by mention pair filters with a range of coreference configurations. Grey fill and % indicates recall after each filter, and the number in the arrow is the size of the result set passed to the next filter or to the downstream process.

where fills cannot be extracted due to tokenisation or preprocessing differences. These are cases where trailing punctuation or HTML entities have been included in fills. We can never find fills such as International Center for Reproductive Health. because the trailing period is a separate token in our setup. We refer to these as PREPROCESSING errors. Potentially these instances should have been labelled as inexact in the results, but these cases are rare and not a major concern.

While 97% recall is an excellent starting point, 53M candidates for only 100 queries is a huge, likely an impractical search space for any downstream process. Hence NER is commonly used as the starting point for SF.

**NAMED ENTITIES** Most errors here are due to NER errors, and these 38 errors result in nearly a 10% recall loss. 25 errors are caused where no token in the fill has been tagged as part of a NE (NO NER); and 13 where some tokens were missed (NER BOUNDS). Hence, in agreement with previous analyses (Min and Grishman, 2012), NER error has a large impact on SF.

On this data set we have 10% recall loss that most SF or RE approaches would never be able to extract. However, it is still quite unconstrained and a high recall bound in comparison to the following filters. Recall errors could be substantially

| experiment | R (%) | \|search space\| | % (1) |
|---|---|---|---|
| NP N-GRAMS | 97 | 53,966,773 | 49,401 |
| ...+ NAMED ENTITIES | 90 | 562,318 | 515 |
| ...+ TYPES (1) | 88 | 109,241 | 100 |
| EXACT MATCH + ... (2) | 85 | 105,764 | 97 |
| (1) + COREF (ALL) | 80 | 49,170 | 45 |
| (1) + COREF (NAMED) | 78 | 43,476 | 40 |
| (1) + COREF (NAÏVE) | 76 | 29,171 | 27 |
| (1) + SENTENCE | 64 | 18,331 | 17 |
| (2) + COREF (ALL) | 77 | 47,439 | 43 |
| (2) + COREF (NAMED) | 73 | 30,089 | 28 |
| (2) + COREF (NAÏVE) | 73 | 27,770 | 25 |
| (2) + SENTENCE | 61 | 16,978 | 16 |
| (1) + COREF (ALL) + NON-UNIQUE | 65 | 19,958 | 18 |
| (1) + COREF (NAMED) + NON-UNIQUE | 62 | 17,692 | 16 |
| (1) + COREF (NAÏVE) + NON-UNIQUE | 61 | 13,960 | 13 |
| (1) + SENTENCE + NON-UNIQUE | 48 | 8,084 | 7 |
| (2) + COREF (ALL) + NON-UNIQUE | 63 | 18,953 | 17 |
| (2) + COREF (NAMED) + NON-UNIQUE | 60 | 16,712 | 15 |
| (2) + COREF (NAÏVE) + NON-UNIQUE | 56 | 13,064 | 12 |
| (2) + SENTENCE + NON-UNIQUE | 43 | 7,236 | 7 |

Table 4.7: Results on TEST given sets of filter configurations. The ellipses indicate the previous line. *% (1)* indicates size of search space relative to configuration (1).

reduced if SF approaches were to take into consideration all NEs in documents as a set of candidates, and capture entity pairs across sentences outside those found by coreference resolution. While there has been some work in extracting relations across sentences without coreference (Swampillai and Stevenson, 2011), RE across sentence boundaries is effectively limited to coreference chains between sentences. Currently, whole document extraction is not a research focus for SF, and the implementation of whole document techniques throughout SF pipelines would likely be beneficial. Allowing k-best NER outputs to be passed to later stages in the pipeline would allow for up to 7% recall to potentially be regained.

| Experiment | COREF (ALL) | SENTENCE |
|---|---|---|
| NN COREF FAILURE | 9 | 16 |
| NNP COREF FAILURE | 6 | 52 |
| PRP COREF FAILURE | 13 | 20 |
| ROLE INFERENCE | 4 | 4 |
| LOCATION INFERENCE | 3 | 3 |
| GENERAL INFERENCE | 0 | 2 |
| NO NER | 9 | 17 |

Table 4.8: Error types for COREF (ALL) and SENTENCE.

**TYPES** All errors created by the TYPES filter are due to incorrect NER types on mentions proposed by CoreNLP. We do not aggregate the NE type over the coreference chain when using coreference resolution. Applying this filter cuts down the search space substantially, with minimal loss to recall. Adding TYPES results in a recall loss of 2% (12 errors), but cuts down the search space by 80%.

**EXACT MATCH** This filter is present in Table 4.7, but is not included in Figure 4.3. Requiring that the query name is exactly matched as part of document retrieval (EXACT MATCH) loses a 2% recall, as 13 fills are not found. Effectively this is the recall error created by the IR component of SF. Five error cases occur when an alias is required, e.g. Quds Force for IRGC-QF and Chris Bentley for Christopher Bentley. Eight errors occur where the query term is a reference to an entity but not its name, all pertaining to the query GMAC's Residential Capital LLC.

**COREF (ALL)** This filter is the starting point for many recent SF approaches: fills that are mentioned in the same sentence as queries. Table 4.8 shows that largest category of recall loss due to the COREF (ALL) filter are fills that are actually mentioned in the same sentence, but lost due to errors in the automatic coreference resolution we have used for evaluation. NNP COREF FAILURE, NN COREF FAILURE,

and PRP COREF FAILURE indicate failure to resolve named, nominal and pronomial coreference respectively.

Example 1 shows an instance of NNP COREF FAILURE: the underlined mention of Fyffes is not correctly resolved to the query Fyffes PLC.

(1) **candidate lost:** (Fyffes PLC, `org:top members/employees`, David McCann)
**context:** Investment and consultancy group DCC PLC agreed to pay the sum principally to banana giant *Fyffes PLC* to compensate for DCC's euro85 million (US$135 million) in profits from the February 2000 sale of Fyffes shares.
. . .
The case pitted Flavin, one of Ireland's most successful entrepreneurs who now serves as DCC chairman, against his former close friend and longtime business partner, Fyffes chairman *David McCann*.

Example 2 shows a NN COREF FAILURE case, where the Hong Kong park is not resolved to Hong Kong Disneyland, and Example 3 provides a NRP COREF FAILURE case where He is not resolved to John Negroponte.

(2) **candidate lost:** (Hong Kong Disneyland, `org:top members/employees`, Andrew Kam)
**context:** The Walt Disney Co. said Thursday a former Coca-Cola executive with 20 years of China experience has been appointed head of *Hong Kong Disneyland*. *Andrew Kam* has been named managing director of the Hong Kong park, Walt Disney Parks and Resorts said in a statement.

(3) **candidate lost:** (John Negroponte, `per:schools attended`, Exeter Academy)
**context:** *John Negroponte*, due to be named Friday as US Deputy Secretary of State, is known as a "diplomat's diplomat" and is also intimately identified with the battered US operation in Iraq.
. . .

> Schooled at the elite *Exeter Academy* and then Yale University, he married socialite Diana Villiers, whose father was a former chairman of British Steel.

The remainder of the errors are cases where mentions of the fills do not occur in the same sentence as mentions of the query. ROLE INFERENCE indicates that an individual's role is mentioned, e.g. Gene Roberts, the executive editor, where The Inquirer is mentioned in a previous sentence. LOCATION INFERENCE errors require inference over additional location knowledge, e.g. a French company is headquartered in France. The search space has been substantially reduced, by a further 55% to 0.1% of the original space. However, the recall upper bound has dropped to 80% of all fills.

**COREF (NAMED) and COREF (NAÏVE)** While coreference is important for high recall, more difficult coreference cases (common noun and pronoun coreference) may generate a large number of spurious cases. Using COREF (NAMED) as the mention pair filter loses 2% recall, to an upper bound of 78%, for a 12% reduction in the search space. However, using a full coreference system generates may more candidates than using simple proper noun coreference. COREF (NAÏVE) has an upper bound of 76%. This is only 4% lower recall than COREF (ALL), but for a 41% reduction in search space. In addition, CoreNLP coreference is much more computationally expensive than our naïve approach as it requires parsing.

**SENTENCE** Errors for SENTENCE are listed in Table 4.8. GENERAL INFERENCE indicates that inference or more sophisticated analysis is required to find the fill, such as correctly identifying the relation between entities referred to in an interview.

SENTENCE results in a recall upper bound of 64%. While this gives us a small search space, we are now losing a substantial proportion of the correct fills.

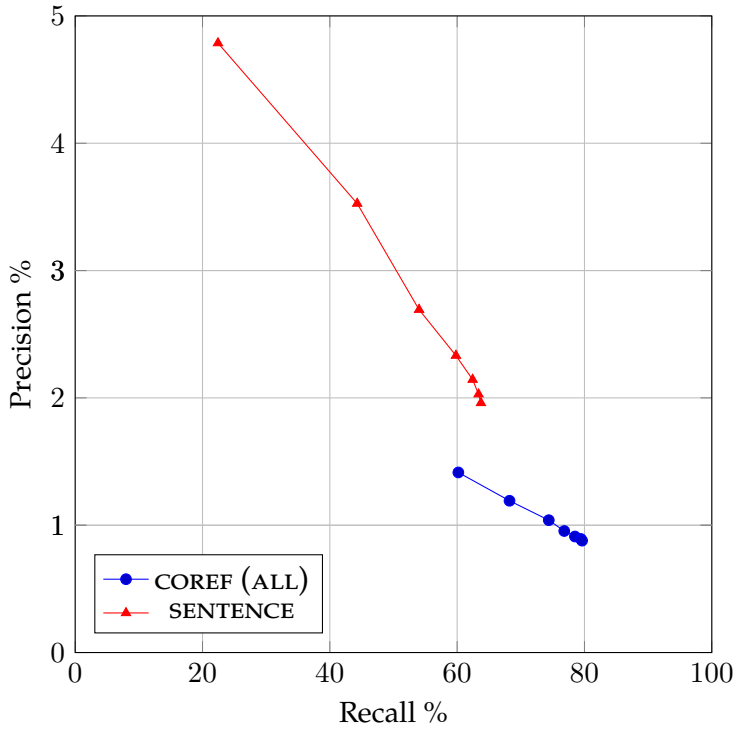



Figure 4.4: Effect of COREF (ALL).

### 4.8.1 Dependency filters

Precision-recall curves for the dependency path filters are given in Figures 4.4, 4.5 and 4.6. Dots from low recall to high recall indicate maximum dependency path length from $n = 1$ to $n = 7$. Dependency paths of length 7 give maximum recall in our experiments. Results for the addition of the NON-UNIQUE constraint are given in Table 4.7. Note that the scales on the graph axes are very different! In a typical system, the precision will be high, and so gaining precision is not that helpful. Gaining recall will have greater effect on F1.

**Use of coreference** While critical for recall, use of coreference allows a large number of candidates and presents a key trade-off for SF, as demonstrated by Figure 4.4. At maximum dependency path length, coreference gives 16% greater recall at a cost of 1.1% precision, roughly half the precision of no coreference.

Higher precision indicates that fewer candidates are generated. Fewer candidates allows for SF approaches to be scaled to larger amounts of data, and enables techniques that take advantage of redundancy or clustering to be used. Hence the higher precision no coreference approach may allow for more precise learning methods to be used, which may provide better results overall than an approach using coreference.

**Short dependency paths** In all of our filter configurations, a short dependency path length is sufficient for extracting the majority of slot fills for that particular configuration. Improving precision of fills found on short dependency paths may be a more effective and scalable approach to improving F1 rather than focusing on long paths. However, the gain in recall is substantial enough to make use of all paths worthwhile.

In Figure 4.5 we consider SENTENCE. Limiting the dependency path length to $n = 3$ loses 11% recall, but gains 0.7% precision. While this loss of recall is high, the reduction in unique dependency paths is substantial. For maximum path length three there are 10,732 paths (1,551 unique); for all paths there are 17,394 paths (2,863 unique).

**Verb** Figure 4.6 shows the VERB filters has less impact or recall or precision than some other dependency filters. For COREF (ALL) with all paths, adding the VERB filter loses 6% recall for a 0.1% gain in precision. Some slots not included in this analysis, such as `per:title`, tend to be described by shorter paths that often do not include verbs. These slots are also frequent in the TAC11 dataset.

**Non-unique** The frequency of a dependency path may be a critical feature for learning, as paths that occur only once will not been seen by a bootstrapping process or may not be considered by other machine learning approaches. Applying the NON-UNIQUE filter (Table 4.7) has a large effect on recall: COREF (ALL) loses 15%

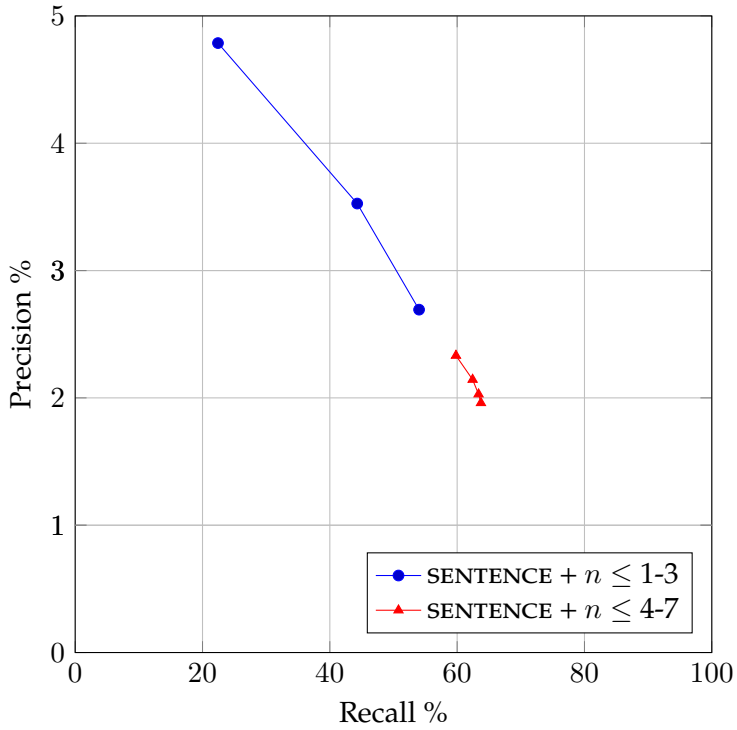



Figure 4.5: Effect of short paths, taking the SENTENCE points from Figure 4.4.

recall for a 41% reduction in the size of the search space; SENTENCE loses 15% recall for a 44% reduction in search space. To recover this recall, the strictness of this filter could be relaxed by further generalising dependency paths or using a different similarity metric to direct match of paths. However, this is the upper bound for approaches which consider only exact dependency paths as a feature.

**Bootstrapping** A small amount of training data quickly finds slot fills via bootstrapping. One iteration has a recall of 24%, with 7,665 candidates generated. Two to four iterations have recall of 37%–39% (maximum recall), with 31,702–37,797 candidates. The recall upper bound for these configurations is 43%—annotating more training data as seeds will allow for better precision, but may only minimally improve recall in this setup. Labelling additional seeds does not further connect or add more candidates back into the graph: the upper bound of 43% is enforced by the distribution of tuples in the corpus, rather than which tuples are seeds. We note that limiting bootstrap to one or two iterations is ideal for the best trade-

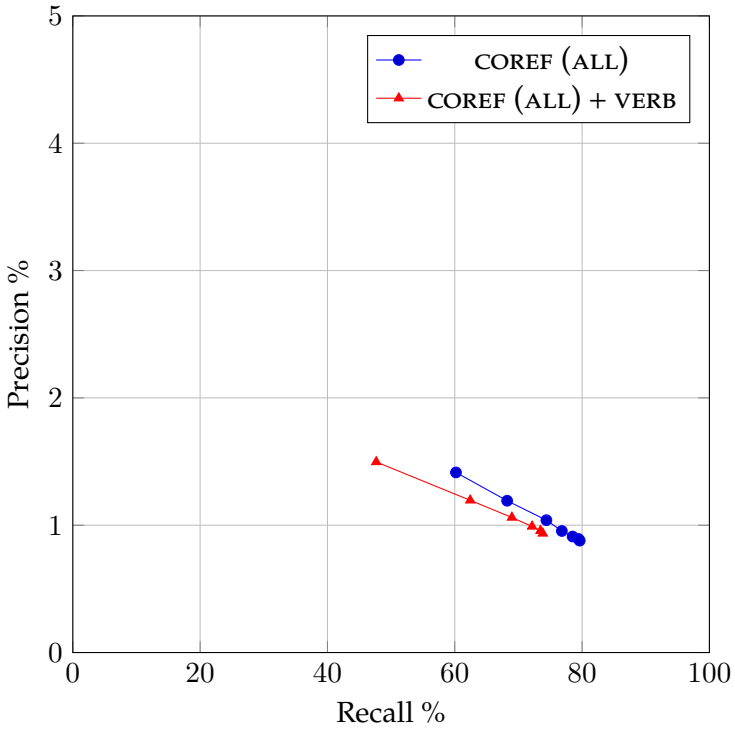



Figure 4.6: Effect of the VERB filter.

off between recall and search space. However, closer analysis of discriminative paths is required for a full SF system. Even if we included the test instances in the training instances, the recall would still be limited to 43%. This demonstrates that systems need distributional features, dependency tree kernels or other similarity comparison as opposed to exact feature matching if dependency paths are to be a useful feature for SF.

## 4.9 Discussion

We have presented an analysis of SF recall bounds given hard constraints applied by standard system components. Pipeline error is common across all NLP tasks, and SF, with a substantial NLP pipeline, has a substantial number of sources of potential error. For the most part, information retrieval is a relatively minor component of SF, with little recall to be gained outside of searching for documents that contain the query verbatim. No doubt this is partially due to the selection of queries—

more ambiguous entities in the long tail would require sophisticated named entity linking or disambiguation—but for the current task, IR is of secondary concern.

NER is a more critical component. A number of NER errors occur across different approaches, and these errors are difficult to resolve later in pipeline. However, the error here is acceptable, particularly as NER types allow us to greatly cut down on the search space.

Coreference resolution is a more complicated decision. Our analysis suggests that high-precision naïve tools, e.g. naïve coreference, can lead to state-of-the-art performance: they substantially cut down the search space with minimal recall loss, potentially allowing for more computationally costly downstream processes. However, using no coreference resolution heavily limits recall upper bound, and ultimately full coreference resolution is required to maximise recall.

Of particular concern is the sparsity of a given feature space, as we see from our reachability analysis over highly discriminative dependency paths. We will explore this in more detail in Chapter 6, but this implicit upper bound is an important consideration when designing a representation.

Finally, we note that the SF task is not strictly an exhaustive evaluation for each query, as the evaluation data is comprised of the time-limited human annotation plus aggregated system output only. There may be fills that are missed in the evaluation results but are correct and returned by our high recall filters, affecting our reported results. A small number of additional correct instances on low-recall filters may have a significant impact on recall and F1. We will address these annotation issues further in Chapter 7.

A small potential difference in precision does not particularly impact this analysis, because changes to recall are much more important. Nevertheless, to evaluate the size of this issue, we manually evaluate a small sample of the queries. We take the first five person and the first five organisation queries, and inspect all candidates at the COREF (ALL) filter stage for these queries (a total of 2,903 candidates).

For these candidates, there were 29 correct fills, and 21 are allowed by the COREF (ALL) filter. However, we manually identify that there are two additional correct fills found in this candidate set that are not marked as correct. One of these two missing candidates can be identified with correct coreference resolution, and the other requires complex long range inference. These additional correct fills that are will not have a large impact on the absolute precision, as there are two of 2,903 more fills. However, the relative difference in true positives, 21 of 23, creates some small uncertainty when comparing results relatively.

## 4.10 Summary

Recent TAC KBP slot filling results have shown that state-of-the-art systems are substantially limited by low recall. In this chapter, we have contributed a maximum recall analysis of slot filling, providing a comprehensive analysis of recall error created in the document retrieval and candidate generation stages.

We have contributed an analysis framework that models techniques for document retrieval and candidate generation in use across slot filling systems. The systematic recall analysis performs provides a precise reason for recall loss for every candidate in the TAC11 data, using popular approaches from TAC13 systems. Importantly, this allows us to get a measure for recall upper bounds, and shows that considered selection of pipeline components is critical for maintaining a high recall upper bound. We have found that $\sim$10% of recall is ignored by most slot filling systems due to NER error, and 8% of recall is lost when queries and fills occur in different sentences. Without coreference, a further 16% of fills are lost, but this can be reduced to 4% using efficient naïve name matching rules for proper noun coreference. We confirm that coreference and accurate NER are critical to high recall slot filling.

We find that using maximum recall bootstrapping, 39% of test slots fills are reachable from the TAC09 and TAC10 training data, limited by an upper bound on non-unique paths of 43%. This highlights that issue of the issue of feature sparsity is of key concern for slot filling, and we will look at the implications of this sparsity and methods for minimising its effect in Chapter 6. Finally, we contribute guidelines for system designers seeking to maximise recall, particularly in regards to NLP components used for candidate generation. Use of naïve coreference resolution in particular is an interesting option for system designers.

This work in maximum recall reachability gives us an interesting starting point for a full slot filling system. It presents us with a clear recall upper bound, and critically an upper bound that is reachable using a naïve baseline technique. To make use of such a configuration for a full system, a more precise approach to the propagation of labels needs to be considered. The focus of the next chapter is to take this build upon this semi-supervised setup in a full slot filling system.

# 5 Label propagation

Our reachability experiments in Chapter 4 represent the SF task and data as a graph, propagating slot labels across this graph. This technique can be used directly for extraction, if sensible constraints are applied to the propagation. In this chapter we frame slot filling as a label propagation task. We first contribute a naïve slot filling system, initially based on the reachability approach of the previous chapter. We define design criteria for the construction of the graph, determined by our analysis in previous chapters and also the experiments described in this chapter. Even with a naïve maximum recall-based label propagation approach, the results for this graph provide a reasonable baseline. We model behaviours and assumptions proposed in our design criteria directly in the graph. We apply Modified Adsorption (MAD) (Talukdar and Crammer, 2009) to the graph. Determining that lack of training data is a key problem, we contribute a comparison of the original TAC data with a large crowd-sourced release of annotated training data by Angeli et al. (2014).

In the final part of this chapter, we provide a detailed breakdown of the expected interaction between slot labels. We then contribute a modification to MAD to model these types of label interaction. We define a matrix that models the interaction between each pair of slots. We break down the label propagation into a set of binary propagations, of one positive label and one negative label for each slot. These binary propagations occur in parallel, and after each iteration the probabilities of each binary distribution are updated by aggregating across all other distributions, weighted by the interaction matrix. Using this approach, we achieve our best F1 in

this chapter of 23%. Our analysis of this system motivates our exploration of issues that still remain in the approach, which we address in more detail in Chapter 6.

## 5.1 Background and motivation

We see label propagation as a natural progression of the bootstrapping approaches that have been successful for SF, as explored in Section 3.5. Both use multiple iterations to retrieve high-confidence instances to use as further training data could be implemented using a range of learning approaches. However, bootstrapping and label propagation approaches model this iterative process directly, and are effective when starting with a small set of training seeds, as is the case in SF.

Bootstrapping approaches to RE, as discussed in Section 3.4.2, start with a small number of entity pair seeds. They find contexts shared by these seeds, use the contexts to find more entity pairs, and add the highest confidence pairs to the set of seeds, repeating as an iterative process (Agichtein and Gravano, 2000; Carlson et al., 2010). Prior work in label propagation for information extraction has primarily focused on this class-instance acquisition. Wang and Cohen (2008) expand sets of named entities (such as car manufacturers), leveraging semi-structured web content to construct a graph over which they propagate labels. Baluja et al. (2008) propagate video viewing preferences across YouTube users. Talukdar and Pereira (2010) propagate entity classes across a number of knowledge bases, experimenting with a number of algorithms. Chen et al. (2006) and Wang et al. (2011) treat relation extraction as a label propagation problem, building a graph from entity pairs and their shared contexts and propagation labels over these graphs.

Bootstrapping and label propagation approaches both make use of similarity between instances to iteratively expand a high confidence set of these instances. There a few key differences which motivate our use of label propagation. Bootstrapping is a greedy process, and is essentially a race from seeds to target instances. If

a particular target instance happens to be similar to a particular seed then it will be labelled with that seed, even if that is not the optimal labelling globally and the target instances would be more strongly labelled by a greater number of slightly more distant seeds. Using a full graph in label propagation should do better at this global labelling, by considering the whole dataset and seed data as opposed to just the portions close to particular seeds.

Importantly, a graph-based label propagation approach allows us to directly consider behaviour and assumptions we want to model. Fundamentally, it allows us to explicitly represent the underlying SF problem in the abstraction of a graph. In this chapter, this representation is our focus. Our goal is to work with the structure of the graph, using existing label propagation techniques over this underlying representation, which we want to design to be appropriately structured for the task. Defining a number of Design Criteria (based on earlier work in this thesis and experimentation with the graph) is a key contribution of this chapter.

The label propagation algorithm itself will be further detailed in Section 5.5.

## 5.2 A naïve SF system

To support a graph-based label propagation extraction process, we first implement a naïve system for slot filling, with the components configured in a pipeline as shown in Figure 5.1. We now detail those components.

### 5.2.1 Components

**Document preprocessing** We continue to use the TAC KBP Source Data (LDC, 2010) as our set of source documents. We preprocess documents by first converting documents to DOCREP (Dawborn and Curran, 2014), a lightweight document representation framework. We tokenise documents using OntoNotes-style token-

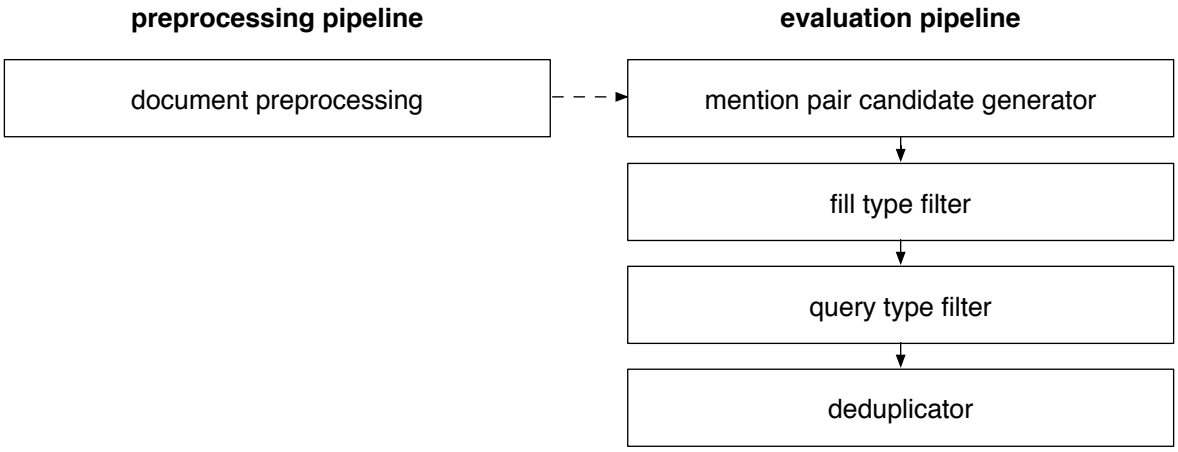



Figure 5.1: naïve slot filling pipeline.

isation[1] and sentence splitting with the Schwa tokeniser.[2] As in Chapter 4, we label NEs using Stanford NER (Finkel et al., 2005), for consistency with other slot filling systems. We then process documents with the BLLIP constituency parser (Charniak, 2000; Charniak and Johnson, 2005), which additionally provides POS tags, and convert these parses to collapsed and propagated Stanford dependencies using the Stanford parser (de Marneffe and Manning, 2008).

Using the BLLIP parser (Charniak, 2000; Charniak and Johnson, 2005) instead of the CoreNLP parser is a change from the previous chapter. We choose to switch to BLLIP as it is a state of the art parser and an accurate parser for NP internals (Kummerfeld et al., 2012). Short-range constructions such as U.S. president Barack Obama require NP internals to have a meaningful dependency path. Where CoreNLP outputs [ $\overset{\text{nn}}{\leftarrow}$ ] between the entities U.S. and Barack Obama, BLLIP (after conversion to Stanford dependencies) outputs [ $\overset{\text{nn}}{\leftarrow}$ president $\overset{\text{nn}}{\leftarrow}$ ]. As seen in the previous chapter, short-range constructions make up a large portion of fills, and so we make the decision to switch to this parser. We calculate corresponding upper bound numbers for this new configuration in Table 5.1. These upper bound numbers have dropped from the numbers in the previous chapter, in particular

[1] The key difference between OntoNotes tokenisation and the more standard Penn Treebank tokenisation is that hyphenated words are split into separate tokens (including the hyphens).

[2] github.com/schwa-lab/libschwa

| experiment | R (%) | \|search space\| |
|---|---|---|
| NAMED ENTITIES + EXACT MATCH | 67 | 956,547 |
| ...+ TYPES | 61 | 289,942 |
| ...+ SENTENCE | 40 | 20,704 |
| ...+ NON-UNIQUE | 29 | 4,377 |

Table 5.1: Results on TEST given sets of filters configurations, using BLLIP. The ellipses indicate the previous line.

NON-UNIQUE has dropped to 29% from 43%. This substantial recall upper bound loss is due to better representation of the data, and the standard preprocessing used in the previous chapter may have been overly generous for two reasons. Firstly, CoreNLP sentence splitting ignores a substantial amount of document structure. This is particularly noticeable in headlines and datelines—which are frequently merged into the first sentence of documents—and semi-structured lists (particularly where sports teams and their members are mentioned) which are merged into a single sentence. In some cases a large document may be treated entirely as a single sentence. The better sentence splitting used in this chapter better represents the data, but a substantial number of previously findable instances are now in different sentences. Secondly, changing to a more accurate parser has resulted in short range paths being now more sparse, as NP internals are better represented. The search space in this case has dropped from 7,236 to 4,377, a relative reduction of 40%, which is indicative of a more discriminative representation.

**Mention pair candidate generator** For each query, we find every NE which matches the text of the query in the corpus of documents. We then add every NE in the same sentence as the query match as a candidate fill for every slot for that query. This is equivalent to the SENTENCE filter in Chapter 4.

**Fill type filter** The fill candidates are filtered by NE type, so that the NE types are consistent with the slots, e.g. `per:city of birth` can only be filled by a LOC.

| pipeline | tp | tp + fp | R (%) | P (%) | F (%) |
|---|---|---|---|---|---|
| mention pair candidate generator | 243 | 74,218 | 45 | 0 | 1 |
| fill type filter | 225 | 21,669 | 42 | 1 | 2 |
| query type filter | 213 | 20,704 | 40 | 1 | 2 |
| deduplicator | 213 | 10,499 | 40 | 2 | 4 |
| NON-UNIQUE | 155 | 4,377 | 29 | 4 | 6 |
| reachability | 114 | 2,025 | 21 | 6 | 9 |

Table 5.2: Naïve pipeline results.

**Query type filter** We also apply the NE type filter to the query match NE, so that any fills that are extracted are consistent. This is straightforward, in the case of `per:city of birth` the query NE must be a PER.

**Deduplicator** The deduplicator filter removes duplicate fills per slot and query. This is done by case-insensitive string match.

### 5.2.2 Evaluation

In this chapter, we continue the evaluation setup of Chapter 4, in evaluating against the full TAC 2011 evaluation for NEs only. This naïve system reflects the SENTENCE filter configuration of Chapter 4—query-fill pairs must be named in the same sentence—with the addition of the query type filter, which was not previously present. Evaluation is performed at each stage of the naïve pipeline, and these results are shown in Table 5.2. These numbers follow from the results in Chapter 4: we lose a substantial amount of recall due to errors in document retrieval, NER error, NER type errors and lack of coreference resolution. This again demonstrates the substantially effect of pipeline error on recall.

The next step is to add an actual extraction component. Here, we implement a graph-based extraction approach following on from our reachability experiments in Chapter 4.

Our earlier reachability experiments use a very simple graph configuration, bootstrapping between NE pairs and dependency paths. This implicitly imposes the NON-UNIQUE constraint, that a dependency path must occur at least twice in the corpus to be present in the graph: if a dependency path is unique, it will only be connected to one NE pair, and will not be possible to find any more pairs using that path. If a NE pair only has one unique path, it will not be connected to anything else in the graph, and will be pruned for efficiency. Applying this implicit constraint to our naïve system gives a recall upper bound of 29%, with results appended to Table 5.2. Similarly, the reachability using the TAC training data has a maximum reachability of 21%. Precision for the naïve pipeline is very low, but at this stage the configuration is still maximising recall.

As we are only evaluating on NE (`name`) slots, we cannot directly compare these results with previous TAC slot filling results. We expect the results to be representative of overall performance. Some slots like `per:title` may be easier than the `name` slots, but overall we expect these results to roughly correspond to overall performance. As discussed, the top result for TAC 2011 was F1 29.5% and TAC 2013 was F1 37.3%. We expect an F1 to be 30% or higher to be likely to be competitive for state of the art.

In the next section, we make use of label propagation process to model the likelihood of slot fills, as opposed to simply maximising the recall. We still want to make use of this semi-supervised process, and we first need to modify the topology of the graph for the full task.

## 5.3 Slot filling graph topology

We begin the process of defining the graph by proposing several design criteria that guide the construction of the graph and the approach to label propagation. These criteria are derived from two sources of analysis. The first set of design

criteria are derived from higher-level aspects of the task that we have covered earlier in this thesis, as well as early experiments with the graph structure not detailed in this thesis. These criteria will primarily drive the design of our baseline configuration. The second group of criteria are developed from experimenting with the graph in this chapter. These are design decisions motivated by results from different versions of the graph.

**Design Criterion 1: The graph should be a direct representation of the underlying data.** This criterion, while somewhat abstract, is our key motivation for using a graph-based approach. We want to explicitly represent a corpus and annotation as a graph, directly representing the behaviour and the assumptions we want in the model. We then apply an algorithm over this abstraction directly. The criterion is important as we want to allow the data to be represented as directly as possible. We are trying to make as few assumptions as possible, to represent the data as closely as we can (as practically as possible).

**Design Criterion 2: The graph is based on entity pairs.** As covered at length in previous chapters, the core of most SF is RE. Extractions are made between entity mention pairs within sentences, and we follow that framing of the task. In this regard, we follow other label propagation approaches to RE, particularly the work of Wang et al. (2011), although much of the rest of our graph setup differs.

**Design Criterion 3: The construction of the graph needs to minimise labelling error.** In a semi-supervised setting, there are three main sources of labelling error. The first are incorrect gold annotations: as with other tasks, errors in training data will result in errors in evaluation. Dealing with this issue is a broader NLP problem, and we will consider the impact of this problem on slot filling later in this thesis. The second source of error is where training data is incorrectly applied to the graph as seeds, as is potentially the case with approaches like distant supervision.

Consider Example 1: the labelling (Obama, `per:employee of`, U.S.) should not be applied to this instance of this pair.

(1) *Obama* was born in the *U.S.*

This labelling may occur in the label propagation process, but the initial graph should not exacerbate this problem.

The third source of error, and the one that it likely the most significant in previous work is semantic drift. Iterating through different ambiguous contexts leads to the semantics of a label drifting from its original seed context. Previous work in semi-supervision, as presented in Section 3.4.2, has identified that minimising semantic drift is of key importance in effective use of semi-supervised approaches in general. Suppose in the above case, a correct seed label is applied to (Obama, `per:employee of`, U.S.) from Example 2.

(2) *Obama* is the *U.S.* president.

This is connected to Example 1 via the node (Obama, U.S.), and so these nodes are very close in the graph. An algorithm will need to take this issue into account, and importantly the underlying graph needs to distinguish between these cases.

This design criterion has the goal of allowing label propagation to identify the properties that informative nodes have, so that it can make use of informative nodes and disregard uninformative ones. To enable this, these nodes need to be separate in the underlying graph.

### 5.3.1 Constructing a graph

These design criteria now give us a framework in which to design a graph structure. Following Criterion 2, we start with a graph that contains entity pairs. This graph is in Figure 5.2. This is the same graph as used in the reachability bootstrapping in

the previous chapter. We begin with **entity pair nodes**, such as (Paulson, Goldman Sachs), extracted from pairs of NEs in sentences.

To actually do any semi-supervised learning, we need to connect these nodes by shared context. We choose lemmatised shortest dependency paths as this context, following Chapter 4, as they are relatively discriminative and have been effective for RE in general (Riedel et al., 2013). An edge between two nodes is defined wherever two entity pairs share this context, i.e. when both pairs are connected by the same dependency path in text. We will discuss edge weights shortly. Relevant entity pair nodes are marked as seed nodes based on annotated instances of those NEs in text, and propagation proceeds from those nodes.

In this work, we collapse NEs pairs by string match. We could use a more sophisticated cross-document coreference resolution system or named entity linking process, and the effect of these different approaches is unclear. However, we do not expect much of an effect on extracting fills from the graph at least, as queries in this evaluation are not ambiguous (as in Chapter 4 and Li et al. (2011), aliases only have a small effect on performance). Note that while an entity pair can exist disconnected from the graph if it has no shared context, these will have no influence on the graph, and we do not include such nodes.

This basic graph allows us to get a measure of reachability, but has a number of issues. In particular, there is no way to assign a label distribution to a particular path, and so there is no way to propagate a distribution that specifically applies to a path. Information about a specific path is instead distributed across edges between entity pair nodes that share that path. This makes aggregating information about each context difficult, e.g. even if a large amount of weight for `org:top employees/members` is assigned to entity pair nodes that share the context $[\ \text{ORG} \xleftarrow{\text{prep of}} \text{chairman} \xleftarrow{\text{appos}} \text{PER}\ ]$, directly propagating this weight on to the dependency is problematic, as it is not aggregated but is divided across many pairs of nodes.

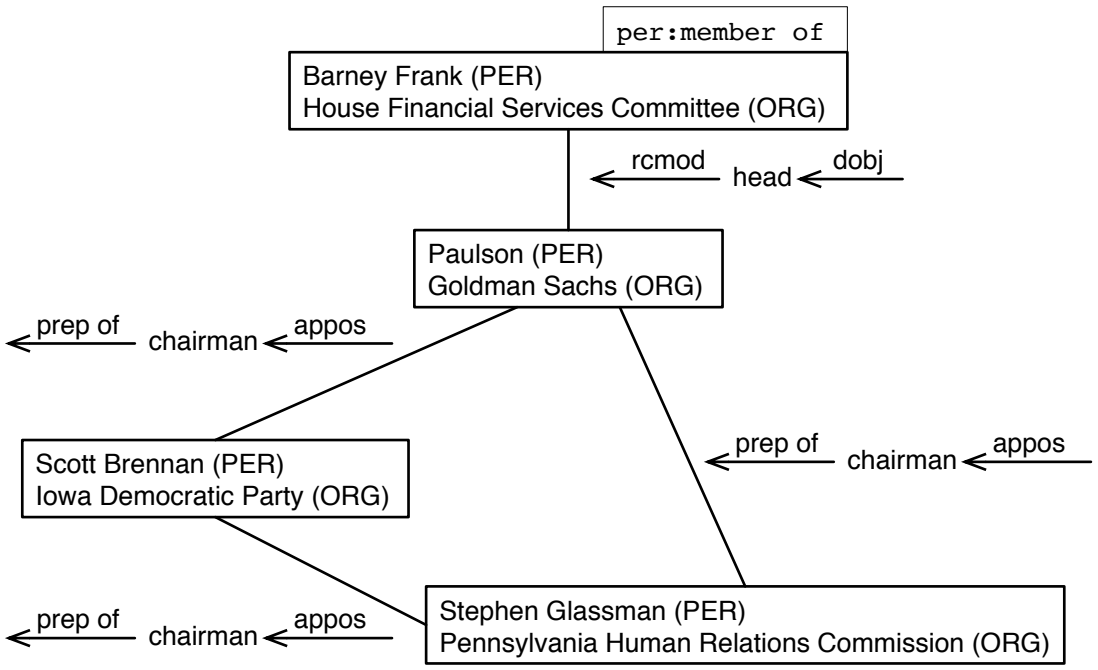



Figure 5.2: Graph with entity pairs. `per:member of` indicates a seed node.

To account for these issues, we add **feature nodes** to the graph, as shown in Figure 5.3. These nodes realise these dependency path contexts in the graph, and have edges to all entity pairs that share that context (the edges that were previously between these entity nodes are removed).

Following Criterion 3, to minimise semantic drift of slots across different (potentially erroneous) entity types, all nodes are typed by their corresponding NER types. For example, all (PER, PER) entity pair nodes and feature nodes (and only those nodes) are in the same graph. These types of pairs remain ordered, that is (PER, ORG) and (ORG, PER) are different graphs.

This graph as shown in Figure 5.3 is a more explicit representation, but still has problems in regards to Criteria 1 and 3: entity pairs are entirely collapsed together without regard for their mentions in text. We used this configuration in early experiments, but found that many seed labellings were problematic, as different relations are collapsed together (as discussed in Criterion 3). SF also requires a contextual justification for a slot fill. We need to know which particular instance of

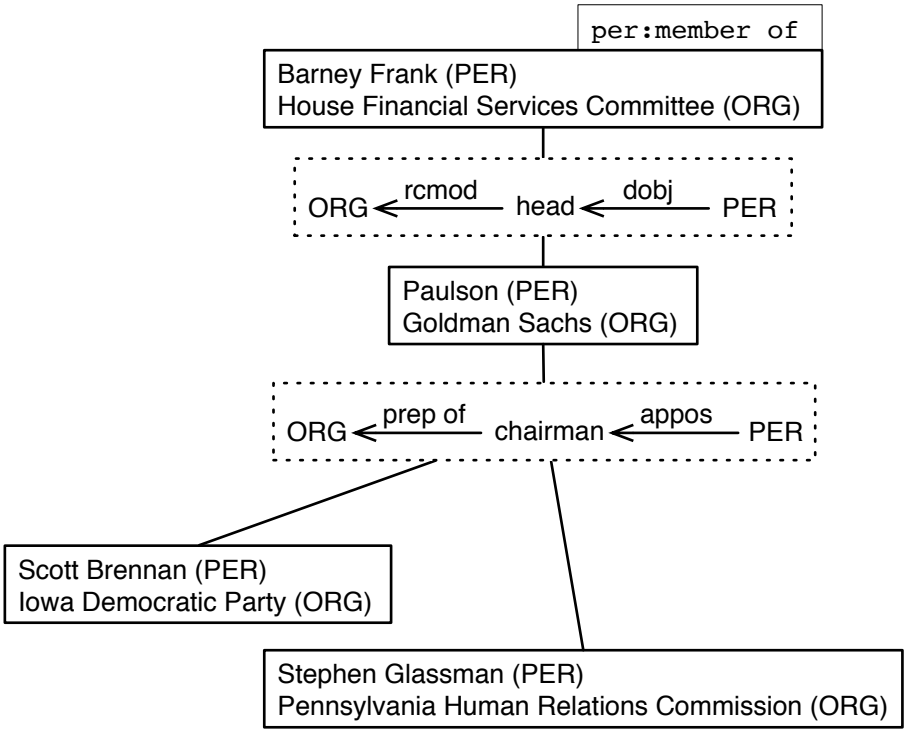



Figure 5.3: Graph with entity pair nodes and feature nodes (dotted node borders). `per:member of` indicates a seed node.

an entity pair is the one that (best) expresses a slot fill. Finally, collapsing these together immediately increases semantic drift for contexts as pairs in semantically different contexts are collapsed together.

To address these issues, we add **mention pair nodes** to the graph, as shown in Figure 5.4. These nodes, placed between every entity pair and feature node (with edges updated accordingly), capture the actual grounded mentions in text, in which the entity pair occurs with the the feature. For example, the top-most mention pair on the right of Figure 5.4 is the context in which Barney Frank heads House Financial Services Committee. This results in each entity pair node being connected to one or more mention pair nodes; each feature node being connected to one or more mention nodes; but each mention node only being connected to one entity pair node and one feature node (as long as there is only one type of feature). Despite not connecting any additional nodes in this setup (the original entity pair

node and feature node are still connected, just via the intervening mention pair nodes), we can now directly label these mention pair nodes as seeds, and retrieve final distributions directly from these nodes.

Here these mention nodes are in-sentence named mentions that match the entity pair names. However, these could also be other named, nominal or pronominal mentions identified by more sophisticated named entity linking or coreference resolution. Note that we only include sentence context in Figure 5.4, but these nodes represent the full context of the node: if we wanted to include document-level features for example, these features would be included as feature nodes and connected to the corresponding mention nodes. It is possible that we could remove the entity pair nodes entity, but without those nodes we would only be able to propagate slots across feature node. In this setup where we only have a single dependency path feature, we could only propagate slots between mention pairs which had the same path, and this is of limited use.

Most importantly for slot filling and Criterion 3, we can now directly use annotated mention instances as seed nodes, removing ambiguity created when collapsing mentions together, and we can now identify specific instances which express a slot fill to use as justification. We use this graph configuration going forward. This configuration is more closely aligned with Criterion 1 than the other previous graphs, as we represent both entity and mention levels, effectively representing KB and sentence-level entities in the graph. We note that for Criterion 3, there are still issues of semantic drift—we have only added a single extra intervening node between contexts—but this at least allows an algorithm to potentially make use of the distinction between contexts, whereas before this was impossible because there was no distinction in the graph. Now that we have defined nodes, we can turn our attention to edge weights.

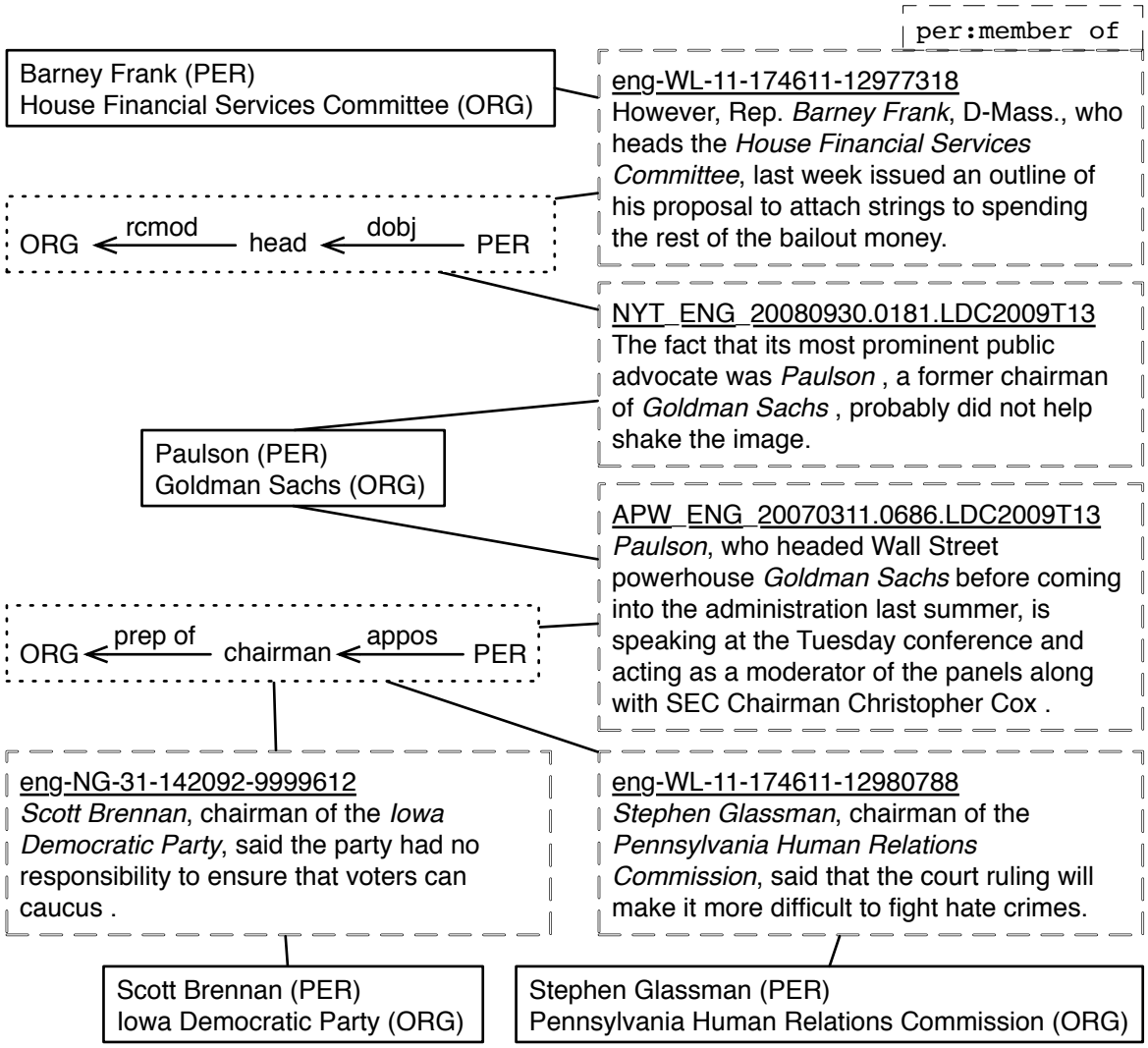



Figure 5.4: Graph with entity pair nodes, mention pair nodes (dashed node borders), and feature nodes (dotted node borders). `per:member of` indicates a seed node.

### 5.3.2 Edges

Edges exist in the graph to represent co-occurrence. Mention pairs are connected to one entity pair, entity pairs are connected to the mention pairs they represent. Feature nodes are connected to the mention pair nodes that they occur with and to other feature nodes they occur with (that is, if two features occur on the same mention, there is an edge between them). There are no edges between entity pairs and features.

Assigning edge weights is potentially a complicated task. In this work, we choose to use a simple model of edge weights. This is in line with other label propagation work (Talukdar and Crammer, 2009; Wang et al., 2011), where focus is on the propagation algorithm itself, improving performance by modifications to the algorithm, as opposed to edge weights. We weight edges by normalised co-occurrence counts. Edge weights are a function of co-occurrence, and a high edge weight ties nodes together, and increases continuity between nodes. We expect that two nodes are expected to express similar labellings when they mutually co-occur: if a mention pair occurs with a dependency path many times, we expect that the label of that dependency path is likely the label of the mention pair, and so the corresponding edge between the mention pair node and the feature node should have a high weight. Similarly, such an edge should have a higher weight than a low-frequency co-occurrence.

Firstly, each edge is assigned the raw co-occurrence count of its two endpoints. For example, an edge between an entity pair node and a mention pair has a count of one, as the mention pair refers to one entity pair. All edges are then normalised per node. As this normalisation requires each edge to have two different weights, the graph becomes a directed graph with all outgoing edges using a node's own normalised weights, and all incoming edges using the normalised weights of the neighbour. This normalisation is required for the Modified Adsorption (MAD)

| counts | baseline |
|---|---|
| entity pairs | 5,423,550 |
| mention pairs | 19,100,193 |
| features | 3,023,841 |
| total nodes | 27,547,584 |
| edges | 78,143,816 |
| seeds | 658 |

Table 5.3: Graph profile for baseline graph.

algorithm, to be detailed later. Exploring different edge weights, particularly in learning edge weight from the data, is a substantial area for future work.

For most of our experiments and analysis we keep pipeline components consistent (including the graph), as making changes to the pipeline complicates analysis and we ideally want to evaluate the label propagation component isolated from the other stages. We will later add pipeline components when we find they substantially improve performance, or where we have made incorrect assumptions in our baseline setup. Number of nodes and edges for this naïve graph, constructed from the preprocessed source documents, are given in Table 5.3.

### 5.3.3 Seeds

We use TAC training annotations and results for TAC 2009 and 2010 as seeds in our graph. We note that for the purpose of these initial experiments, derived from reachability, we only make use of positive seeds. Negative seeds are problematic in our basic setup. Standard label propagation approaches model the competition between positive labels, and do not model negative labels. It is not useful to simply add negative versions of every slot: propagating a ¬`per:employee of` label is useful for identifying examples of `per:employee of` itself, but is not particularly relevant for other slots. That a person is not an employee of a country doesn't indicate anything about whether they were born there (`per:country of birth`):

a negative label being the most probable label is not meaningful. A general `no relation` label would be useful for labelling nodes which fall entirely outside the schema, but we have no real way of constructing such a label given the TAC training data. An important hypothesis of this thesis is that maintaining high recall is more important than improving precision, and so this lack of negative data is not important for these initial experiments.

Using TAC 2009 and 2010 correct annotations and results only, we align the annotations in Table 5.4 to nodes in the graph. The number of aligned annotations is relatively small: we note that 27% of annotations are alignable, but the SENTENCE filter from Chapter 4 would suggest this should be roughly 40%. We identify a number of reasons for this recall loss.

Firstly, the NEs produced by our automatic NER are not entirely consistent with NE spans in this training data. NEs in the annotation data are labelled by human annotation, not Stanford NER. Additionally, Stanford NER is less often used by the 2009 and 2010 TAC systems. Hence, NEs spans in this disagree with our Stanford NEs more frequently than in the TAC 2011 results data. These produce different NE spans which we can not align to our NEs. To get a indication of the impact of this issue, we consider the training sentences in which we identify at least two NEs. In these, we can find both the query and fill term by string match in 1,055 instances. This is substantially more than the 658 of these instances we can align to NEs, and is 43% of the annotations. This is more in line with our expected results, suggesting that differences in NER here are more substantial than in later data where CoreNLP is more popular.

Secondly, we do not use conference resolution in this setup, and we cannot align NEs that do not occur in the same sentence. This is the primary reason that we expect alignment to be around 40%. There are additionally a notable number of instances that occur outside same-sentence mentions, particularly lists, which substantially prevent alignment for some slots. For `org:members`, we cannot align

any of the International Monetary Fund members in Example 3, as there is no mention of the International Monetary Fund. We confirm that this is not an NE issue: only two additional instances for `org:members` can be found when evaluate on falling back to string match above.

(3) The reallocation of voting power, which is supposed to occur every five years, extends more weight to countries experiencing strong economic growth, such as China, India, Brazil, South Korea and Mexico.
The main losers in the reshuffling are Britain, followed by France, Saudi Arabia, Canada and Russia.

Loss when converting alignable labels to seed labels are due to losses in the graph construction process, including nodes which are pruned due to being unconnected to other nodes (the implicit NON-UNIQUE filter). Two slots have no alignable labels, and an extra four slots have no applicable seeds. We will address this lack of data in Section 5.6.1. The reachability results in Table 5.2 are derived from this seed data, and hence give us our baseline: a recall upper bound of 21%, and an F1 at this upper bound of 9%. We note that the TAC 2011 median result was 13% and the TAC 2013 median result was 16%. While these results are not directly comparable, this does give us some indication that this upper bound F1 score, while low, is not that far below the performance of full TAC systems.

Having a baseline graph in place with an recall upper bound, we can now turn our attention to increasing precision. Our naïve reachability approach does not allow us to actually define label boundaries in the graph. To do this, we need to use a more sophisticated label propagation algorithm.

| slot | count | alignable | seeds |
|---|---|---|---|
| org:top members/employees | 626 | 143 | 101 |
| per:employee of | 227 | 77 | 48 |
| per:member of | 168 | 44 | 29 |
| org:subsidiaries | 127 | 14 | 10 |
| per:origin | 108 | 8 | 0 |
| org:city of headquarters | 105 | 42 | 20 |
| per:cities of residence | 97 | 23 | 15 |
| per:parents | 87 | 21 | 6 |
| org:country of headquarters | 87 | 17 | 13 |
| org:members | 86 | 3 | 1 |
| org:stateorprovince of headquarters | 84 | 6 | 4 |
| per:siblings | 81 | 19 | 10 |
| org:founded by | 63 | 16 | 12 |
| org:parents | 58 | 13 | 6 |
| org:political/religious affiliation | 58 | 2 | 0 |
| per:stateorprovinces of residence | 57 | 12 | 6 |
| per:spouse | 56 | 28 | 11 |
| per:other family | 49 | 10 | 3 |
| per:children | 42 | 11 | 7 |
| per:countries of residence | 41 | 17 | 6 |
| per:schools attended | 37 | 3 | 1 |
| org:member of | 27 | 8 | 2 |
| per:country of birth | 23 | 7 | 1 |
| per:city of birth | 22 | 4 | 3 |
| org:shareholders | 22 | 0 | 0 |
| per:stateorprovince of birth | 15 | 3 | 0 |
| per:city of death | 3 | 1 | 0 |
| per:stateorprovince of death | 2 | 1 | 1 |
| per:country of death | 1 | 0 | 0 |

Table 5.4: Counts of TAC 2009 and 2010 NE annotations and results, count of annotations *alignable* using our pipeline, and number of annotations present as *seeds* in the graph after filtering constraints.

### 5.3.4 Comparison with existing work

The work of Wang et al. (2011, 2012) also explores label propagation. As they also use the basic Modified Adsorption label propagation algorithm detailed in Section 5.5, the primary differences relate to graph design. Otherwise, there are a number of differences in algorithmic extensions to Modified Adsorption between approaches, and these are discussed in Section 5.7.1. In Wang et al. (2011), the set of relations (four basic relations and nine temporal relations) and datasets are focused on sports (133,000 documents) and celebrity (88,000 documents) data. The differences in the graph design are as follows:

- Node types. Wang et al. (2011) use two types of nodes: entity pair nodes, and generalised mention pairs nodes. These generalised nodes generalise a mention by representing it as a set of $n$-grams.
- Edge types. In Wang et al. (2011), edges between entity pair nodes and mention nodes are based on co-occurrence, as in our work. However, edges between generalised mention pair nodes are based on the overlap of $n$-grams that make up those nodes.
- Features. As above, features are derived from $n$-grams based on patterns, rather than dependency paths.

## 5.4 Pipeline filtering

On initial inspection of the highest degree nodes in the graph, we identify a number of nodes that are likely uninformative and likely to cause invalid labels to propagate through the graph. The top ten highest degree nodes in the graph are listed in Table 5.5. We note that several nodes appear to be particularly erroneous, rather than just nodes that we don't expect to be useful for slot filling. These are often

| # | path | example |
|---|---|---|
| 1 | [ PER $\xrightarrow{\text{conj and}}$ PER ] | So, South Carolina on Tuesday is huge for [McCain]$_{\text{PER}}$ and [Thompson]$_{\text{PER}}$. |
| 2 | [ PER $\xleftarrow{\text{conj and}}$ PER ] | |
| 3 | [ PER $\xrightarrow{\text{appos}}$ LOC ] | Prof. [Ee-Peng Lim]$_{\text{PER}}$, School of Information Systems, Singapore Management University, [Singapore]$_{\text{LOC}}$ 12. |
| 4 | [ PER $\xrightarrow{\text{appos}}$ PER ] | Among those attending were [Eva Longoria Parker]$_{\text{PER}}$, Ashton Kutcher, Christian Slater, [Natalie Portman]$_{\text{PER}}$, Sting, Mariska Hargitay, Steven Spielberg and Jon Bon Jovi. |
| 5 | [ PER $\xleftarrow{\text{appos}}$ PER ] | |
| 6 | [ ORG $\xrightarrow{\text{conj and}}$ ORG ] | We're searching the web and key sites like [LinkedIn]$_{\text{ORG}}$, Xing, ZoomInfo, blogs, [Jobster]$_{\text{ORG}}$, AOL and others to find free resumes and contacts. |
| 7 | [ ORG $\xleftarrow{\text{conj and}}$ ORG ] | |
| 8 | [ ORG $\xleftarrow{\text{dep}}$ LOC ] | Ironically, Ging was Gaza director of the [United Nations Relief and Works Agency]$_{\text{ORG}}$ ([UNWRA]$_{\text{ORG}}$), the largest nongovernmental employer in Gaza. |
| 9 | [ PER $\xrightarrow{\text{prep of}}$ LOC ] | thePhantomWriters.com and Article-Distribution.com are owned and operated by [Bill Platt]$_{\text{PER}}$ of [Stillwater]$_{\text{LOC}}$, Oklahoma USA. |
| 10 | [ PER $\xrightarrow{\text{poss}}$ LOC ] | And [Canada]$_{\text{LOC}}$'s "Hitman" [David Foster]$_{\text{PER}}$, played the keys on Thriller. |

Table 5.5: Highest degree nodes in the basic graph. #3, #4 and #8 are indicative of repeated parser errors.

cases where the context of entity pairs is not useful for determining the relation between the entities, and is often the case in lists, such as in [ $\xleftarrow{\text{appos}}$ ] (#4–8) in Table 5.5. [ $\xleftarrow{\text{appos}}$ ] cases (in all directions) are typically indicative of parser error in parsing lists. Some cases are not errors, but are still not useful in our graph. In the case of [ PER $\xrightarrow{\text{conj and}}$ PER ] (#1), the path doesn't provide any information about the relation between the entities beyond the fact that they co-occur in some way. While this is a small set of examples, it is indicative of the substantial number of

errors present due to pipeline error: only [ PER $\xrightarrow{\text{prep of}}$ LOC ] (#9) and [ PER $\xrightarrow{\text{poss}}$ LOC ] (#10) are valid and useful. This leads us to define our next design criterion.

**Design Criterion 4: The graph should minimise the impact of pipeline error.** Starting with these examples and working through the top 1000 highest degree nodes in detail, as well as a brief inspection over the whole dataset, we identify a number of categories of features that we want to filter as they appear to be created by pipeline error, or are uninformative. Some pipeline errors are particularly common. Parse errors in lists are particular problematic: as they are generated from every pair of entities in every list, they create a huge number of erroneous edges in the graph and erroneous connect a huge number of nodes together, regardless of whether these errors occur with training or test data. A rule-based filter is an straightforward approach for handling these most common pipeline errors. These rules are as follows.

**Arc filters** We ignore single arc paths with the label `nn` (noun compound modifier, Example 4); `appos` (appositional modifier, Example 5: this example is a parse error); and `dep` (dependent, Example 6).

(4) **path:** [ Gucci $\xrightarrow{\text{nn}}$ Adam Senn ]
**sentence:** According to a report, the "Mean Girls" actresses' latest boy toy is [Gucci] model [Adam Senn], a partial owner of Chelsea eatery II Bastardo, who was in MTV's "The City."

(5) **path:** [ AFA $\xrightarrow{\text{appos}}$ Donald WIldmon ]
**sentence:** Labels: [AFA], bigotry, [Donald WIldmon], karma, religion New!

(6) **path:** [ Ontario Human Rights Commission $\xleftarrow{\text{dep}}$ Barbara Hall ]
**sentence:** Enter [Ontario Human Rights Commission] chief [Barbara Hall] –

best known to Canada for a creepy 2008 manifesto, urging government to give human-rights mandarins the power to censor media publications they don't like.

Applying a seed to a corresponding mention for these paths would induce error in labelling. Note that there are two different reasons for ignoring these paths. The first are paths which are parse errors, such as in Example 4 where the path should include model. All of the above examples are errors. However, these short paths are not useful even when correct, and do not express the NE slot fills we are modelling (although they may express a much more general relation). Example 7 is such a correct [$\xrightarrow{\text{appos}}$] example:

(7) **path:** [ United Nations Relief and Works Agency $\xleftarrow{\text{appos}}$ UNWRA ]
**sentence:** Ironically, Ging was Gaza director of the [United Nations Relief and Works Agency] ([UNWRA]), the largest nongovernmental employer in Gaza.

There are no slots that we are considering that can be expressed by apposition of entities (particularly as we are not considering `alternate names`), and so we also choose to exclude these type of paths, particularly as they are conflated with the large error cases.

**Path arc filters** We ignore paths containing a `dep` arc, such as Example 8, as the `dep` label is indicative of parser uncertainty or of complex constructions not representable using Stanford dependencies.

(8) **path:** [ Allie $\xleftarrow{\text{appos}}$ cousin $\xrightarrow{\text{conj and}}$ son $\xleftarrow{\text{dep}}$ Sean Preston ]
**sentence:** Britney Spears was spotted out the other day with her son [Sean Preston] and her cousin, [Allie].

**Symmetric path filter** We ignore symmetric paths of length 2, such as [ $\xrightarrow{\text{nsubj}}$ contribute $\xleftarrow{\text{nsubj}}$ ]. When using collapsed and propagated dependencies, these symmet-

ric paths are almost always created by a single `conj` arc being collapsed. As with single `conj` arcs, these do not provide useful context. In Example 9, the path only provides context that both entities are both saying something, and not anything useful for extracting `per:spouse`.

(9) **path:** [ Jason Mesnick $\xleftarrow{\text{nsubj}}$ say $\xrightarrow{\text{nsubj}}$ Molly Malaney ]
**sentence:** According to People Magazine, Bachelor couple, [Jason Mesnick] and [Molly Malaney] officially said "I do" on Saturday in California.

**Path length filter** We use only dependency paths of a limited length, as more complex structures tend to include more errors, particularly over long sentences. We use lengths less than four as a baseline. As seen in Section 4.8, this allows us to filter these erroneous paths with a minimal drop in recall. Additionally, we discard paths which pass through an NNP or NE that is not the query or candidate fill, as we are interested in extracting relations directly between pairs of entities rather than relations mediated by a third entity. This filter rejects paths such as in Example 10.

(10) **path:** [ Holly Montag $\xleftarrow{\text{poss}}$ bash $\xleftarrow{\text{prep following}}$ arrest $\xleftarrow{\text{prep from}}$ stem $\xleftarrow{\text{vmod}}$ attorney $\xleftarrow{\text{prep with}}$ reach $\xrightarrow{\text{nsubj}}$ Stephani Pratt ]
**sentence:** [Stephani Pratt] reached a deal with the district attorney stemming from her Oct. 18 DUI arrest following sister-in-law [Holly Montag]'s birthday bash at a Hollywood nightclub.

We provide graph statistics for the filtered version of the graph in Table 5.6. We note that this has substantially reduced the size of the graph, with the filtered version having 49% of the nodes and 48% of the edges. We recalculate upper bounds for this new graph, and these results are in Table 5.7. While this filtering

| counts | baseline | filtered |
|---|---|---|
| entity pairs | 5,423,550 | 3,275,446 |
| mention pairs | 19,100,193 | 9,249,592 |
| features | 3,023,841 | 1,007,978 |
| total nodes | 27,547,584 | 13,533,016 |
| edges | 78,143,816 | 37,780,554 |
| seeds | 658 | 328 |

Table 5.6: Graph profiles for graph in this chapter.

| | pipeline | tp | tp + fp | R (%) | P (%) | F (%) |
|---|---|---|---|---|---|---|
| naïve | NON-UNIQUE | 155 | 4,377 | 29 | 4 | 6 |
| | reachability | 114 | 2,025 | 21 | 6 | 9 |
| filtered | NON-UNIQUE | 152 | 3,328 | 28 | 5 | 8 |
| | reachability | 104 | 1,528 | 19 | 7 | 10 |

Table 5.7: Filtered graph pipeline results, along with original pipeline.

has slightly reduced our recall, this is for cases with substantial pipeline error that we would not expect to be able to extract anyway.

We inspect the new highest degree nodes in Table 5.8, and note that the large error cases have been removed, with the new top features being actually meaningful for propagating labels. Even those most are still short dependency paths with minimal context, most of these paths—like [ ORG $\xrightarrow{\text{prep in}}$ LOC ] (#3), an ORG in a LOC—are meaningful enough that they could almost directly map to slots. Some of these, such as [ PER $\xrightarrow{\text{dobj}}$ tell $\xleftarrow{\text{nsubj}}$ ORG ] (#6) and [ ORG $\xleftarrow{\text{dobj}}$ tell $\xrightarrow{\text{nsubj}}$ PER ] (#7), may be less useful for defined slots, but we expect them to be less of a problem for propagation—there is a more meaningful relationship between then entities than simply co-occurring, and they are not fundamentally erroneous.

We note that number of seeds is low, and we will address this issue later in this chapter. We will first focus on actually performing meaningful label propagation, as opposed to simply using upper bound reachability.

| # | path | example |
|---|---|---|
| 1 | [ PER $\xrightarrow{\text{prep of}}$ LOC ] | thePhantomWriters.com and Article-Distribution.com are owned and operated by [Bill Platt]$_{\text{PER}}$ of [Stillwater]$_{\text{LOC}}$, Oklahoma USA. |
| 2 | [ PER $\xrightarrow{\text{poss}}$ LOC ] | And [Canada]$_{\text{LOC}}$'s "Hitman" [David Foster]$_{\text{PER}}$, played the keys on Thriller. |
| 3 | [ ORG $\xrightarrow{\text{prep in}}$ LOC ] | "The gross margin (at RIM) is a nightmare," Peter Misek, an analyst with [Canaccord Adams Inc.]$_{\text{ORG}}$ in [Toronto]$_{\text{LOC}}$, said in an interview with Bloomberg Radio. |
| 4 | [ ORG $\xrightarrow{\text{poss}}$ LOC ] | Last week, [London]$_{\text{LOC}}$'s [Daily Telegraph]$_{\text{ORG}}$ published excerpts of what the newspaper said were questions from the committee about China's handling of the Tibetan protests. |
| 5 | [ PER $\xrightarrow{\text{prep in}}$ LOC ] | Money-market funds are pulling back from investing in unsecured commercial paper from banks, JPMorgan Chase & Co. analysts led by [Alex Roever]$_{\text{PER}}$ in [New York]$_{\text{LOC}}$ wrote in a report dated Oct. 3. |
| 6 | [ PER $\xrightarrow{\text{dobj}}$ tell $\xleftarrow{\text{nsubj}}$ ORG ] | "He was an early warner about Fannie and Freddie," [Robert Litan]$_{\text{PER}}$, vice president for research and policy at the Kauffman Foundation, an organization dedicated to the promotion of entrepreneurship, told [The Times]$_{\text{ORG}}$. |
| 7 | [ ORG $\xleftarrow{\text{dobj}}$ tell $\xrightarrow{\text{nsubj}}$ PER ] | |
| 8 | [ PER $\xrightarrow{\text{prep of}}$ ORG ] | "Wednesday is a safety stand down day," said Sgt. [Thomas Sost]$_{\text{PER}}$, of [New Jersey State Police Marine Services Bureau]$_{\text{ORG}}$. |
| 9 | [ ORG $\xrightarrow{\text{prep of}}$ PER ] | |
| 10 | [ PER $\xrightarrow{\text{nn}}$ spokesman $\xrightarrow{\text{nn}}$ ORG ] | [AstraZeneca]$_{\text{ORG}}$ spokesman [Jim Minnick]$_{\text{PER}}$ said Tuesday the company could n't comment about Arkansas ' lawsuit because officials had not been notified of its filing or had a chance to review it . |

Table 5.8: Highest degree nodes in the filtered graph.

## 5.5 Modified Absorption

We now build upon our naïve semi-supervised approach, moving beyond our basic reachability to a more sophisticated label propagation (Zhu and Ghahramani, 2002) approach. Chen et al. (2006) make use of a baseline label propagation technique for RE, but we follow the work of Wang et al. (2011, 2012) in leveraging Modified Adsorption (MAD) algorithm (Talukdar et al., 2008; Talukdar and Crammer, 2009). MAD extends basic label propagation by taking into account the impact of very frequent uninformative features and potentially incorrect seed labels. MAD has been shown to out-perform other label propagation approaches on other information extraction tasks (Talukdar and Pereira, 2010).

We expect that these aspects of MAD will be well-suited to SF. In general, label propagation approaches produce a labelling which smooths over nodes which are similar in the graph. In our setting, we primarily make use of this effect in a fashion similar to bootstrapping. High-scored edges are those between mention pairs and features which frequently co-occur. We expect such pairs and features to frequently occur in similar semantic contexts, and hence express similar slots. This extends in a iterative fashion: if an entity pair frequently occurs strongly with a number of contexts, we expect these to be semantically similar contexts with similar slot labelling. This makes label propagation a good fit of the task.

Of course, this semantic similarity does not always exist. An entity pair such as (Barack Obama, USA) is mentioned in a large number of contexts which may indicate one (or more) of many slots or none at all. Additionally, there are many frequently occurring contexts that are not actually semantically meaningful for SF. Consider [ PER $\xrightarrow{\text{dobj}}$ tell $\xleftarrow{\text{nsubj}}$ ORG ] from Table 5.8. This is one of the highest degree nodes in the graph, but is not likely to be informative for our task. MAD uses a measure of entropy to downweight the influence of these less informative high-degree nodes. This entropy is a calculated directly from the distribution of the

nodes in the graph (see the following section), and is measured by a *dummy label*, an extra label that represents how unreliable a node is. Nodes that are attached to many different contexts are likely too general as features and not informative for label propagation. This label exists in the label distribution with the other labels, and the higher the probability of the dummy label, the less reliable the node. Some of these unreliable nodes may still end up having a large influence over the graph, particularly if attached to many seeds, but one of the key defined goals of MAD is to minimise this effect.

Additionally, unlike some other label propagation approaches, MAD allows seed nodes to be relabelled. We expect this to be useful in a task where annotation errors are an issue, particularly in a graph-based setting where an incorrect node can have a large influence over the graph. This is relevant when making use of distant supervision, as we will in Section 5.6.1: we want to correct for potentially mislabelled seeds.

We now describe MAD, following the work in Talukdar and Crammer (2009). A full derivation and proofs are available in that work, here we reiterate the aspects relevant to this work. We note the bootstrap reachability experiments in Chapter 4 indicate that a graph built on dependency paths will be sparse, and this will limit recall in these initial experiments in this chapter. We will address relevant issues for the MAD algorithm in Chapter 6.

### 5.5.1 Algorithm

We start with an undirected graph $G = (V, E, W)$, where $v \in V$ is a node (in our case, one of the three types of nodes we have defined), and an edge $e = (a, b) \in V \times V$ indicates the label of nodes $a, b \in V$ should be similar, with the strength of that similarity indicated by weight $W_{ab} \in \mathbb{R}_+$ (the normalised co-occurrence count between nodes in our case). Labels are given by $L = 1, \ldots, m$ where $m$ is the number of labels. For SF, this is a simple mapping of slots to integers. Each $v$ is

assigned a prior labelling $\mathbf{Y}_v \in \mathbb{R}_+^m$ (the weights of the seed labels for each node). In our setting, a seed label gives a weight of $1$, else the weight is $0$. The output for each node is similar vector $\hat{\mathbf{Y}}_v \in \mathbb{R}_+^m$. We follow Talukdar and Crammer (2009) in describing edges weights $W$ as a matrix of transition probabilities $P$, where edge weights are normalised for each node. Formally:

$$P(v'|v) = \begin{cases} \dfrac{W_{v'v}}{\sum\limits_{u:(u,v)\in E} W_{uv}} & \text{if } (v',v) \in E \\ 0 & \text{if } (v',v) \notin E \end{cases} \tag{5.1}$$

We use the standard random walk conceptualisation of MAD, where the final labelling of a given node is the convergence of the endpoints of all random walks from that node through the graph. Note that this is opposite (but still equivalent) from how we have discussed bootstrapping and label propagation until now. Previously, we have discussed the process as starting from seed nodes, then propagating out to target nodes. MAD is typically instead discussed as the convergence of random walks from target nodes to seed nodes. These two conceptualisations are equivalent (Talukdar and Crammer, 2009), and the input and output of the graph can still be treated and analysed in the same way before. The random walk framing is primarily for the purposes of describing the algorithm.

Every random walk has three possible actions for every node. These are:

- *inject*: the walk stops and returns $\mathbf{Y}_v$ for the current node;
- *continue*: the walk continues to a neighbour with probability relative to $W_{v'v}$;
- *abandon*: the walk is abandoned and an all-zeros vector is returned.

For each of these actions, each $v$ is assigned the corresponding probabilities $p_v^{\text{inj}}$, $p_v^{\text{cont}}$ and $p_v^{\text{abnd}}$. For each $v$, $p_v^{\text{inj}}, p_v^{\text{cont}}, p_v^{\text{abnd}} \geq 0$ and $p_v^{\text{inj}} + p_v^{\text{cont}} + p_v^{\text{abnd}} = 1$. These probabilities are derived from the entropy of the transition probabilities (the normalised outgoing edge weights) for each node:

$$H[v] = -\sum_{u} P(u|v) \log P(u|v) \tag{5.2}$$

In Talukdar and Crammer (2009) this entropy is then passed through the following monotonically decreasing function:

$$f(H[v]) = \frac{\log 2}{\log 2 + e^{H[v]}} \tag{5.3}$$

However, calculating $e^{H[v]}$ is problematic in a practical setting, as $H[v]$ can be very large. Our implementation (as well as the code provided by Talukdar and Crammer (2009)) instead passes the entropy through the following approximation:

$$f(H[v]) = \frac{\log 2}{\log 2 + H[v]} \tag{5.4}$$

Next, the components of the probabilities are defined:

$$c_v = f(H[v]) \tag{5.5}$$

$$d_v = \begin{cases} (1 - c_v) \times \sqrt{H[v]} & \text{if node } v \text{ is seed node} \\ 0 & \text{if node } v \text{ is not seed node} \end{cases} \tag{5.6}$$

$$z_v = max(c_v + d_v, 1) \tag{5.7}$$

Equation 5.6 ensures that only nodes with seed labels can *inject*. Equation 5.7 ensures proper normalisation of the probabilities, and also allows for $p_v^{\text{abnd}} > 0$ when $c_v + d_v < 1$ (as in the below equation for $p_v^{\text{abnd}}$). The probabilities are:

$$p_v^{\text{cont}} = \frac{c_v}{z_v} \tag{5.8}$$

$$p_v^{\text{inj}} = \frac{d_v}{z_v} \tag{5.9}$$

$$p_v^{\text{abnd}} = 1 - p_v^{\text{cont}} - p_v^{\text{inj}} \tag{5.10}$$

Finally, for the purposes of the dummy label which represents trustworthiness in a node, $r \in \mathbb{R}_+^{m+1}$ where $r_l = 0$ for all $l \in L$, and $r_v = 1$ for the dummy label $v$.

**Input:** graph $G = (V, E, W)$

prior labelling $\mathbf{Y}_v \in \mathbb{R}^{m+1}$ for $v \in V$

probabilities $p_v^{\text{inj}}, p_v^{\text{cont}}, p_v^{\text{abnd}}$ for $v \in V$

**Output:** $\hat{\mathbf{Y}}_\mathbf{v} \leftarrow$ for $v \in V$

1: $\hat{\mathbf{Y}}_\mathbf{v} \leftarrow \mathbf{Y}_\mathbf{v}$

2: $\mathbf{M_{vv}} \leftarrow \mu_1 \times p_v^{\text{inj}} + \mu_2 \sum_{u \neq v} (p_v^{\text{cont}} \mathbf{W_{vu}} + p_u^{cont} \mathbf{W_{uv}}) + \mu_3$

3: **repeat**

4: $\quad D_v \leftarrow \sum_u (p_v^{\text{cont}} \mathbf{W_{vu}} + p_u^{cont} \mathbf{W_{uv}}) \hat{\mathbf{Y}}_\mathbf{u}$

5: $\quad$ **for all** $v \in V$ **do**

6: $\quad\quad \hat{\mathbf{Y}}_\mathbf{u} \leftarrow \frac{1}{M_{vv}} (\mu_1 \times p_v^{\text{inj}} \times \mathbf{Y_v} + \mu_2 \times D_v + \mu_3 \times p_v^{\text{abnd}} \times \mathbf{r})$

7: $\quad$ **end for**

8: **until** convergence

**Algorithm 1:** Modified Adsorption algorithm

The full MAD algorithm in presented in Algorithm 1. Line 1 applies the seeds to the graph, and line 2 precomputes normalisation term $M_{vv}$ as an efficient way to obtain label scores. Each iteration of the propagation occurs in lines 3–8. Line 4 calculates, for every node, the sum of the label weights of every neighbouring node. This is used when to the label distribution of every node in lines 5–7. Line 6 is this update, which sets the new label distribution of a node as the sum of its seed labels (weighted by $p_v^{\text{inj}}$), its neighbouring labels (weighted by $p_v^{\text{conj}}$, as in line 4), and its dummy label (weighted by $p_v^{\text{abnd}}$, $\mathbf{r}$ is a zero vector except for $\mathbf{r}_v = 1$ representing the extra dummy label). $\mu_1$, $\mu_2$ and $\mu_3$ are hyperparameters which control the weighting of $p_v^{inj}$, $p_v^{\text{cont}}$ and $p_v^{\text{abnd}}$. We follow Talukdar et al. (2008) in setting $p_v^{\text{inj}} = 1.0$, $p_v^{\text{cont}} = 0.01$ and $p_v^{\text{abnd}} = 0.01$. We briefly experimented with varying the hyperparameters and found this setting provided consistently equal or better results than other hyperparameter settings.

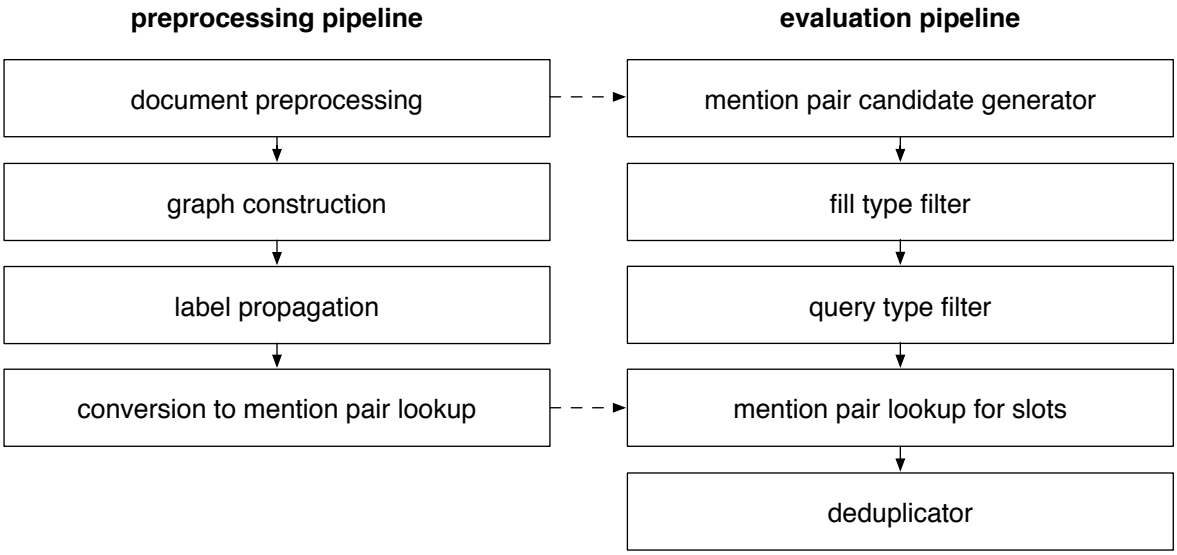



Figure 5.5: Graph-based slot filling pipeline.

The algorithm outputs a label distribution with probabilities of each label. We can apply a probability threshold to determine a final decision as to whether a mention pair expresses a slot. Raising this threshold trades recall for precision.

### 5.5.2 MAD as a node program

In order to support large-scale asynchronous distributed graph processing, we modify MAD to be implemented as a node program in GraphLab (Gonzalez et al., 2012) (now Dato Core[3]). In practice, we run the graph synchronously (distributing graph updates across processes and machines), which makes this primarily an implementation detail, as a parallelisation of the algorithm.

### 5.5.3 Evaluation

When evaluating using this approach, we vary the score threshold for nodes to be accepted as correct fills. We add the graph into our pipeline as in Figure 5.5. The entire propagation process happens prior to evaluation: we predict slot labels for the entire graph (and hence for every mention pair in the corpus). At evaluation

[3] dato.com/products/create/open_source.html

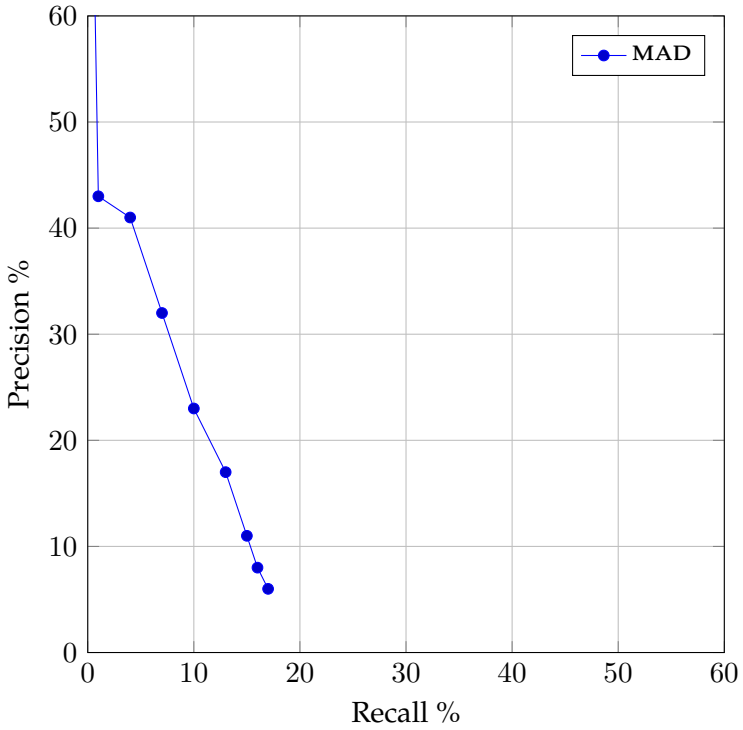



Figure 5.6: PR curve for filtered graph pipeline using MAD. Note that the maximum percentage on each axis is 60%.

time we simply take query-fill mention pair candidates, look them up in the graph, and return any slot labels on that node (above the probability threshold). Integrating this process as part of the pipeline, we generate the results in Figure 5.6. Each point on this graph is a different threshold setting. Note that recall is low, even at low precision. Our top performance on this curve is an F-score of 15%, with a recall of 13% and a precision of 18%. This F1 substantially outperforms the naïve results: the previous filtered pipeline produced a reachability F1 of only 10%, although the recall has dropped from the 19% upper bound. We now look to improve on our results, beginning with improving the training data used.

## 5.6 Additional data

Currently, our propagation process consists of positive labels which all compete in the same space. This disregards two factors which we expect are critical: use

of negative data, and modelling interaction between slots. First we will discuss these as design criteria, beginning with the use of negative data, and then we will approach incorporating these criteria into graph construction and propagation.

**Design Criterion 5: Negative labels should be modelled in the graph.** As discussed in Section 5.3.3, directly applying negative seeds to the graph is problematic when all labels complete in the same space. We will further address this competition in Section 5.7. However, without modifying the propagation process, we can straightforwardly apply a `no relation` label, which indicates confidence that a node does not express *any* relation in the schema. Such a negative label competes with every other slot. The TAC annotations and results do not contain such a label, and so we must add additional annotations.

### 5.6.1 Adding data

Up until this point, we have been using a small number of seeds in the graph, derived from the TAC KBP training data. As well as being very limited for the slots that are represented, several slots are missing entirely. We also only have negative labels relative to each individual slot rather than negative labels for the whole schema, a `no relation` slot. There are two additional sources of training data we could leverage: additional manual annotations or distant supervision.

The largest source of additional annotations is the Stanford MIML-RE data (Angeli et al., 2014), a dataset derived from manual correction of a distant supervision dataset, and we apply this data to the graph. Raw annotation numbers, as well as the number of seeds that we can align into the graph, are given in Table 5.9. As with the TAC annotations, we lose a substantial number of annotations that cannot be aligned to the graph. The MIML-RE data, despite being isolated sentences, contains nominal and pronominal mentions which we do not represent, as we do not make use of coreference resolution we lose these annotations.

We can only align 33% of all annotations (not including `no relation` [4]), this is again below our expected alignment of 40% from the SENTENCE recall upper bound analysis. This additional 7% loss appears to be mostly caused by NE span conflicts: there are a very large number of NE spans where our NER differs from the MIML-RE spans. This is somewhat unexpected as both use the same Stanford NER. This may be a result of some later component of the distant supervision process, an example of an incorrect span in the MIML-RE data is in Example 11. Note that that when running the full sentence through Stanford NER, the NE $[\textit{John Gay}]_{\text{PER}}$ is labelled.

(11) Former $[\textit{Gunns}]_{\text{ORG}}$ director, $[\textit{John}]_{\text{PER}}$ Gay, had previously put...

Using the string match fallback that we used for the earlier annotation—checking for a string match of the query and fill in sentences with two NEs—is less useful here, as nominal and pronominal mentions are included. Using this fallback finds 77% of instances, which is in line with what we would expect from systems using coreference resolution in Chapter 4, suggesting that query and fill NE span alignment is the key reason for annotation alignment loss here.

Despite these alignment issues, this data provides us with a substantially larger set of seeds than we had access to previously, including a set of new `no relation` seeds. Use of this data in addition to the TAC seeds gives us the results shown in Figure 5.7, an increase in performance, with a top F1 of 21%, greater than our previous top F1 by 6%. Overall, the PR curve is better than without this data. However, we note that overall F1 still remains low due to low recall. We will investigate this issue in the next chapter.

We note that this number of seeds is low overall in terms of both the number of nodes in the graph and the distribution over slots. Only 0.05% of all nodes are initially labelled, and some slots are still very limited, with seven slots having

[4] If we include `no relation`, then we can align only 27% of annotations. However, this count of annotations also includes fills that are dates, numbers and strings, so this number is not directly comparable with our NE-based upper bounds.

| slot | count | alignable | seeds |
|---|---|---|---|
| no relation | 11,049 | 1,874 | 466 |
| per:employee of | 4,960 | 1,418 | 679 |
| per:countries of residence | 2,371 | 1,077 | 418 |
| org:city of headquarters | 1,809 | 785 | 50 |
| org:country of headquarters | 1,435 | 416 | 178 |
| per:stateorprovinces of residence | 1,027 | 437 | 239 |
| per:cities of residence | 946 | 401 | 155 |
| org:member of | 822 | 112 | 37 |
| org:top members/employees | 821 | 287 | 176 |
| org:parents | 779 | 174 | 75 |
| org:stateorprovince of headquarters | 738 | 144 | 56 |
| per:country of birth | 507 | 173 | 87 |
| org:founded by | 505 | 158 | 60 |
| per:city of birth | 497 | 261 | 193 |
| org:subsidiaries | 409 | 84 | 30 |
| per:origin | 379 | 41 | 0 |
| per:spouse | 367 | 160 | 63 |
| per:stateorprovince of birth | 259 | 59 | 30 |
| org:members | 180 | 39 | 14 |
| per:children | 162 | 44 | 24 |
| per:parents | 139 | 43 | 75 |
| per:city of death | 121 | 54 | 32 |
| per:country of death | 109 | 42 | 14 |
| org:political/religious affiliation | 96 | 7 | 2 |
| per:schools attended | 74 | 19 | 10 |
| org:shareholders | 47 | 15 | 6 |
| per:other family | 46 | 12 | 2 |
| per:stateorprovince of death | 44 | 7 | 4 |
| per:siblings | 39 | 17 | 6 |

Table 5.9: Counts of annotations of NE slots in the Stanford MIML-RE data, number of annotations alignable using our pipeline, and number of annotations present as seeds in the graph once we have applied our filtering constraints.

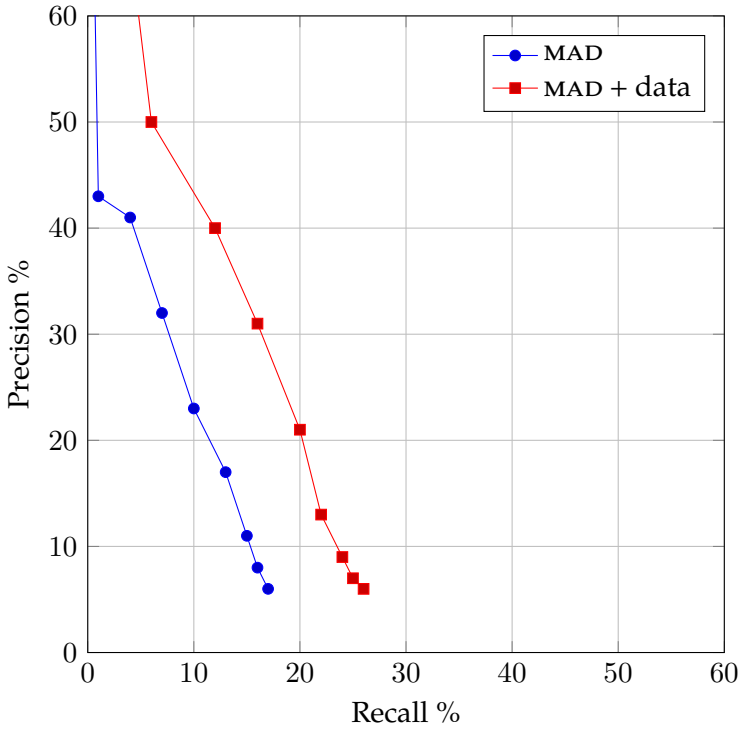



Figure 5.7: PR curve for filtered graph pipeline using MAD and MIML-RE data. Note that the maximum percentage on each axis is 60%.

ten or fewer seeds after annotations are aligned to our graph. This is particularly limiting in the case of slots which have wide variation in representation: `per:other family` captures many more types of relationships than `per:parents`, but has far fewer annotations. Only seven slots and `no relation` have more than 100 seeds.

Now that we have added a course-grained `no relation` label, we want to consider adding finer-grained negative labels, and allowing labels to interact beyond simple direct competition in the graph.

## 5.7 Label interaction

Inspecting the final state of the graph and the state between iterations, we note a key flaw with our label propagation setup. In the current configuration, all labels directly compete. However, this is often not a reasonable assumption. Issues are particularly noticeable for the location slots. The `city of`, `stateorprovince of`

and `country of` slots are distinct and compete, despite all encoding some kind of `location of` information. We want to consider and model the different possible interactions between slots.

**Design Criterion 6: Interaction between slots should be modelled.** Considering all pairs of slots, we find that there are four different types of real-world interactions between slots in SF: *mutually exclusive*, *inverse*, *inverse subset*, and *no interaction*. We first describe these types and then discuss how the interactions are modelled. We note that there are no subset slot interactions in TAC SF, but will discuss this interaction as modelling *subset* is required to model *inverse subset*.

**Mutual exclusion** Mutually exclusive slots are those which cannot possibly filled by the same value no matter the context, as determined by the TAC definition of slots. Slots that are mutually exclusive include all location-based granularities. A fill cannot be both a `city` and a `country`. We note that in some cases it may still be valid to propagate location information between granularities. Example 12 may be a fill for any `LOC of residence` slot depending on the granularity of LOC:

(12) PER lived in LOC.

For location-based slots specifically, we collapse slots together for propagation, i.e. to `per:LOC of residence`, and then split based on gazetteer for output. This is not an entirely precise approach, as there instances where this is not valid, Example 13 is a valid fill for `per:city of residence` but never for `per:country of residence`, yet these labels will still be collapsed together. However, we expect these cases to be relatively rare.

(13) PER lived in the city of LOC.

| category | slots |
|---|---|
| employee v. member | per:employee of, per:member of |
| student v. member | per:schools attended, per:member of |
| family | per:spouse, per:parents*, per:children*, per:siblings, per:other family |
| ORG hierarchy | org:member of*, org:members*, org:parents**, org:subsidiaries**, org:political/religious affiliation |
| shareholders v. affliation | org:shareholders, org:political/religious affiliation |
| founder v. affliation | org:founded by, org:political/religious affiliation |

Table 5.10: Mutally exclusive and inverse slots, as derived from the TAC slot definitions. Slots in a category are mutually exclusive. * and ** indicate slots in a category which are inverses of each other.

Non-location mutually exclusive cases which are explicitly represented in the label propagation process are those which are by the TAC definition strictly mutually exclusive, and we list these categories in Table 5.10. We expect that some of these cases apply outside TAC definitions, but strict mutual exclusion decisions do come down to the schema. For example, the slots in *student v. member* are mutually exclusive, because person cannot be a member of a school, nor can a person be a student of an organisation that is not an educational institution, by definition.

**Inverse** Inverse slot interactions are between mutually exclusive slots which are also inverses of each other, such as per:parents and per:children. These are marked in Table 5.10.

**Subset** Subset slot interactions are between slots where one slot is a subset another. No slots exist in this relationship in the TAC KBP schema. However, we still want to model this interaction for the inverse subset (below), and as the subset

interaction is in the space of relations that exist outside TAC. ACE is defined with relations and sub-relations, and we may want to model both of these in the graph. In TAC itself, it may still be useful in cases such as modelling location granularities: having both a `per:LOC of residence` and a `per:city of residence` slot may provide a better model for Examples 12 and 13 above.

**Inverse subset** Inverse slots are slots where one slot is a subset of the inverse of the other. There is only one slot which requires this interaction, `org:top members/employees`, which is inverse subset to `per:employee or member of` (or `per:employee of` + `per:member of`). A top employee is always an employee but not vice versa. When `per:employee of` and `per:member of` are separate this needs to be modelled slightly differently, but we will also refer to this as inverse subset. We note that in experiments up to this point, `per` slots have been prohibited from occurring in the same graph as `org` slots, but we want to allow for this information to be leveraged across types.

**No interaction** Many pairs of slots do not interact in a binary fashion, including several which may appear to exclude each other. Consider `org:top members/employees`, `org:founded by` and `org:shareholders`. A single PER may fill any number of these slots for a given ORG. Note that there is no interaction only by the definition of slots. It is likely that many slots may be correlated, just not by definition, and we address this case next.

**Correlated** There is no defined interaction in the above **no interaction** slots, but it often reasonable to make inferences based on the likely co-occurrence of these slots. For example, while a company `org:founded by` and `org:shareholders` have no definition-based interaction, it would not be unreasonable to infer that a company founder is also a shareholder, and we can derive such a prior given co-occurrence in training data. Hence, we want to be able to capture this when modelling slot

interaction. Note that the same directionality applies here: it is more likely that a founder of a company is also shareholder than the reverse.

**Implicit mutual exclusion** We note that the above interactions still allow for a number of slots to overlap: for example, it is still possible for a (PER, LOC) mention pair to be labelled with `per:country of birth`, `per:country of death`, `per:countries of residence` and `per:employee of`. However, it is unlikely that the context of a single mention pair (e.g. a single sentence) expresses all of these fills. Even expressions of more than one fill in a single context are much less frequent than expressions of a single fill. We refer to this as linguistic mutual exclusion.

### 5.7.1 Modelling slot interaction

Baseline label propagation approaches allow all labels to compete. In the case of MAD, extensions such as MADDL (Talukdar and Crammer, 2009) and ICMAD (Wang et al., 2011) modify the algorithm to allow for non-mutually exclusive labels and subset labels respectively. As opposed to modifying the objective to support our modelling of slot interactions, we choose to separate the propagation into multiple binary propagations which are simultaneously processed and which may or may not interact. This is analogous to using a many-vs-one approach, except that we allow individual propagations to interact in different ways.

We compare this approach to (Wang et al., 2012), who add these constraints to label propagation by implementing an Integer Linear Programming-based (ILP) constraint satisfaction layer. By coupling the label propagation and ILP layers they improve precision and recall over their previous work, (Wang et al., 2011). In this section, we follow a different approach, splitting the propagation into multiple binary propagation without modifying the objective function of the MAD algorithm itself.

For each slot $s$, we construct a separate distribution which uses the labels $s$ and $\neg s$. These separate distributions are propagated in parallel, and are aggregated after every iteration: $s$ models the slot (and is the label used for final extraction), and $\neg s$ represents every other slot that competes or otherwise interacts with this (including `no relation`), weighted based on our set of interaction types. For every iteration, we propagate each binary pair of labels by MAD. If we define every slot to compete with other slot, then this is the same MAD process as before: a distribution of (`per:LOC of birth`: 0.4, `per:LOC of residence`: 0.6) simply becomes two distributions (`per:LOC of birth`: 0.4, ¬`per:LOC of birth`: 0.6) and (`per:LOC of residence`: 0.6, ¬`per:LOC of residence`: 0.4). While we now have two separate distributions, probabilities of the positive labels remain the same.

We could also establish an opposite extreme, where there is no interaction between any labels. However, we want to model the more complex interaction between labels that sits between these extremes by modifying each $s$ and $\neg s$ based on the other parallel propagations.

We define a matrix $E$, with all labels (slots) as rows and columns. Each cell $E_{ij}$ is assigned a value in $-1 < E_{ij} < 1$, to the $\neg i$ label. While we allow $E_{ij} \in \mathbb{R}$, for this rule-based setup we only use values in $\{-1, 0, 1\}$. We also note that this allows for non-symmetrical label interaction, required for subset relationships.

In effect, our labelling on each node, $\mathbf{Y}_i \in {\mathbb{R}_+}^m$, is now distinct for each label, and is defined by a positive and negative label, $s_i$ and $s_{\neg i}$, where:

$$\mathbf{s}_i = \mathbf{Y}_i - \sum_{i \neq j, E_{ij} < 0} \mathbf{Y}_i E_{ij} \tag{5.11}$$

$$\mathbf{s}_{\neg i} = \sum_{i \neq j, E_{ij} > 0} \mathbf{Y}_i E_{ij} \tag{5.12}$$

The output for each node is a similar set of vectors, and we can again apply a probability threshold to determine if a label $s$ should be extracted for a particular mention pair.

**Mutual exclusion** If the labels are mutually exclusive, then $E_{ij} = 1$ and $E_{ji} = 1$, and $l$ entirely competes with $i$ as part of the $\neg i$ label. Notably, the `no relation` slot always completes entirely with every other label.

**Inverse** While inverse slots are conceptually distinct from other mutually exclusive labels, they are still mutually exclusive for the purpose of interaction. Consider `per:parents`, `per:children` and `per:other family`. While `per:parents` and `per:children` are mutually inverse, all three slots compete with each other. A high confidence in `per:parents` shouldn't downweight the likelihood of `per:children` any more than it would `per:other family`. These are all mutually exclusive slots. Similarly, a low confidence in `per:parents` says nothing specific about the likelihood of `per:children`. Hence, we model this in the same way as mutual exclusion, and assign $E_{ij} = 1$ and $E_{ji} = 1$ in this case.

**Subset** The core idea for subset slots is that when a slot $i$ is a subset of $j$, it contributes entirely to the slot, but $j$ does not contribute to $i$. We model this as assigning the value $E_{ij} = -1$ and $E_{ji} = 0$. Algorithmically, we implement this as adding to the $l$ rather than $\neg l$, as in Equation 5.11.

**Inverse subset** The inverse subset relation is the represented in the subset relationship above.

**No interaction** If labels do not interact at all, then $E_{ij} = 0$ and $E_{ji} = 0$, and $l$ will not contribute to the $\neg s$ label.

**Correlated** While we only make use of definition-based values $-1, 0, 1$ in our matrix $E$, other real values $-1 < x < 1$ allow for correlation to be modelled. Values $-1 < x < 0$ model correlation, such labels will add to the positive distribution of a target slot, weighted by $x$. Similarly, values $0 < x < 1$ allow for competition (but not mutual exclusion) between slots.

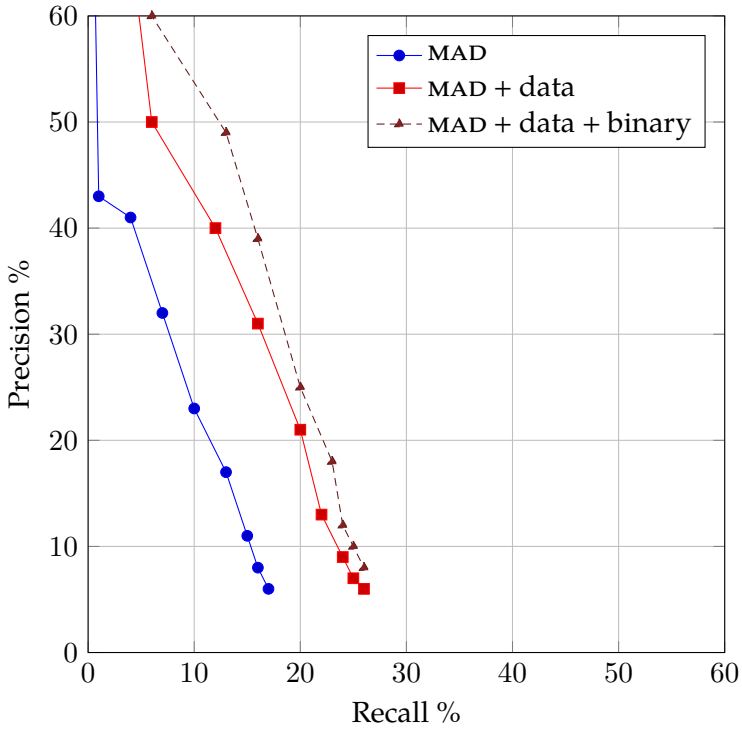



Figure 5.8: PR curve for filtered graph pipeline using MAD with binary distributions and MIML-RE data. Note that the maximum percentage on each axis is 60%.

### 5.7.2 Evaluation

We replace the propagation part of our slot filling pipeline with this algorithm, and generate the results in Figure 5.8. We include the best results for each evaluation in Table 5.11. The results show 2% improvement over the previous results for top F-score, with a top F1 of 23%. The use of binary distributions to model slot interaction appears to work well, with this approach having higher or equal precision and recall at all tested thresholds. We note that we have not attempted to optimise the values within the interaction matrix, or add weights derived from data for correlated interactions, and we leave this a a promising direction for future work.

We compare these results to Angeli et al. (2014). To informally compare performance on our filtered dataset, we take their reported per-slot results and apply

| approach | R (%) | P (%) | F (%) |
|---|---|---|---|
| naïve NON-UNIQUE | 29 | 4 | 6 |
| naïve reachability | 21 | 6 | 9 |
| filtered NON-UNIQUE | 28 | 5 | 8 |
| filtered reachability | 19 | 7 | 10 |
| MAD | 13 | 18 | 15 |
| MAD + MIML-RE data | 16 | 31 | 21 |
| MAD + MIML-RE data + binary distributions | 16 | 39 | 23 |

Table 5.11: Best results for each experiment in this chapter.

them to our distribution of slot fills. These results are in 5.12. Overall F1 for the Angeli et al. (2014) comparison is 24%, compared to our 23%.

| slot | total | our system correct | our system wrong | Angeli et al. (2014) R (%) | Angeli et al. (2014) P (%) | Angeli et al. (2014) tp | Angeli et al. (2014) fp |
|---|---|---|---|---|---|---|---|
| org:top members/employees | 118 | 32 | 25 | 60 | 26 | 71 | 336 |
| per:employee of | 71 | 18 | 35 | 46 | 32 | 33 | 151 |
| per:member of | 47 | 1 | 2 | 0 | 0 | 0 | 0 |
| org:subsidiaries | 32 | 2 | 5 | 3 | 25 | 1 | 96 |
| org:parents | 24 | 1 | 5 | 54 | 26 | 13 | 68 |
| org:country of headquarters | 22 | 6 | 3 | 62 | 62 | 14 | 13 |
| per:countries of residence | 20 | 8 | 13 | 40 | 42 | 8 | 28 |
| org:city of headquarters | 19 | 5 | 2 | 61 | 52 | 12 | 18 |
| org:shareholders | 18 | 0 | 0 | 0 | 0 | 0 | 0 |
| org:stateorprovince of headquarters | 17 | 0 | 2 | 35 | 64 | 6 | 10 |
| per:children | 17 | 3 | 1 | 18 | 62 | 3 | 10 |
| per:cities of residence | 17 | 0 | 2 | 30 | 52 | 5 | 16 |
| org:member of | 11 | 0 | 8 | 0 | 0 | 0 | 0 |
| per:stateorprovinces of residence | 11 | 3 | 3 | 7 | 33 | 1 | 22 |
| org:members | 8 | 4 | 1 | 0 | 0 | 0 | 0 |
| per:spouse | 8 | 0 | 3 | 85 | 54 | 7 | 7 |
| org:founded by | 7 | 2 | 7 | 38 | 89 | 3 | 1 |
| per:city of birth | 6 | 0 | 0 | 17 | 50 | 1 | 6 |
| per:other family | 6 | 0 | 0 | 0 | 0 | 0 | 0 |
| per:siblings | 6 | 0 | 1 | 33 | 50 | 2 | 6 |
| per:country of birth | 3 | 0 | 6 | 0 | 0 | 0 | 0 |
| per:parents | 3 | 0 | 2 | 28 | 64 | 1 | 2 |
| org:political,religious affiliation | 2 | 0 | 0 | 100 | 25 | 2 | 6 |
| per:city of death | 1 | 0 | 1 | 19 | 75 | 0 | 0 |
| per:country of death | 1 | 1 | 2 | 10 | 100 | 0 | 0 |
| per:stateorprovince of birth | 1 | 0 | 4 | 10 | 50 | 0 | 1 |
| per:schools attended | 0 | 0 | 0 | 48 | 78 | 0 | 0 |
| per:stateorprovince of death | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| total | 535 | 86 | 133 | 34 | 19 | 183 | 797 |
| F1 (%) | | | 23 | | | | 24 |

Table 5.12: Comparison with Angeli et al. (2014) results. *tp* indicates comparative true positives, *fp* indicates comparative false positives.

## 5.8 Discussion

In this chapter, we have made improvements to the basic label propagation approach, establishing a process for TAC KBP slot filling by label propagation. This approach appears to be reasonable. However, overall performance remains low. In this chapter we achieve a maximum F-score of 23%. Inspecting the graph provides us with avenues for improvement, which we will explore in Chapter 6. In Section 6.2.2, we provide a detailed analysis finding that errors are due to error in the pipeline (in coreference resolution, parsing and NER typing), ambiguous context, and instances that are correct but missing from results. The graph is still quite disconnected, with a few nodes representing short-range dependencies such as $[\ \text{PER} \xrightarrow{\text{poss}} \text{LOC}\ ]$ and $[\ \text{per} \xrightarrow{\text{prep of}} \text{LOC}\ ]$, creating relatively dense parts of the graph, but much of the graph remaining quite sparse. We expect this sparsity makes the graph labelling fairly unstable in regards to initial distribution of labelled seeds. There may be very large regions that are initially unlabelled, and many of these nodes will be labelled by potentially distant initial seeds that dominate an unlabelled area, contributing to semantic drift. Despite a large amount of additional training seeds, there are still very few seeds relative to the number of nodes. Only 0.02% of nodes are initially labelled. We investigate these issues to improve on this label propagation setup in the next chapter.

## 5.9 Summary

In this chapter, we have proposed that a label propagation approach is a good fit for slot filling. We focus on using a graph-based label propagation approach to directly model behaviour and assumptions we make concerning the task, based on a number of design criteria. We target explicitly representing the underlying data and task in the graph, applying a label propagation approach on this setup.

Our first contribution of this chapter was a naïve slot filling model, derived from the reachability approach of the previous chapter. We then contribute a graph-based model of the core RE component of model of slot filling. This baseline graph gives an F1 of 10%, which is a reasonable starting point, considering that only a naïve propagation algorithm has been used. We contribute a number of design criteria that motivate design of the graph.

Following other work in label propagation, we apply the MAD algorithm to our task. This increases our F1 to 15%, and provides us with a launch point for further analysis. We analyse the distribution of the graph in detail, and expect that the small number of seeds—658 total—is a major limiter to performance. We align the Angeli et al. (2014) MIML-RE data to our graph, which provides us with 8,401 additional labelled nodes. Notably, this also provides us with an explicitly negative label, which was missing from the original set, giving us a jump in performance an F1 of 21%.

In order to better leverage the data we do have, and to better model the interaction between instances, we developed an analysis for the interactions between different slots. For example, `per:parents` and `per:children` are inverses, and we categorise the relationships between slots into a small number of categories. We map this categorisation to an interaction matrix, and use this matrix to define a modification of MAD. This modification allows for slots to interact in different ways. Some slots mutually exclude each other in propagation, such as `per:parents` and `per:children`, and other slots do not interact at all and simply propagate without regard for given slots based on these defined constraints. This approach outperforms our previous configuration, and gives us our best F1 in this chapter of 23%, showing that modelling this interaction helps performance

Defining a useful representation of the context of a potential slot fill that is discriminative enough for complex relations is a major concern for SF and this work. In the next chapter, we experiment with modifying the topology of our graph to

reflect different, more useful representations, aiming to decrease the sparsity and disconnectedness in the graph to provide a result that has higher recall and is higher overall.

# 6 Sparsity and disconnectedness

In the previous chapter, we scaled our graph to a substantially larger training dataset, and implemented several modifications to a label propagation approach. However, performance remained fairly low. This suggests that there is a larger underlying problem with the graph representation of the task or the setup of the task itself. In this chapter, we perform a number of analyses in order to better identify these issues.

Firstly, we investigate the substantial recall lost in the construction of our graph in Section 6.1. In particular, we find that the graph is substantially disconnected, and large numbers of nodes are in disconnected subgraphs without any seed nodes. In Section 6.2, we add a trigger word (Yao et al., 2011) feature, from tokens derived from the dependency path, as a more general but still discriminative feature. The addition of these nodes and respective edges results in all target fill nodes being in the same graph as required seeds. Analysis of precision errors reveal that there are instances which require more context to be identified as correct and incorrect. To account for this, we add syntactic modifiers to our other features in Section 6.3. This allows the graph to better discriminate between instances, although these features are very sparse and do not have a substantial effect on performance.

While inspecting the graph and results, we find that a large number of errors occur in very close proximity to seeds. This is a concern: if distributions are conflicted close to seeds, it is likely that these will cause considerable semantic drift, and it is unlikely that more distant nodes could be correctly labelled. We contribute

| pipeline | tp | tp + fp | R (%) | P (%) | F (%) |
|---|---|---|---|---|---|
| SENTENCE | 213 | 20,704 | 40 | 1 | 2 |
| NON-UNIQUE | 162 | 3,328 | 30 | 5 | 8 |
| reachability | 138 | 2,092 | 26 | 7 | 11 |
| binary | 143 | 2,641 | 27 | 5 | 9 |

Table 6.1: Current graph upper bounds with *binary* evaluation.

a detailed analysis of some of these nodes, identifying that lack of context and subtle differences in slots contribute to nodes with the same feature representation being assigned a wide number of potentially conflicting slots. Alongside this, we consider evaluating when strict justification is required from the TAC 11 evaluation data, and find that our results drop substantially as many correct instances are marked as incorrect. We note that these are annotation issues which need to be resolved for better evaluation, and we will continue this analysis in Chapter 7.

## 6.1 NON-UNIQUE and reachability

In Chapter 5, we provided recall upper bound analysis for the graph representation. These numbers are repeated in Table 6.1. As before, SENTENCE requires that query and fill NEs are in the same sentence (with no coreference resolution), NON-UNIQUE filters out pairs that have a unique dependency path between them (this is implicit in the construction of the graph), and *reachability* uses a maximum recall bootstrap over the graph. Until now, we have only provided high-level analysis of the recall loss of the NON-UNIQUE and reachability bounds, and we analyse these errors in detail now.

Firstly, we consider the fills lost directly due to not being represented in the graph as the path they occur with is unique in the corpus, i.e. the difference between SENTENCE and NON-UNIQUE. Investigation of these errors finds them to be caused either by parse errors, such as the incorrect path [ PER $\xrightarrow{\text{appos}}$ chairman $\xrightarrow{\text{prep of}}$ minis-

ter $\xrightarrow{\text{poss}}$ ORG ] between M. Enkhbold and MPRP in Example 1,[1] or genuinely infrequent paths such as [ PER $\xleftarrow{\text{nsubjpass}}$ elected $\xrightarrow{\text{prep as}}$ associate $\xrightarrow{\text{prep of}}$ ORG ] between Chen Zhu and Institute of Medicine in Example 2.

(1) After that, M. Enkhbold, former chairman of MPRP and Mongolia's then prime minister, offered his resignation.

(2) China's Health Minister Chen Zhu has been elected as foreign associate of the United States Institute of Medicine (IOM), according to the IOM website.

This NON-UNIQUE upper bound has already resulted in recall loss, but there is a further gap between this upper bound and reachability. To more generally explore reachability, we consider a *binary* reachability evaluation with only two labels, `relation` (made up of all positive slot labels) and `no relation`. Instead of being concerned with extracting a particular correct slot, we only require that any positive label (i.e. `relation`) is found for a fill to be extracted. This continues to be effectively restricted by NE types, i.e. the `relation` label in the PER PER graph is composed only of PER PER slots, because there are only PER PER seeds in that subgraph. We add results for this experiment to our current graph upper bounds in Table 6.1. This configuration achieves 1% higher recall than normal reachability. This indicates that the graph has disconnected sections with different seed distributions. If these disconnected sections had seeds which included the same slot types, then `relation` would be the same across these sections, and *binary* and *reachability* numbers would be equal. Instead, 1% of target fills are in subgraphs isolated from any relevant (i.e. correct) seed. Additionally, the 3% recall gap between binary and NON-UNIQUE indicates that there are disconnected subgraphs in the graph that contain target fills but do not contain any seeds at all.

[1] Incorrectly parsed as [former chairman of [MPRP and Mongolia's then prime minister]] instead of [[former chairman of MPRP] and [Mongolia's then prime minister]].

This difference in recall is only between upper bounds, but we expect that this will also affect propagation over parts of the graph where graphs are not entirely disconnected but still have few connections. Understanding this graph disconnectedness is key to understanding training data and feature sparsity of the task as a whole, and we investigate this next.

## 6.2 Improving graph connectedness

Reachability is dependent on two things: the connectedness of a graph, and the distribution of seeds across that graph. Reachability could be maximised in a highly disconnected graph, by using a very large number of seeds that we distributed perfectly across disconnected sections. In practice, this is not possible, and so the graph must minimise disconnected sections, with a smaller number of seeds distributed as usefully as possible across any remaining disconnected components. As we are evaluating relative to a particular set of training data, it is possible that this particular set is overly clustered in particular subgraphs. By further analysing the disconnectedness we may be able to identify regions of the graph where training data is lacking, and this may reveal issues in the construction of the graph or in the training data itself.

We note that as a starting point, our current filtered graph is made up of six distinct, explicitly disconnected subgraphs for the query-fill type pairs, i.e. subgraphs for PER PER, PER ORG, PER LOC, ORG PER, ORG ORG and ORG LOC. We generate several metrics for the distribution of disconnected subgraphs within these NE type pair subgraphs. Table 6.2 provides statistics for the distribution of nodes in subgraphs. We note that there are a large number of disconnected subgraphs across all of the type pairs, in particular PER PER has 106,628 subgraphs. While many nodes are contained in the largest subgraph, 86.4% in the PER PER case, across

| | | nodes | | | |
|---|---|---|---|---|---|
| types | subgraphs | count | min | max | % in top-1 |
| PER PER | 106628 | 4,741,069 | 4 | 4,097,586 | 86.4 |
| PER ORG | 41891 | 1,905,222 | 3 | 1,652,777 | 86.7 |
| PER LOC | 62280 | 2,310,087 | 4 | 1,939,992 | 84.0 |
| ORG PER | 41685 | 2,036,654 | 3 | 1,785,527 | 87.7 |
| ORG ORG | 27652 | 1,188,487 | 3 | 1,013,466 | 85.3 |
| ORG LOC | 26469 | 1,002,356 | 4 | 846,230 | 84.4 |

Table 6.2: Subgraph statistics for the filtered graph. *count* is number of subgraphs for each type pair signature, the other statistics are over numbers of nodes in subgraphs (e.g. *max* is the maximum number of nodes in a subgraph).

these subgraphs 12%–16% of nodes are disconnected from the largest subgraph and will have separate distributions of nodes for the purpose of reachability.

Our goal now is to maximise recall by minimising this disconnectedness. By definition, to make the graph more connected we need to add edges. The most straightforward method to do this, without fundamentally changing the graph topology, is to add additional feature nodes which generalise over more contexts and allow for more mention pairs to be connected via feature nodes. This is effectively creating a richer or more general representation of a mention pair instance by increasing the size of the feature space. For our current graph, sparsity and uniqueness of dependency paths is a major contributing factor to disconnectedness. As with other RE feature sets, such as in Mintz et al. (2009), we do not want to overgeneralise. A bag-of-words context feature, for example, will likely overgenerate spurious features and result in too large a set of features to represent in the graph without further pruning. For these experiments, we want to target minimally generating additional feature nodes while maximally increasing recall. To this end we follow Yao et al. (2011) in using a *trigger word* feature to generalise, where trigger words are every lemma on the dependency path. This is still a very general feature, and we expect it will substantially connect the graph. However, we expect

that overgeneration will be limited. This feature is derived from dependency paths, which are themselves a restricted, discriminative context, and so we expect these trigger word features to still be reasonably discriminative.

We represent trigger word features in the same format as path features, but without the path elements, e.g. [ ORG president PER ]. Note that president is a word on the dependency path between the two NEs and may or may not occur in the lexical sequence between the two NEs. These are included in Figure 6.1 (we have removed some of our previous nodes for clarity). Note that edges between entity pairs and mention pairs have not changed, and these additional feature nodes instead substantially increase connections between mention pair nodes.

Now that we have more than one type of feature connected to individual mentions in the graph, there is a question of whether there should be edges between features. From the perspective of Design Criterion 1 in Chapter 5, representing co-occurrence of features by edges is probably ideal. Features co-occur and are semantically similar for relations simply because they co-occur. The entire graph is built on this assumption, and smoothing over nodes that co-occur is fundamental to the graph. However, adding these edges compounds errors due to feature non-independence. Not only do dependent features act independently in our graph, but they now reinforce each other. Our ongoing goal is to represent the data in the graph, and as long as this co-occurrence is represented it is up to the propagation to take this into account. In the case of MAD, features that occur frequently with other features will be downweighted in label propagation due to their high entropy. We experimented with not including these extra edges, but differences in results are minimal. We choose to include these edges to follow Criterion 1, and include normalised co-occurrence edges between features that occur on the same mention pairs. It is possible that these edges would be more appropriate as directional edges, allowing only specific features to propagate to general features or vice versa. However, we leave that for future work. As edges are normalised per-node, more

| counts | filtered | with triggers | % increase |
|---|---|---|---|
| entity pairs | 3,275,446 | 7,459,555 | 128 |
| mention pairs | 9,249,592 | 22,357,171 | 142 |
| features | 1,007,978 | 1,208,523 | 20 |
| total nodes | 13,533,016 | 31,025,249 | 129 |
| edges | 37,780,554 | 207,296,600 | 449 |
| seeds | 9,059 | 15,037 | 66 |

Table 6.3: Graph profile for graph with trigger features, compared with *filtered* with MIML-RE data from Chapter 5.

| pipeline | tp | tp + fp | R | P | F |
|---|---|---|---|---|---|
| NON-UNIQUE | 180 | 5,235 | 34 | 3 | 6 |
| binary | 180 | 5,232 | 34 | 3 | 6 |
| reachability | 175 | 4,586 | 33 | 4 | 7 |

Table 6.4: Graph upper bounds with trigger word features.

specific nodes will propagate more strongly than general nodes, as they have lower entropy, but the implications of this could be further considered.

The addition of these trigger features to our graph substantially increases the size of the graph, with details in Table 6.3. The number of nodes increases by 129%, and as there are many edges between these nodes, the number of edges increases by 449%. We note that this is still a reasonable graph size for processing in memory, but may not scale well with larger corpora. Upper bound metrics for this new graph are in Table 6.4. NON-UNIQUE requires a mention pair to express any non-unique feature, and not strictly a unique dependency path, and has increased by 4% from the previous graph. Critically, with the exception of 5 `org:shareholders` instances (lost because there are no seeds in the ORG ORG graph in our training data, if there was at least one seed these would be found), our reachability is now equal to our NON-UNIQUE upper bound. Ultimately, the graph is now substantially more connected. We can see this reflected in the subgraph distributions in Tables 6.5.

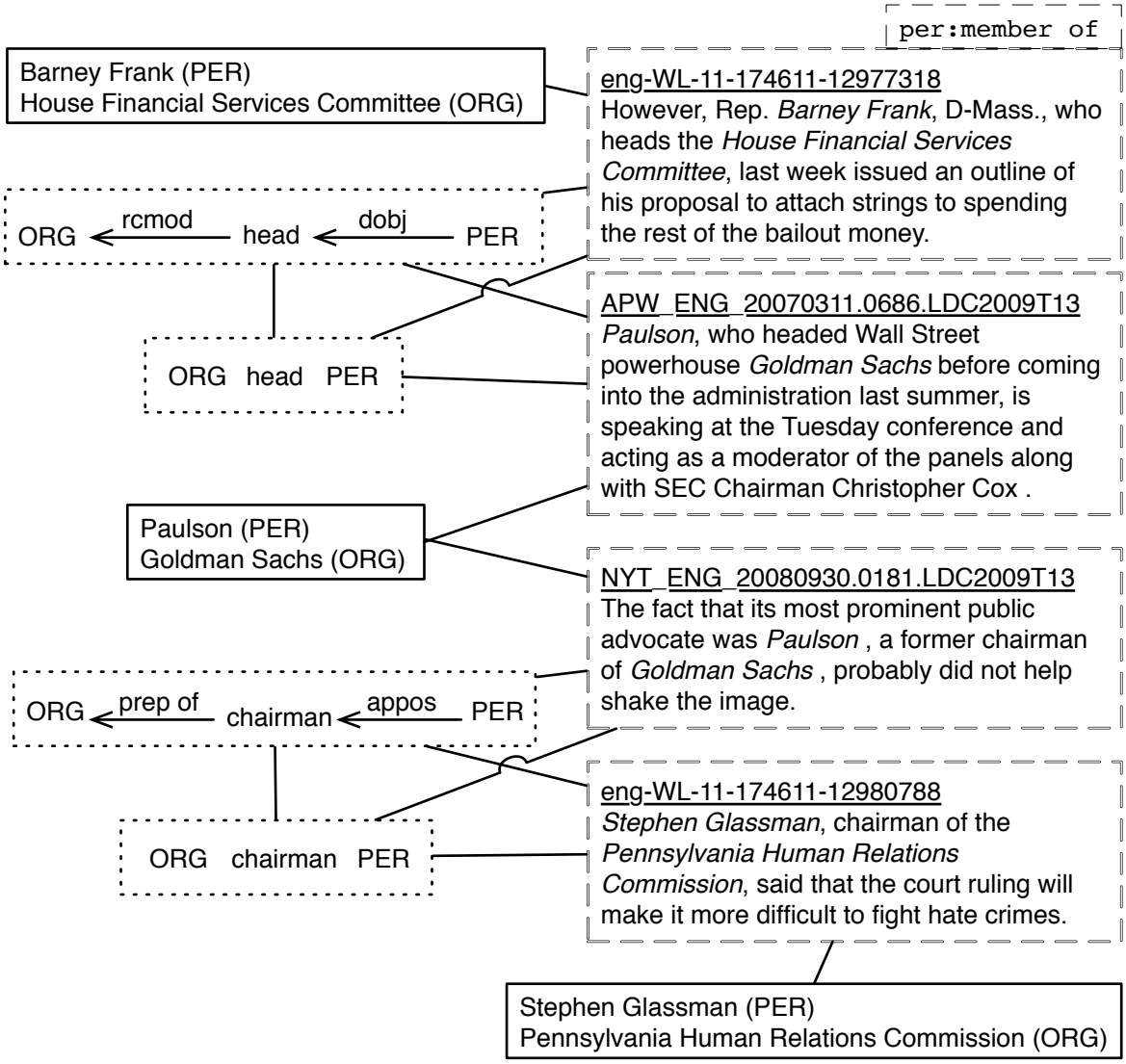



Figure 6.1: Graph with additional trigger word feature nodes. `per:member of` indicates a seed node.

| | | nodes | | | |
|---|---|---|---|---|---|
| types | subgraphs | count | min | max | % in top-1 |
| PER PER | 5,263 | 5,706,077 | 4 | 5,651,512 | 99.0 |
| PER ORG | 32 | 2,514,511 | 3 | 2,506,266 | 99.7 |
| PER LOC | 1,554 | 3,160,311 | 4 | 3,144,930 | 99.5 |
| ORG PER | 831 | 2,514,413 | 4 | 2,506,179 | 99.7 |
| ORG ORG | 1,646 | 1,512,679 | 3 | 1,496,581 | 98.9 |
| ORG LOC | 746 | 1,394,949 | 4 | 1,387,838 | 99.5 |

Table 6.5: Subgraph statistics for the filtered graph. *count* is number of subgraphs for each type signature, the other statistics are over numbers of nodes in subgraphs (e.g. *max* is the maximum number of nodes in a subgraph).

There are roughly two orders of magnitude fewer, e.g. the PER PER subgraph has reduced from 106,628 to 5,263 subgraphs, and 99% of the nodes are contained in the main subgraph.

### 6.2.1 Evaluation

Results for the graph are in Figure 6.2, along with our previous best results. Our recall upper bound has improved, although this only helps F1 at low precisions; at high precisions this approach achieves lower F1. This is likely due to a relatively high threshold being required to maintain precision, now that more propagation is occurring over more edges. Note the position of the threshold dots, which are much more skewed towards low precision. On both lines, dots from left to right set the same threshold. We can see that while the first dots are relatively close, the second dots on both lines are much further apart, with a 20% difference in precision. Similarly, the third points are separated by a 26% precision difference, for a gain for of 12% recall for the trigger graph results. It is likely that this new graph has traded precision for recall. This is not particularly surprising as these features are less discriminative. The best F1 is 23% on both configurations. We

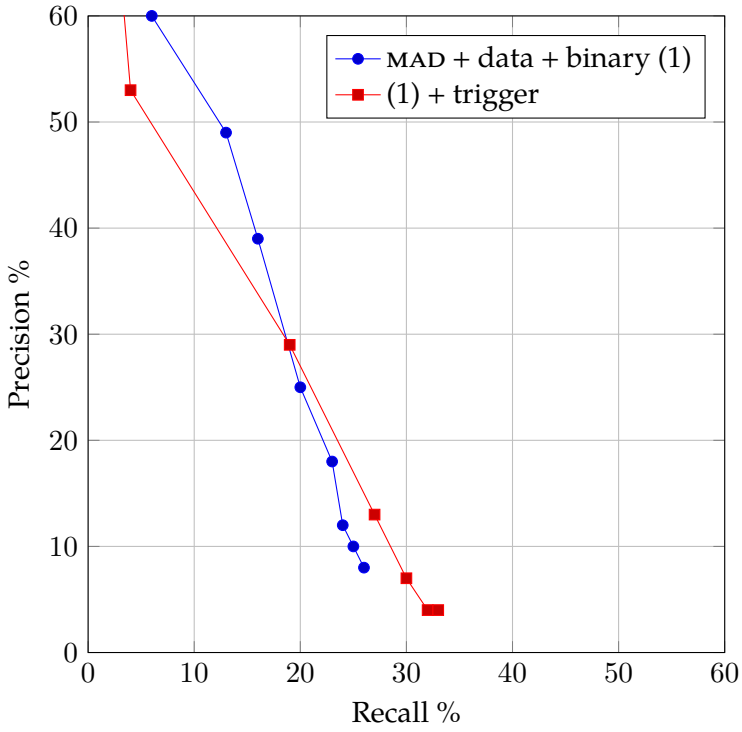



Figure 6.2: PR curve for label propagation. Comparison of earlier results with addition of trigger word features.

choose to continue including these extra features, as they represent more detail in the graph, as per Criterion 1. We have been targeting recall so far in this chapter, and having made reasonable gains, particularly to the upper bound, we now turn our attention to precision errors in this evaluation.

We note that Yu and Ji (2016) had great success with the use of trigger word features, despite them not being of much use here. It is difficult to compare the use of these features, as the two underlying approaches are quite different. However, we can speculate on the reasons for this. Firstly, Yu and Ji (2016) likely had a much higher recall upper bound. As we saw in this section, trigger word features are high recall, and as they leverage trigger sets there are more features available for higher recall. They also make use of coreference which, as seen in Chapter 4, is a substantial difference. Finally, their precision would have also been higher than ours, as they only use a small set of trigger words for each slot, and ultimately this

leads to higher performance. It's likely the trigger words won't actually allow for more discriminative decisions, and so improving performance will see the same challenges that we see here. Taking their unsupervised technique and adding it to a label propagation-based approach is interesting future work.

### 6.2.2 Precision error analysis

As we have discussed, we see in Table 6.5 that most nodes are now in the six main subgraphs. All seed nodes are included in these main subgraphs, except for two `no relation` seeds in PER PER. This gives us maximum reachability: for this particular dataset all other seed nodes and all fill nodes are in the main subgraphs. These more general trigger word features are, by definition, less discriminative than our original shortest dependency path feature space. While it is useful to increase the recall upper bound, we still need to be able to discriminate between correct and incorrect instances. We now look at precision errors to get some measure of the errors introduced by these new features.

For analysis, we rank fills that the system returns as correct by score, and compare these with the gold answers. We look at the top 100 results. 38 of these are incorrect. A number of these errors are due to pipeline error: Three cases require proper noun coreference resolution to resolve inexact fills. In Example 3, the correct fill is Eddy Hartenstein, but this is not present in the context of the sentence. Note that is not a recall error, as we can find this particular instance, rather the error is because we cannot canonicalise the name Hartenstein.

(3) **wrong relation:** (DirecTV, `org:top members/employees`, Hartenstein)
**context:** *Hartenstein* was chairman and CEO of *DirecTV* from 2001 to 2004.

Two errors are caused by parse error, such as the error in Example 4.

(4) **wrong relation:** (Ahmad Qattan, `per:country of residence`, Egypt)
**path:** [ $\xleftarrow{\text{dobj}}$ send $\xleftarrow{\text{advcl}}$ represent $\xrightarrow{\text{nsubjpass}}$ ]
**context:** *Egypt* will be represented by State Minister for Legal and Parliamentary Affairs Mufid Shehab while oil powerhouse Saudi Arabia, a major supporter of the Lebanese government, is sending Arab League ambassador Ahmad Qattan.

Five errors are due to LOC granularity errors (e.g. in Example 5, North America is not a LOC; similar cases occur for entities like Europe).

(5) **wrong relation:** (Denso, `org:country of headquarters`, North America)
**context:** Douglas Patton, senior vice president for *Denso* in *North America*, said the key to any system was not so much the type of warning it gave as the way the system caught a driver drifting off.

The lack of discriminative context relates to many of the remaining errors. 16 errors do not have an immediately apparently error in their representation, but may have an ambiguous feature set that is affected by context: in Example 6, suspended can apply to any person from any role, including a top employee, and the graph has propagated this label as a form of semantic drift.

(6) **wrong relation:** (Jacksonville Jaguars, `org:top members/employees`, Justin Durant)
**context:** The *Jacksonville Jaguars* have suspended rookie linebacker Justin Durant and second year offensive lineman Richard Collier for two games in the wake of their arrests last week.

Ten of these cases are instances which appear to be correct (at least in the context of an individual sentence), but are not seen in the results data (i.e. are not part of the human annotation or participating system results), such as Example 7.

(7) **relation:** (DirecTV, `org:city of headquarters`, El Segundo)
**context:** *DirecTV*, based in *El Segundo*, California, is the leading U.S. satellite TV service with 16.8 million subscribers.

As many errors appear to require more context to identify them as correct or incorrect, we next address this lack of context.

## 6.3 Improving graph discriminability

In our analysis in the previous section, we note that 16 of 38 errors are due to lack of discriminative context. As this is the largest category of error, we now want to consider improving precision, by increasing the discriminability of the graph to a level that reflects what is required by the task. Our current most discriminative feature is the dependency path. To make the feature space more discriminative, we can make these paths more discriminative, by adding extra context outside the shortest dependency path. In our precision error analysis above, we see that lack of context broadly applies to most error types. A motivating example error case is in Example 8.

(8) **wrong relation:** (FMR Corp., `org:parents`, Fidelity)
**context:** A February filing by *Fidelity* parent firm *FMR Corp.* listed 1.065 billion in PetroChina shares as of Dec. 31.

This mention pair has the features [ ORG $\xrightarrow{\text{nn}}$ firm $\xrightarrow{\text{nn}}$ ORG ] and [ ORG firm ORG ]. This is not enough context for labelling, as this feature space does not capture any notion of parent. We only know is that there is some kind of ORG firm-based relation. This lack of context for making a correct slot labelling is particular significant where modifiers to the path negate or restrict the relation. This maybe as simple as a literal not, or as in the case above an adjective which specifies the relation between

the NES: if the adjective was instead child the situation would be reversed. To discriminate between these cases, the modifier is required. However, simply adding these modifiers to all individual features would likely substantially disconnect the graph. Adding modifiers to trigger words alone results in a feature set sparser than dependency paths. In this case of Example 8, [ ORG $\xrightarrow{\text{nn}}$ firm $\xrightarrow{\text{nn}}$ ORG ] occurs 95 times in the corpus. [ ORG parent-firm ORG ]—arguably a straightforward construction—appears only once.

Specific terms can remain sparse even in huge amounts of text, and training data rarely provides labels for sparse examples. [ ORG firm ORG ] itself, for example, does not appear in training data for `org:parents` or `org:children`. Nevertheless, as with other design decisions we have made for the graph, adding more information to the graph is preferable as it at least allows the algorithm to discriminate between instances. We now discuss adding these features to the larger graph.

Adding modifiers to the dependency paths with this amount of data will result in a feature space that is much, much sparser than our original space. However, we still want to incorporate these features to allow us to discriminate between instances at all. To do this, we extend our current feature space: we add a modified path feature, where we add all modifiers (outside the shortest dependency path) to each token on the shortest dependency path, including the endpoints; we also add we add a modified trigger features, which adds the modifiers to a trigger word (but only those modifiers outside the shortest dependency path). Example 8 generates the features in Table 6.6. Note that in this case, no token has multiple modifiers, if it did, these would be included as separate features (for each modifier). Examples of these new feature nodes are included in Figure 6.3, again we have removed some nodes for brevity. Note that the top mention pair node is connected to the modifier nodes (with former modifiers), and the bottom is not.

Statistics for this graph including these two new feature types are shown in Table 6.7. A large number of feature nodes are added, although the increase in

| type | feature |
|---|---|
| path | [ ORG $\xleftarrow{\text{nn}}$ firm $\xleftarrow{\text{nn}}$ ORG ] |
| modded path | [ ORG $\xleftarrow{\text{nn}}$ firm-parent $\xleftarrow{\text{nn}}$ ORG ] |
| trigger | [ firm ] |
| modded trigger | [ firm-parent ] |
| triggers | [ firm ] |
| modded triggers | [ firm-parent ] |

Table 6.6: Modded features for Example 8.

| counts | filtered | with triggers | % increase | with modifiers | % increase |
|---|---|---|---|---|---|
| entity pairs | 3,275,446 | 7,459,555 | 128 | 7,459,555 | 0 |
| mention pairs | 9,249,592 | 22,357,171 | 142 | 22,357,171 | 0 |
| features | 1,007,978 | 1,208,523 | 20 | 5,663,298 | 369 |
| total nodes | 13,533,016 | 31,025,249 | 129 | 35,480,024 | 14 |
| edges | 37,780,554 | 207,296,600 | 449 | 298,713,700 | 44 |
| seeds | 9,059 | 15,037 | 66 | 15,037 | 0 |

Table 6.7: Graph profiles for graph with trigger features and graph with modifier features, compared with *filtered* with MIML-RE data from Chapter 5.

number of edges is less drastic (but still substantial). Our recall upper bound does not change as this was already accounted for by the trigger features—all we are adding here is the potential for an algorithm to discriminate between instances. Results for this graph are in Figure 6.4, and make almost no change to the previous results. This is likely due to these nodes being sparse, and hence having fewer connections and little influence over the graph. As the weights as implemented are insufficient, future work should consider removing nodes that are too general, either by filtering the feature nodes that make up the graph, or handling this as part of propagation.

We note that while MAD allows for the relabelling of seed nodes, no such relabelling occurs in our label propagation process. This is due to the relatively sparse distribution of seed labels in our graph configurations: there are no seeds

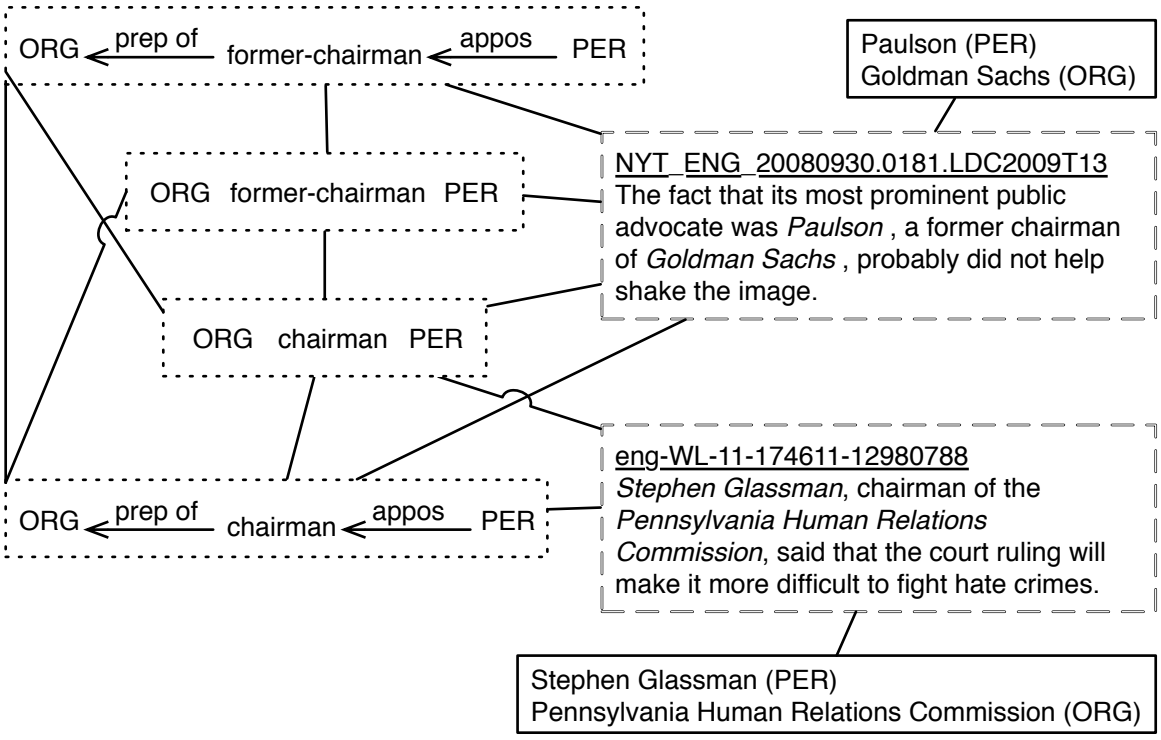



Figure 6.3: Graph with additional modifier features.

adjacent to each other,and there is no consistently labelled region with enough weight to relabel a seed. The hyperparameter $p_v^{\text{inj}} = 1.0$ is 100 times $p_v^{\text{cont}} = 0.01$, and as seeds are at least two edges apart, a seed would need at least in the order of 10,000 conflicting seeds (two edges away) to relabel that seed. We simply do not have enough seed nodes for this to occur.

We have now explored increasing the detail in the graph, to allow a label propagation algorithm to potentially discriminative between cases. However, there has been actual little improvement in results. Inspecting the graph for this analysis, we find that many errors occur in very close proximity to seed nodes. This leads us to step away from considering the graph topology to look at another source of error: error caused substantially by close proximity seed nodes due to conflicting annotations, and we analyse this in the next section.

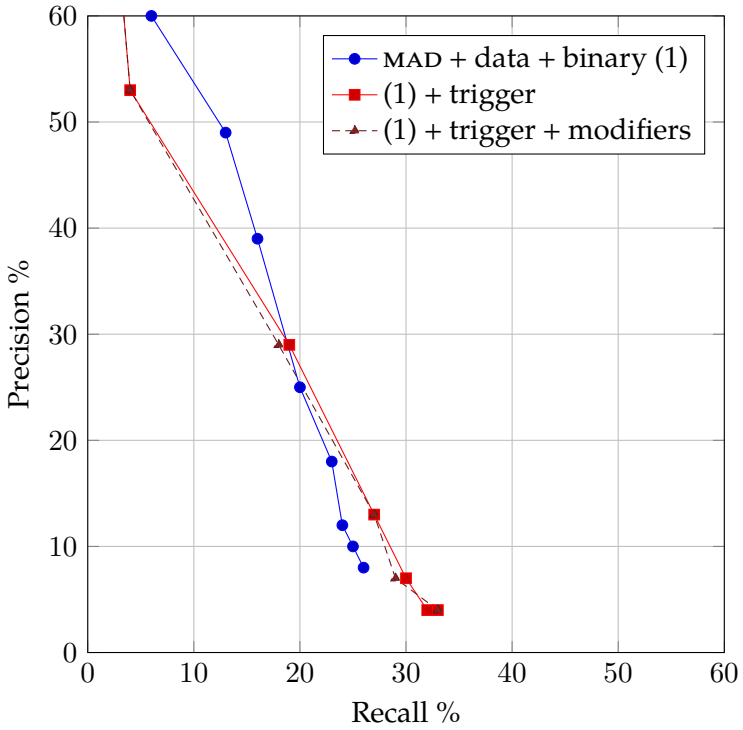



Figure 6.4: PR curve for label propagation. Comparison of earlier results with addition of modifier features.

## 6.4 Annotation analysis

While inspecting the graph and results for the previous experiments, we find that many errors occur in close proximity to seed nodes. This is concerning, as because if errors are occurring near to seed nodes, it suggests that there may be an underlying issue with how nodes are connected: close nodes are meant to be semantically similar in the construction of the graph. If close nodes are incorrect, the chances of getting distant nodes correct is reduced as this iterative process relies on keeping semantic drift low. Upon further inspection of the graph, we find that while there are definitely cases where lack of discriminability of the representation causes propagation across incorrect instances. In cases such as the parent firm example in the previous section, paths without modifiers allow for semantic drift, and label distributions can be substantially conflicted even one node away from

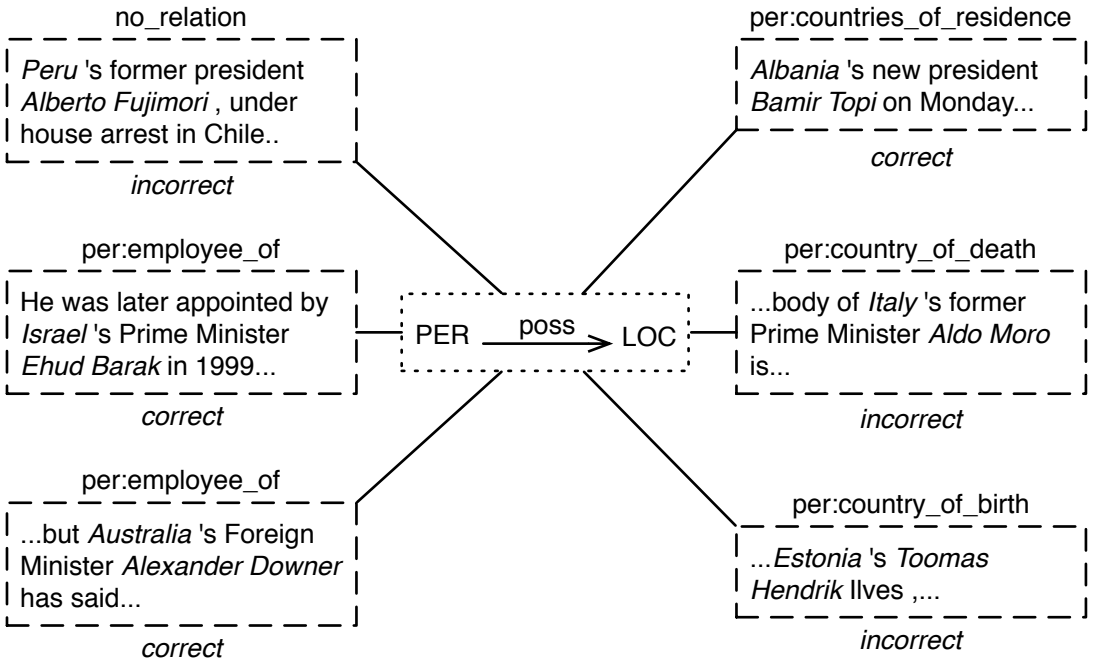



Figure 6.5: Example seeds attached to $[\ \text{PER} \xrightarrow{\text{poss}} \text{LOC}\ ]$. Nodes are marked as *correct* if they are validly annotated seeds, otherwise are marked as *incorrect* (for reasons discussed in text).

seeds. Figure 6.5 contains a small sample of seed nodes attached to $[\ \text{PER} \xrightarrow{\text{poss}} \text{LOC}\ ]$. Note several different labels that immediately result in a conflicted distribution on the feature node. We want to look at this problem in more detail, and begin by inspecting these nodes adjacent to seed nodes.

For this analysis, we are continuing to use the MIML-RE annotations (Angeli et al., 2014) as the main source of training data in our graph. In this data, annotations only have one label: limiting RE to a maximum of one label per instance is a near-ubiquitous simplifying assumption in the literature (Riedel et al., 2010; Hoffmann et al., 2011; Surdeanu et al., 2012). However, nodes connected to a number of these seeds may immediately be assigned a distribution of conflicting labels. Effectively, this results in very immediate increase in label ambiguity on the first iteration, and this only increases on further iterations. Where this initial conflicted labelling occurs for a high-degree node, a conflicted distribution will

be substantially propagated through the graph. This typically results in whatever slot happens to be most frequent on high-degree nodes—typically the slots those which are themselves most frequent over the whole task—dominating a particular region of the graph or even the whole graph. Exploring this problem, we look at the conflicted distribution of two high frequency dependency paths in detail.

### PER-LOC possessive

We consider the seeds which attach to the dependency path feature $[\,\text{PER} \xrightarrow{\text{poss}} \text{LOC}\,]$, some of which are shown in Figure 6.5. This path is extracted from a construction such as LOC's PER, e.g. *Italy*'s *Simone Corsi* in Example 9. This node is the second highest degree node in our graph. Note that this is a useful feature, that can provide strong support for particular slots, especially `per:LOC of residence` slots.

(9) Meglio was overall champion on 264 points, followed by *Italy*'s *Simone Corsi* on 225 and Gabor Talmacsi of Hungary on 206.

Its attached seed distribution is in Table 6.8, and again a small number of these examples are in Figure 6.5. This node has 201 attached seeds split over 8 different labels. Many of these are for `per:employee of` and `per:countries of residence`, which is a large conflict, but the number of different labels is of significant concern.

We summarise the source of these seeds. `per:employee of` annotations are generally for government positions, such as Example 10, and PER roles appear to always be mentioned somewhere in context. This can simply be as a job title, e.g. president, or a more distantly referred to role, such as a list of heads of government.

(10) **relation:** (Jean Eyeghe Ndong, `per:employee of`, Gabon)
**context:** *Gabon*'s Prime Minister *Jean Eyeghe Ndong* earlier denied reports that the president had died on Sunday.

| seed | count |
|---|---|
| per:employee of | 89 |
| per:countries of residence | 83 |
| per:country of birth | 11 |
| no relation | 7 |
| per:stateorprovinces of residence | 6 |
| per:origin | 2 |
| per:cities of residence | 2 |
| per:country of death | 1 |

Table 6.8: Seeds attached to [ PER $\xrightarrow{\text{poss}}$ LOC ].

Some per:employee of seeds are incorrect due to type errors, especially for bands, e.g. the band Sugarland being annotated as a LOC. Many per:* of birth annotations appear to be incorrect given the context of the sentence, typically indicating residence, not birth, as in Example 11. Even though this relation is a true fact, it is not possible to derive this from the context.

(11) **wrong relation:** (Amir Khan, per:country of birth, England)
**context:** *England*'s *Amir Khan* owns the World Boxing Association crown but Alexander called out another rival, Tim Bradley, after the victory rather than make a case for a unification showdown.

per:* of residence is problematic for the single-fill assumption, as national-level government employment is an acceptable justification for residence (as in Example 12), as is employment for U.S. state senators and state representatives.

(12) **relation:** (Stephen Harper, per:country of birth, Canada)
**context:** *Canada*'s Prime Minister *Stephen Harper* says the United States should not reopen talks on the North American Free Trade Agreement as the two U.S. Democratic presidential hopefuls have proposed.

per:origin annotations for this slot are incorrect, fills are Finland and Venezuela, but these should be demonyms, and per:origin should probably not actually be considered as a name slot as part of the TAC KBP definition.

All the no relation annotations here are incorrect, as a relation does exist in all of these cases. These are all from conflict with the per:* of residence issue above, as in Example 13. They are all either national or U.S. state employees, which is a valid inference to make for the task (including that this considers a *former* president) and should be a valid seed for the graph.

(13) **wrong relation:** (Suharto, no relation, Indonesia)
*Indonesia*'s former president *Suharto* died from multiple organ failure on Sunday, a local police official told reporters at the hospital where he was admitted on January 4.

The per:country of death case is a interesting incorrect annotation, as the sentence itself is correct, but requires inference beyond the mention pair or the path. In Example 14, per:country of death is filled via inference from Rome, not Italy.

(14) **wrong relation:** (Aldo Moro, per:country of death, Italy)
1978 - The bullet-riddled body of *Italy*'s former Prime Minister *Aldo Moro* is found in parked car in central Rome, 54 days after his abduction by Red Brigade terrorists.

Based on this analysis, once annotation errors are resolved, conflict on $[\,\text{PER} \xrightarrow{\text{poss}} \text{LOC}\,]$ is between per:employee of and per:* of residence. We expect that $[\,\text{PER} \xrightarrow{\text{poss}} \text{LOC}\,]$ always indicates per:* of residence. However, it may or may not indicate per:employee of, as they must be a national or U.S. state employee for this to be true. If this is the only feature for a pair, then we cannot discriminate between the two cases, to allow for this, *national or U.S. state employee* needs to be encoded in the graph.

| seed | count |
|---|---|
| org:founded by | 11 |
| org:top members/employees | 9 |
| no relation | 1 |

Table 6.9: Seeds attached to [ ORG $\xrightarrow{\text{poss}}$ PER ].

This cannot simply be modelled by label interaction, because if we did not have the per:* of residence slots at all, then we would still need to model both these cases: no relation and per:employee of seeds could otherwise both be validly attached to this node, and whichever was dominant would dominate local mention pairs. We note that resolving the problem in the annotation is required before being able to address this distinction.

### ORG-PER possessive

While some of the previous error in the previous example were clearly annotation errors, there is a substantial amount of conflict where slot differences are subtle. The adjacent seed distribution feature for [ ORG $\xrightarrow{\text{poss}}$ PER ]—PER's ORG, e.g. *Warren Buffett*'s *Berkshire Hathaway* in Example 15—is in Table 6.9.

(15) *Warren Buffett*'s *Berkshire Hathaway* will receive a license in New York to open a new bond insurance business, a state regulator said Friday.

This distribution is problematic for the slots org:top members/employees and org:founded by, and the difference in how these slots are expressed can be very subtle.[2] Consider Example 16: what slots for the query Virgin Group does Richard Branson fill? This is ambiguous given this sentence only as context. No amount of local features can certainly disambiguate between these slots for the seeds attached to this node.

[2]Note that org:shareholders could also be generally be problematic for this path, although is not present in our training data.

(16) Universal Music Group said Friday it has reached a tentative agreement to buy V2 Music Group Ltd, a unit of British billionaire *Richard Branson*'s *Virgin Group* and home to recording artists such as Sterephonics and Paul Weller.

Potentially, we can solve this on a global level: an entity pair may be mentioned elsewhere in the corpus, and will have its own distribution informed by perhaps more discriminative dependency paths such as [ ORG $\xleftarrow{\text{prep of}}$ founder $\xleftarrow{\text{appos}}$ PER ] or [ ORG $\xleftarrow{\text{dobj}}$ led $\xleftarrow{\text{rcmod}}$ PER ]. But it remains the case that no local representation for this conflicted node will provide a distinction between the labels. This particular error is a substantial problem in the MIML-RE data, where annotators have likely included too much of their own world knowledge when making annotation decisions. This annotation issue is a critical problem: if we cannot produce useful distributions adjacent to seeds then we have little hope of propagating useful distributions across the graph. We explore this exactness problem in detail in Chapter 7. Before we address this issue, we first want to briefly consider justification of SF extractions, as this is also related to the problem of annotating which instances express which slot.

## 6.5 Fill justification

To this point, we have been using `anydoc` evaluation, where a justification span for a fill is not required as long as the fill text itself is correct. We now consider using strict evaluation and not allowing `anydoc`. Using `anydoc` is reasonable (and follows prior work (Surdeanu et al., 2012)) for SF evaluation performed after the actual running of the shared task: otherwise, a correct fill returned by our system will only be marked as correct if a system participating in TAC 11 outputs the same specific instance of a fill. Where correct fills are mentioned across many documents, choosing between which of several equivalent instances to report may be arbitrary, and this results in not all instances of the same fill being reported in the evaluation

data. Example 17 gives an example of a correct fill in the evaluation data, and Example 18 gives a fill that is not present, but these are both correct fills. In this particularly notable (but not unique) case, the sentences are exactly the same!

(17) **relation:** (DirecTV, `org:top members/employees`, Neal Tiles)
**document:** `eng-NG-31-105524-11957201`
**context:** On September 19, 2005, it was reported by TVweek.com that former *DirecTV* executive *Neal Tiles* had replaced G4 founder Charles Hirschhorn as the channel's CEO.

(18) **relation:** (DirecTV, `org:top members/employees`, Neal Tiles)
**document:** `eng-NG-31-100788-10908825`
**context:** On September 19, 2005, it was reported by TVweek.com that former *DirecTV* executive *Neal Tiles* had replaced G4 founder Charles Hirschhorn as the channel's CEO.

In addition, while redundant fills are marked in results and we can account for those, fills which are obviously redundant—such as those that exactly match KB entries—are not returned by any system and are not in the results at all. This may be problematic for our analysis, as these are often top-ranked results. However, justification is an important part of the task—if a fill is not justifiable it is incorrect—and we should experiment with not allowing `anydoc` to provide us with a very strict evaluation. We now explore adding this justification requirement. Note that our system may find non-redundant correct fills that no other TAC 11 system found, but neither justification setting will include these instances as they are not in the evaluation data or KB.

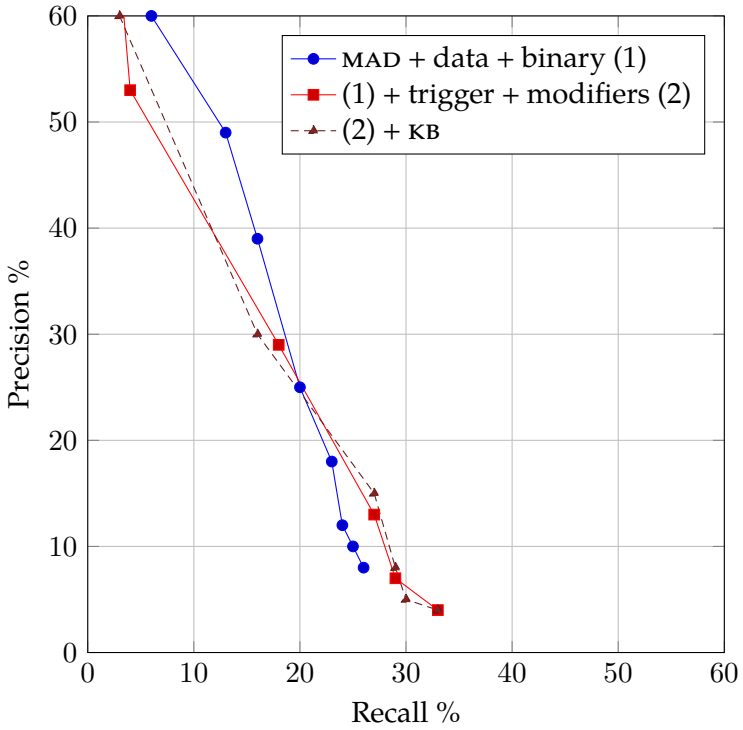



Figure 6.6: PR curve for label propagation. Comparison of results with KB filtering.

### 6.5.1 KB filtering

Before we switch from an `anydoc` evaluation, we first need to consider a small number of instances which are not justifiable (i.e. are incorrect) not because of their context but because they are redundant with the KB. Our analysis in Section 6.2.2 has a substantial number of instances which appear correct, but may be redundant with the KB. We need to extend our pipeline and evaluation to include two components: KB redundancy and identification of inexact or redundant fills.

We adapt our analysis to include fills marked redundant and inexact in the evaluation data. To address instances that no system returns because they are in the KB, we add an additional filter which prevents output of fills which directly match KB entries. There are 36 SF queries in TAC 11 that have corresponding KB entities (the remaining 74 queries are for entities not already in the KB). Implementing such a filter is non-trivial. The TAC KB is not a direct mapping to slots, even with a manually defined mapping from infobox attributes to slots. While an attribute like

(key people, Hajime Sasaki) for an organisation is trivially mappable to org:top members/employees, slots such as (deathplace, Little Rock, Arkansas) require more unpacking into different per:LOC of death slots. This primarily occurs for slots filled by locations—of the 44 location fills for entities already in the KB, 10 of these contain a comma. In total, 50 non-date entries with a comma exist for this query set. To simplify this analysis, we manually normalise these entries along with the 19 entries that contain a newline. We then filter out candidate fills that exactly match these KB fills. Results using the KB filter allowing anydoc evaluation are in Figure 6.6. These results are very similar to the recall upper bound, which is not surprising as there are very few instances filtered out: only one correct instance and eight incorrect instances are removed. However, this configuration now considers particular instances, and allows us to evaluate without anydoc.

### 6.5.2 Document-level justification

We now evaluate without anydoc, and these results are shown in Figure 6.7. By performing this evaluation, we can identify where evaluating directly against the test data is inconsistent with the anydoc evaluation, in terms of instances that our system would get correct and incorrect if it had of participated in the actual shared task. We see a marked decrease in recall, as instances that were previously marked as correct are now incorrect. The top F1 is 16%. To get a better idea of the types of errors identified by the different evaluation, we again perform our top 100 ranked analysis. We find confusion between per:employee of and per:member of substantially affects precision, likely because the MIML-RE annotations were not strict on the per:employee of/per:member of split; there are annotations which are actually per:member of which should be per:employee of.

Of the 69 incorrect fills in the top-100 ranked fills, 11 of these errors are where one of these two slots was a correct fill but the other was incorrect; two were where one of these slots looked to be correct but was not in the assessment results; a

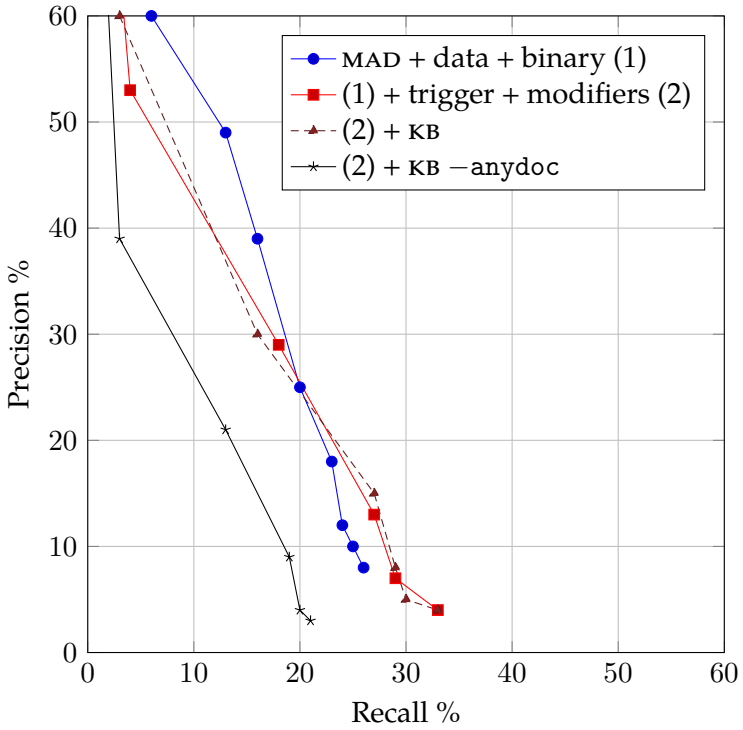



Figure 6.7: PR curve for label propagation. Comparison of earlier results without `anydoc`.

further ten were cases that were incorrect in both cases. In total, almost half (33 of 69) of our errors are due to this confusion. 24 cases are genuine errors, the largest easily identifiable categories being five NER typing errors and six cases of the general features in our model overgeneralising.

The remaining fourteen of these errors are cases which look like they should have been annotated as correct. At the best F1, getting these correct would be an extra 20% recall, even at the recall upper bound this would be an extra 12% of instances. Again, this is only the apparently correct results in a small sample, and so missing fills appear to be a substantial problem when using the evaluation data. This, alongside the issues with the MIML-RE data, lead us to undertake a detailed analysis of annotation issues in the next chapter.

While we will not go much deeper into the other error cases, we do want to address the issue of distinction between `per:employee of` and `per:member of`. We have three options for improving the discrimination between these slots:

1) Collect additional training annotations on cases difficult to split.

2) Use a post-graph filter to assign one of the two slots, similar to the location gazetteer filters. What form this would take is unclear, but this could make use of a gazetteer to filter context by job title. This particular solution would only apply for fills with a job title in context, and generating these gazetteers is non-trivial.

3) Modify the evaluation to use the combined slot. The slot was merged from TAC 2013 onwards, and so it may be appropriate to merge for evaluation.

A post-graph filter would not allow us to make use of the graph to learn appropriate fills, and the first option is the more appropriate, still valid solution. Changing the schema is changing the task, and distinguishing between fill types is part of the task, even if the distinction is subtle. As we want to ensure that any subtle annotation decisions we make are consistent, this also leads us to the work in the next chapter.

## 6.6 Summary

In this chapter we have explored reasons for low performance in the graph, particularly as related to sparsity and discriminability. Substantial recall is lost due to the initial construction of our graph. Profiling the graph, we found 12-16% of nodes are not included in any of the main typed subgraphs, and these nodes are in small, disconnected subgraphs without any seed nodes. To better connect our graph, we add a trigger word feature, a more general feature derived from the discriminative dependency paths. This drastically increases connectivity in the graph, a 449%

increase in the number of edges providing a 7% increase in reachability, making reachability equal to the upper bound.[3]

We then turned our attention to precision errors, and identify that while dependency paths are very discriminative, they are actually not discriminative enough for a number of instances. Even if this context is very sparse, we still want to represent it in the graph, and so we introduce modifier features. These new features, on both paths and triggers, add syntactic modifiers that were originally outside the shortest dependency path to every feature. Considering a more sophisticated selection process for adding these features is a direction for future work. In this chapter, we added all features to the graph, but removal of overly general features, replacing them with more discriminative features only is possibly a better model, allowing for more discriminative features to have greater influence over the graph.

The remainder of this chapter was concerned with issues of applying annotations and justifying both manual and automatic annotations. We found that a large number of errors occur in very close proximity to seeds, and identify that this as major concern for propagation. We contributed a detailed analysis of some of these nodes, e.g. [ PER $\xrightarrow{\text{poss}}$ LOC ], which has 201 annotations in the MIML-RE data over 8 different slots (including `no relation`). There are some fundamental conflicts caused by a lack of sufficient disambiguating context, but many of these conflicts are actually caused by subtle problems in annotation.

Both of these factors lead us to want additional annotated data, but we need to investigate concerns about annotation consistency for slot filling, and we explore this in Chapter 7. In particular, we want to ensure that annotation of subtly different slots and contexts—the most difficult examples—remain as consistent as possible. Large numbers of different slots annotated to the same context makes discriminating between slots very difficult. However, making annotation consistent appears may be difficult for slot filling, and we explore this in the next chapter.

[3] With the exception of ORG shareholders, for which we have no training data

# 7 Relation explicitness

Understanding how a relation is explicitly expressed in text is central to SF, and other RE tasks, where explicit contextual justification of an extraction is required. Problems arise in both training and evaluation if it is difficult for human annotators to decide whether this justification actually exists.

After attempting to annotate more data to improve the label propagation results in Chapter 6, we discovered a large disagreement between annotators when it came to assigning negative labels. Disagreement between positive labels—that is, every label except for the `no relation` label—was rare, but deciding whether a label or `no relation` should be assigned was much less clear. In Example 1 (which will discuss in detail below), deciding whether there is a `per:LOC of residence` relation between Billy Mitchell and Wisconsin comes down to how much an annotator believes a relation is explicitly expressed, as a probability based on the context expressing the relation.

(1) A member of one of Milwaukee's most prominent families, *Billy Mitchell* was probably the first person with ties to *Wisconsin* to see the Wright Brothers plane fly.

This uncertainty is symptomatic of the ambiguous fundamental goals of SF. The original goal for SF was to complete a Wikipedia-style KB, but this is an abstract target for annotation and evaluation. Overall, Wikipedia is relaxed on standards for infoboxes. Most facts in infoboxes are not cited, and have no actual justification.

For the most part, this is sufficient for humans to make use of this data. However, other downstream applications may required higher standards. A highly sensitive financial or legal application may only want very explicit extractions. Applications might opt for a more relaxed approach, such as for providing suggestions when partially automating the writing of emails. TAC itself is the most recent, detailed schema for a RE task. Much consideration was put into the TAC and earlier MUC and ACE definitions of relations, but none of these sources really address how explicit relations need to be.[1]

In this chapter, we design and carry out several exploratory annotation tasks. These annotation tasks are all quite small, but we are not attempting to create a new dataset: we are exploring and demonstrating fundamental problems. These problems are revealed on even on a small amount of data, and a larger dataset would generate more examples, but will not change the fundamental issues. These annotation tasks are exploratory, as they are either not practical to implement on a useful scale, or would slow the annotation process so much that the yield would be too small for training for the full task.

Our first contribution is to provide a more concrete definition of explicitness, particularly as this notion of explicitness is somewhat distinct from typical annotator disagreement. We require an annotator to use a strict definition of explicitness when re-annotating part of the Angeli et al. (2014) data, a dataset annotated with a more general definition of slots. We also contribute a substitutability criterion for this kind of explicit annotation, based on ideas from lexical semantics.

We next analyse a number of categorisations for degrees of explicitness with which a particular slot fill is expressed in text. We propose to minimise this disagreement by creating a more structured annotation task, based on decomposing sentences into small decisions: small facts and probabilistic inferences that we expect an annotator needs to be confident in, to make a decision. Finally, we ask

[1] MUC and ACE definitely focused on short-ranged relations such as prepositions, which are often more explicit, but for longer range relations this is undefined

annotators to rank sentences that all express the same fact, based on how confident they are that the fact is expressed. This reveals that annotators tend to be internally consistent in their confidence within particular categories of decisions, but the relative confidence of a category may differ. We use this to contribute guidelines for those evaluating SF and RE, as well as guidelines for future task designers and annotators that should be considered for consistent, meaningful evaluation.

## 7.1 Motivation

We return to Example 1 from the introduction, repeated in Example 2. Determining whether this relation is expressed is outside the scope of the TAC schema, as it is neither certainly accepted nor certainly rejected.

(2) **relation:** (Billy Mitchell, `per:LOC of residence`, Wisconsin)
**context:** A member of one of Milwaukee's most prominent families, *Billy Mitchell* was probably the first person with ties to *Wisconsin* to see the Wright Brothers plane fly.

As an additional complication, different annotators may explicitly or implicitly incorporate their world knowledge into this decision. Whether or not this is appropriate for the task, it will ultimately lead to differences in annotation. If an annotator is not aware that Milwaukee is in Wisconsin in Example 2, their confidence that a relation is expressed is likely to be substantially lower.

Approaches which do not use human-annotated training data, including purely distant supervision-based approaches, still require some notion of explicitness, particularly for evaluation.

This notion of explicitness has not been explored by relation extraction tasks. Typically, systems derive relation definitions from KBs in an ad hoc fashion, often simply based on the name of the fact type. For example, the popular NYT cor-

**3.6 PER: City of Birth**

***Content:*** Name
***Quantity:*** Single
***Description:*** The geopolitical entity at the municipality level (city, town, or village) in which the assigned person was born. This slot must be filled with the name of a city, town, or village.

- Hong Kong, Macau, Gaza, and Jewish settlements should be classified as cities.
- Capitol Districts (e.g. Washington D.C.) should **NOT** be classified at the city level, rather they should be classified at the state or province level.
- GPEs below the city level (e.g. 5 boroughs of New York City) are **NOT** valid fillers.

| Entity | Document Context | Correct Filler |
|---|---|---|
| Hank Williams | Williams lived in Georgiana in the mid 1930's with his mother, Lillie, and his sister, Irene, after his birth in Mount Olive West | Mount Olive West |
| Tom Lehman | Lehman was born in Austin, Minnesota but … | Austin |

Figure 7.1: Definition for `per:city of birth`.

pus evaluation used by Riedel et al. (2013) makes use of Freebase relations such `/people/person/place_lived`, but their actual evaluation is typically either on a simple interpretation of *a place in which a person lived*, or a direct comparison with the KB tuple without regard for whether a mention is valid or not. Often both types of evaluation are performed, but they may not agree, and in particular, may not be internally consistent or reproducible given different human evaluators. Annotators for one evaluation may mark Example 2 as correct, and others as incorrect, because there is no consistent expectation of explicitness.

TAC SF provides relatively extensive slot definitions, which we covered at length in Chapter 2. However, these still provide a definition that is primary about ontology—the types of entities that fill a particular slot—rather than about evidence for a slot. Consider the `per:city of birth` example from Chapter 2, repeated in Figure 7.1: there is discussion of types, but no further discussion of how a person can be considered *born* in a city (the document context examples themselves are typically very short-range, straightforward and explicit instances). This slot definition is more about what constitutes the entities types (the city) rather than the concept or event of birth.

Within SF definitions, there is no scope for measuring how explicit a relation actually is in a sentence, or whether it is useful for broader inference about an

entity. Even given this detailed definition, there can still be a substantial difference between what relations annotators consider to be explicitly defined.

This lack of consideration for explicitness has a significant impact on the evaluation of relation extraction techniques, most notably where manual evaluation of results on held-out data is used: without calibration of how explicit relations need to be across annotations, results may vary substantially. In a task where performance improvements come in small increments, evaluation differences will produce significantly different outcomes. Specific detail on what explicit justification is required for adjudication is not covered in the extraction literature (Mintz et al., 2009; Surdeanu et al., 2012; Riedel et al., 2013).

## 7.2 Defining explicitness

As covered in Chapter 3, early work in relation extraction, as a task derived from MUC-7 and others (Brin, 1998; Agichtein and Gravano, 2000) was directly inspired by the problem of constructing KBs. This basic framing has continued through to current TAC KBP. Relations themselves are derived from instances in KBs: in MUC-7, *general relational objects* are derived from templates; the work of Brin (1998) on book-author pairs defines the criteria for pairs as *given a potential author and title and where they are mentioned on the Web, a human can generally tell whether this is a legitimate book* (for the given author). ACE describes a *reasonable reader rule* where relations are annotated only if there are no reasonable interpretations of the sentence where a relation doesn't hold, but this is not further defined (LDC, 2005). This same basic justification idea carries through to distantly supervised approaches (Mintz et al., 2009), which are trained directly from KB instances aligned to text. TAC KBP has developed detailed descriptions for annotating particular slots, but these are still ultimately derived from Wikipedia infoboxes.

Almost all settings require human annotation or adjudication for evaluation, and a meaningful annotation requires an understanding of how these relations are actually defined. Annotation guidelines for SF make an effort to emphasise that the relations need to be explicitly expressed. The assessment guidelines state that *if a filler cannot be justified solely by the justification strings or their surrounding context, it should not be labelled as correct* (Li et al., 2013). In particular, the requirement for annotators and systems to not rely on external knowledge as justification for extractions is well documented.

Even with this documentation, annotation decisions are still difficult, and in general can substantially change based on subtle differences in an annotator's mental model.

### 7.2.1 Similarity of problem to distant supervision

We note that the typical approach of deriving relation types from KBs is essentially the same as distant supervision: tuples from a KB are aligned to text to create training data. Fundamentally, dealing with this explicitness problem is very similar to dealing with the distant supervision assumption, as we need to ask how we know if an aligned tuple is actually expressed or not?

This also relates to the idea of only one-sense-per-discourse for tuples, as part of the distant supervision assumption. Distant supervision assumes that KB tuples can be aligned to all matching pairs of entities. As we discussed in Chapter 3, this is a very rough approximation, and the largest source of annotation error is not confusion between different relation labels, but rather whether the relation is actually expressed. This matches our definition of the explicitness issue: giving annotators a weak description of explicitness means that they are trying to align a given tuple to a sentence without knowing if it should actually be aligned.

## 7.3 Strict explicitness

The TAC SF task requires a very strict level of explicit justification for a fact to be judged correct, by analysts who have been highly trained to look for evidence and who are critically adjudicating output of SF systems.

On the other hand, a typical native speaker happily accepts facts with a much lower burden of explicit proof when comprehending text. What a human may uncritically believe are facts is typically of far larger scope than would be extracted for TAC. Because of this, actual fact extraction in SF is somewhat unnatural, and annotators have to be trained to stop their usual loose comprehension mechanisms from applying when judging slot fills. A reader may read Example 3, and believe that FirstGroup is based in the U.K.—which is true—but for the purpose of slot filling this relation is not actually explicit.

(3) **wrong relation:** (FirstGroup, `org:country of headquarters`, U.K.)
**context:** *FirstGroup* is the largest bus operator in the *U.K.*, operating a fifth of local bus services.

In this section, our contribution is to quantify the effect that a change in explicitness—a strict versus a casual reading—has on the task of SF and RE more broadly. TAC requires explicit justification of facts from a local context, but humans build knowledge at different levels of evidence: ranging from the *local context* of a sentence, including the document read so far, or the whole document; the application of wider *world knowledge* (knowledge about specific named entities) and *commonsense knowledge* (general knowledge about non-specific entities); and simple *inference* through to complex non-monotonic *reasoning*. In many cases, how a reader uses probabilities to infer relations is particularly critical. Probabilistic inference typically applies to commonsense knowledge, e.g. how likely is it that someone lives in

the location in which they work?[2] This human reading represents as a gradient, from accepting only fully explicit facts without inference, to full inference from implicit or probabilistic evidence.

The level of justification is highly dependent on the NLP application. For instance, relation extraction over legal documents may require very strict justification, whereas a personal assistant for recalling information from email should mimic finding the facts the human reader is likely to have extracted.

The MIML-RE data (Angeli et al., 2014) is closer to this more casual human reading, particularly regarding the use of background knowledge, as the crowdsourced annotators are not trained to use a very precise mode of reading, nor an explicit standard for justification. For example, given the sentence and questions in Example 4, annotators were asked to select from the top five most likely slots (as determined by the active learning process):

(4) In 1993, GD Searle withdrew from *India* and sold its holdings to *RPG Group*.
Which option below describes the relationship between *India* and *RPG Group*?

These slots are expressed as sentences, such as *RPG Group* is a subsidiary of *India* and *RPG Group* is headquartered in the country of *India*. Annotators may also manually enter a custom relation or no relation. This annotation is the largest-scale annotation of TAC types, and has been effective in the state-of-the-art and top-performing SF approaches in general.

In this section, we aim to quantify the effect a change in explicitness—a strict versus a casual reading—has on the task of SF by re-annotating a portion of this MIML-RE annotation with a strict explicitness requirement.

[2]This particular question changes greatly on whether location is a city, state or a country, and also on the particular entities that are involved. It is more possible that someone working on the border of the U.S. state of New Jersey could live in a another state than it is someone working in the Australian island state of Tasmania.

### 7.3.1 Annotation

The Stanford MIML-RE data (Angeli et al., 2014) contains 33,748 sentences from the 2010 and 2013 TAC KBP source documents, as well as a July 2013 dump of Wikipedia. Sentences are selected using an active learning-style strategy, selecting instances calculated to be most useful for improving performance of their MIML-RE approach to RE (Surdeanu et al., 2012). Sentences estimated to be moderately difficult to classify are emphasised. Cases which are relatively straightforward or very difficult are not selected for annotation. The actual annotation is over a single pair of named entities (NEs) in each sentence. An instance is annotated with either a single slot type, or `no relation`. Note that while SF slots are used, the task is treated as a traditional RE task where a relation is extracted for a single pair of arguments and not otherwise aggregated.

To streamline our re-annotation, we continue to filter the instances to only include annotations over pairs of PER, ORG and LOC NEs, based on the slot types in the original annotation. From these sentences, we randomly select 1,000 instances to re-annotate. The sentences that make up this dataset were selected as interesting by the active learning process in Angeli et al. (2014): our goal is to characterise the effect of a change in explicitness across the whole data set set, so we randomly select examples for re-annotation. One expert annotator annotates all 1,000 examples. The annotator was able to choose from any slot in the TAC 2014 definition of slots, as well as `no relation` if no slot is valid, or `error` if the instance is not valid (if the sentence is not in English or the NE bounds are invalid). The annotator was permitted to assign multiple slots to a particular instance if appropriate. We reduce LOC relations (city, state/province and country) to a single slot, following our work in Chapter 5.[3]

[3]This is also to simplify the annotation overhead. The difference between these granularities is minimally important for this experiment, and reduces the number of labels to choose from.

After re-annotation, we identified that the per:employee of and per:LOCs of residence slots frequently occurred on the same instance. This is due to SF allowing the inference that national and U.S. state-level employees are also residents of their employing LOC/GPE. Example 5 is an instance of this for Khaleda Zia and Bangladesh.

(5) *Khaleda Zia* was sworn in as the Prime Minister of *Bangladesh*.

As the co-occurrence of per:employee of and per:LOCs of residence was the only case of multiple slots on a single instance, we treat them as a separate slot per:employee and residence in our analysis, and consider corresponding per:employee of instances to be annotated with this joint slot, so as not to penalise the original annotation for only having single slots annotated.

### 7.3.2 Substitutability test

Initially the purpose of our re-annotation was to evaluate the quality of the MIML-RE data for slot filling. We soon discovered that the only real quality/agreement issue was judging explicit justification, a problem we had seen numerous times when looking at our slot filling system output in Chapter 6. Example 6 shows such a case, being labelled as per:country of birth when there is no real explicit justification of this fact:

(6) **wrong relation:** (Amir Khan, per:country of birth, England)
**context:** *England*'s *Amir Khan* owns the World Boxing Association crown but Alexander called out another rival, Tim Bradley, after the victory rather than make a case for a unification showdown.

As our next contribution, we define a substitutability test to assist annotators in identifying whether a relation is actually explicit. This is similar to those used to make judgements in lexical semantics (Cruse, 1986), e.g. in the sentence He plays

the violin very well., substituting fiddle for violin doesn't result in a sentence with a different truth condition. In this case, this is a test for synonymy, demonstrating fiddle and violin are synonyms. We use a slightly different approach, substituting entity "antonyms"—entities of the same type but different referent—to see if the relation apparently expressed by the sentence is not longer true.

In Example 7, the original annotation contains the relation (Mahmoud Ahmadinejad, `per:employee of`, Iran). While it may be reasonable for a human to make this probabilistic inference from this sentence, this relation is not actually explicit. There is no explicit connection between President and Iran.

(7) *Iran* launched its first space research center on Monday in Tehran in presence of President *Mahmoud Ahmadinejad*, the official IRNA news agency reported.

Our substitutability test says that, if the slot fill can be replaced with a different NE of the same type and the sentence or context is still plausible (e.g. doesn't break known facts about the world), then an annotator should not annotate the relation.

In Example 8, replacing Mahmoud Ahmadinejad with Barack Obama makes it clear that the relation is not actually expressed without world knowledge about the NEs:

(8) *Iran* launched its first space research center on Monday in Tehran in presence of President *Barack Obama*, ...

The reason we can substitute Barack Obama here is that Iran is not explicitly linked to the presidency. If it was, then the substitution would break known facts about the world, as in the following case:

(9) *Iran* launched its first space research center on Monday in Tehran in presence of its President *Barack Obama*, ...

As it is often possible to contrive sentences that have different interpretations, we put the additional constraint that the relation between other pairs of NEs in the sentence remain the same when substituting a NE, e.g. the relation between Iran and Tehran remains the same. If we do not follow this the substitution tends not to be useful, e.g. substituting London for *Iran* creates a somewhat odd sentence:

(10) *London* launched its first space research center on Monday in Tehran in presence of President *Mahmoud Ahmadinejad*, ...

Additionally, it requires that the substitution be focused on the type of the entity rather than the specific entity. If an annotator knows that Barack Obama has never visited Iran, and uses this specific fact to justify the context as being implausible, this is a poor choice of entity and a more general *president* or *person* would be more suitable.

Our substitutability test is a useful thought experiment to evaluate whether explicit justification exists for a relation. It is important to note that it cannot be used where there is more than one valid slot fill for an entity-relation pair. Using this substitutability test makes the annotation task a somewhat more complicated manual process. However, it provides annotators with a useful rule-of-thumb to actually make a consistent annotation decision regarding explicitness.

### 7.3.3 Results

Of the original 1000 instances, 688 instances are annotated the same in the re-annotation, resulting in a Cohen's kappa of $\kappa = 0.64$ measured between the original annotation and the re-annotation (treating each annotation as a distinct single annotator). 130 agreements are on `no relation`, and 258 disagreements include `no relation`. Counts for the extra and missing `no relation` instances are shown in Table 7.1.

The remaining 54 disagreements are distributed across other slots, with most disagreement on per:LOC of birth and per:LOCs of residence (11 instances); per:employee of and per:LOCs of residence (7 instances); and per:LOC of birth and per:employee and residence (6 instances).

The original annotation contains 161 no relation annotations, our re-annotation contains 357 no relation, an extra 20% of instances. Ignoring disagreement in these instances gives $\kappa = 0.92$, as they make up the bulk of the actual disagreement.

### 7.3.4 Discussion

Many of the differences in annotation are due to systematic errors that are resolvable using an expectation of explicitness and the substitutability test. The slot which contains the most instances re-annotated as non-explicit, per:LOCs of residence, is made up primarily of examples like Example 11, where there is an indication of presence in a location but not of actual residence:

(11) **non-explicit relation:** (Anthony David, per:LOCs of residence, Atlanta)
**context:** *Anthony David* was born in Savannah, but got involved in the music business in *Atlanta*.

We can substitute any other city for Atlanta because David could have been involved in the music business there, showing it doesn't explicitly justify residence. Ultimately, this and many other relations come down to a probability—in this case, that working in a city indicates residence—and because a probability is a factor this relation cannot be considered explicit. The slot org:LOC of headquarters follows a similar pattern, e.g. Example 12, indicating an organisation's presence in a location but no actual evidence for headquarters.

| slot | - | + |
|---|---|---|
| org:alternate names | 7 | 0 |
| org:founded by | 6 | 0 |
| org:LOC of headquarters | 45 | 3 |
| org:member of | 5 | 2 |
| org:members | 1 | 0 |
| org:parents | 5 | 3 |
| org:political/religious affiliation | 0 | 0 |
| org:subsidiaries | 0 | 3 |
| org:top members/employees | 11 | 1 |
| per:alternate names | 9 | 0 |
| per:children | 4 | 0 |
| per:employee of | 35 | 14 |
| per:LOC of birth | 11 | 0 |
| per:LOC of death | 3 | 0 |
| per:LOCs of residence | 78 | 3 |
| per:other family | 0 | 0 |
| per:parents | 1 | 1 |
| per:schools attended | 0 | 0 |
| per:siblings | 0 | 0 |
| per:spouse | 6 | 0 |
| per:employee and residence | 0 | 0 |
| error | 0 | 1 |

Table 7.1: Disagreement on `no relation`. - indicates a slot in the original annotation with a corresponding `no relation` in the re-annotation, and + indicates the reverse.

(12) **non-explicit relation:** (United Biscuits, `org:LOC of headquarters`, UK)
**context:** The Jacob's brand is owned by Valeo Foods in Ireland and *United Biscuits* in the *UK*.

The slot `per:employee of` is interesting because the conflict is in both directions, i.e. it is the slot that had most additional positive labels in the re-annotation. The Iran example, reiterated in Example 13, is a typical example of an instance re-annotated as non-explicit, where background knowledge of NEs is required.

(13) **non-explicit relation:** (Mahmoud Ahmadinejad, `per:employee of`, Iran)
**context:** *Iran* launched its first space research center on Monday in Tehran in presence of President *Mahmoud Ahmadinejad*, the official IRNA news agency reported.

Cases where our annotator has added a label are typically where a person is a member of an organisation, such as in the following:

(14) **explicit relation:** (Anthony Koutoufides, `per:member of`, Carlton)
**context:** In a tight final quarter, *Carlton*'s *Anthony Koutoufides* starred and Fraser Brown made a match-saving tackle on Essendon's Dean Wallis ...

Membership is part of the `per:employee or member of` slot in TAC, but it is unclear how this was defined in the original annotation which only references `per:employee of` (in our annotation we keep these slots separate, and so this may be an artefact of our annotation process). `org:top members/employees` has similar reasons for conflict, being a similar inverse slot.

Many of the `per:LOC of birth` conflicts are due to relatively permissive labelling in the original annotation, e.g. Example 15. These entities are likely related in some way, but assigning `per:LOC of birth` to this context is a large leap:

(15) **non-explicit relation:** (Bob Marley, per:LOC of birth, Jamaica)
**context:** "*Jamaica* needs another *Bob Marley*, a leader for the people," said Prince Alla, 60, a contemporary of Marley and himself an influential reggae artist.

Outside the no relation label, the disagreements continue the pattern of larger probabilistic inference in the original annotation beyond what is explicit. In Example 16, there is an indication of per:LOC of residence, but not birth, but there is confusion between these labels:

(16) **disagreed relation:** (Peter Kane, per:LOC of birth, England)
**context:** *Peter Kane* (1918-1991) was one of *England*'s greatest flyweight boxers and a world champion in the 1930s.

### 7.3.5 Task implications

SF systems that make use of very large, less precise training data sets (created using distant supervision) typically have better F1 performance than corresponding approaches with smaller, more precise training data. One of the observations of Angeli et al. (2014) and the success of distant supervision in general indicates that a large amount of reasonable quality data generated quickly and relatively cheaply can be of huge benefit to performance. We expect that as recall appears to be harder to gain than precision in SF (Chapter 4), any approach which increases coverage is likely to improve performance overall, even if there are known precision problems with the data.

As we have discussed, humans do not often require the same explicitness requirements as TAC does, and may need less evidence to make use of information. To employ SF techniques we need to consider what applications can make use of the underlying setting of SF. It is important to note that changing the explicitness

required directly reflects the precision-recall trade-off: a higher expectation of explicitness increases the precision requirement of systems. Depending on what level of justification is required, a theoretically perfect SF system could still have precision-recall trade-off.

There are several potential options for structuring a task that takes different levels of explicitness into account. Annotating explicitness itself may be an interesting direction for future TAC shared tasks. Tasks could allow a looser definition of justification, but require that systems still provide justification and additionally indicate confidence in a particular degree of explicitness.

## 7.4 Degrees of explicitness

The strictest explicitness criteria is likely too stringent a definition for many applications. We now attempt to extend this definition by grading how explicit instances are, as opposed to only considering the strictest definition. To begin, we consider a few examples in detail. Note that we continue the requirement that both entities be mentioned in a sentence to even consider that a relation between them exists.

As in the previous section, we first consider instances where relations are clearly explicit or not explicit in text, where there is no ambiguity as to whether relations are expressed. Clearly non-existent relations in a sentence are those in which relations are not even referenced, such as a relation in Example 17; or where relations are (nearly) explicitly contradicted such as in Example 18:

(17) **wrong relation:** (Betty Smith, `per:LOC of residence`, Brooklyn)
**context:** The 1943 book A Tree Grows in *Brooklyn* by *Betty Smith* uses the tree of heaven as its central metaphor, using it as an analogy for the ability to thrive in a difficult environment.

(18) **wrong relation:** (Levy Mwanawasa, `per:employee of`, Democratic Republic of Congo)

**context:** Zambian President *Levy Mwanawasa* and his *Democratic Republic of Congo* counterpart Joseph Kabila are scheduled to hold bilateral talks Thursday in Zambia's copperbelt province.

The vast remainder of examples fall in between these two extremes, as evident from the comparison between our re-annotation and the Stanford annotations, with our annotation marking 20% more annotations as `no relation`. We attempt to categorise these instances that have disagreement. We have an annotator (different to the one that completed the initial re-annotation) categorise the instances that were annotated with a slot in the MIML-RE data but with `no relation` in the re-annotation. An ad hoc categorisation schema is built through this annotation. We attempt to separate instances in a way that helps us unpick reasons for disagreement. We develop a loosely defined 6-degree scale, and assign instances to these categories. The results are shown in Table 7.2, and provide counts of the number of each category in the data. The degrees in this scale roughly correspond to certainty in the relation.

Despite this scale, in post-annotation analysis and discussion we find that these categories are highly subjective, and rely on an annotator's linguistic and background knowledge, as well as their willingness to categorise something as a particular level of certainty. In this annotation, Examples 19, 20, and 21 are annotated as *reasonable*, *inferable* and *guessable*, but are difficult to actually separate and categorise for a more formal annotation task:

(19) **reasonable relation:** (Theodore Frelinghuysen, `per:LOCs of residence`, New Jersey)

**context:** He received his diploma from *Theodore Frelinghuysen*, *New Jersey*'s

| label | definition | counts |
|---|---|---|
| correct | We believe these instances should have been annotated with slots (disagreement with the original re-annotation). | 6 |
| reasonable | Instances that appear to be correct, but that fail the substitutability test. Very likely to be interpreted by a human as correct, except for this strictest reading. | 84 |
| inferable | It is likely that a fact is expressed, and a human would say that it is very likely but there is some uncertainty. Uncertain, but more likely than not. | 33 |
| guessable | A human would essentially be guessing from the context, but there is enough support to allow for this guess to be made. Essentially in the range of a 50-50 guess. | 35 |
| unreasonable | Essentially wrong, but not inconceivable that the relation could be incorrect. Uncertain, but unlikely, especially given the context. | 30 |
| wrong | Instances which are very clearly wrong due to structural issues or nearly direct contradiction in test. | 22 |

Table 7.2: Results for degrees of certainty, with counts of the number of those instances in the re-annotated data.

first major-party vice-presidential candidate, who had run unsuccessfully with Henry Clay in 1844.

(20) **inferable relation:** (Regent University, `org:founded by`, Pat Robertson)
**context:** *Pat Robertson*'s *Regent University* in Virginia Beach, once boasted on its website that scores of its graduates worked at the Bush White House.

(21) **guessable relation:** (Billy Mitchell, `per:LOCs of residence`, Wisconsin)
**context:** A member of one of Milwaukee's most prominent families, *Billy Mitchell* was probably the first person with ties to *Wisconsin* to see the Wright Brothers plane fly.

Example 19 depends on what it means to be a vice-presidential candidate of a state; Example 20 indicates some kind of ownership, but `founder` is unclear; and Example 21 requires a decision on what is means to be a member of a prominent family in a location, and whether that indicates `residence` in that location. Importantly, these examples all require some form of background knowledge and contain uncertain, probabilistic elements. Calibrating these examples is difficult using this simple scale.

## 7.5 Splitting by types of knowledge

As a second attempt at developing an annotation schema, we try to split instances on several variables, asking several questions:

- Is KB, or commonsense knowledge or linguistic inference (whether probabilistic or certain) required?
- Is a relation is expressed by one piece of evidence or multiple pieces of evidence joined by simple inference?
- Can the sentence be easily altered to strengthen or weaken a relation?

We find that there is no obvious way of categorising instances based on these questions, and hence no obvious way of structuring an annotation task, primarily because many of these instances incorporate all of these elements. Critically, we find that attempting to fully describe why annotation decisions are made is more complex than we anticipated. In the following examples we provide informal annotator comments for why a positive annotation is expected.

(22) **relation:** (Jake, `per:spouse`, Reese)
**context:** Jake , 28 , and Reese , 33 , held hands as they walked around with a salesgirl , looking at high-end lamps and other home accessories , an observer

tells Star .

**why:** Holding hands indicates a relationship. Ages are important to indicate not a parent-child relationship, but age is probably not so important here, because of the rest of the context. They are in the age zone for marriage. There's a sense of looking for stuff for their first home, honeymoon period/newlyweds feel from the sentence.

(23) **relation:** (John Howard, `per:LOCs of residence`, Sydney)

**context:** Australian former Prime Minister John Howard , who will meet the runners when they reach New York , paid tribute to the group on Monday as they were farewelled in Sydney .

**why:** Assumption that a former PM would pay tribute in person. There are different possible cases: e.g. a death tribute could be paid anywhere; runners are likely to have been in media where they are. Using world knowledge, New York is not in Australia, most former PMs live in the country they governed. On the balance of probabilities, hangs on whether the tribute is in person or important enough to fly somewhere for. Probably not significant enough to fly there.

(24) **relation:** (BYD Company, `org:LOC of headquarters`, China)

**context:** Founder of BYD Company , he is the wealthiest man in China as of late 2009 .

**why:** The wealthiest man in China probably lives in China, and so a company they founded (particularly what appears to be their main business) would likely be in China. This is particularly the case because any very wealthy company in China specifically is likely to be Chinese.

These instances highlight a complex mental process on the part of the annotator, which is difficult to formalise in an annotation schema. As we cannot easily define

a schema for splitting these instances, we attempt to use a small annotation task to gather more information about how annotators interpret explicitness of instances.

### 7.5.1 Annotating confidence

We have 5 annotators annotate a small number of examples, each instance being composed of a sentence and a relation. We ask annotators to score each example on a scale of 1–10 for how likely they think it is that the relation is true after reading the sentence. We do not further calibrate the scale, as we want them to read as a typical casual reader and not be pedantic in analysis (we emphasise this as part of the task). Annotators score 50 examples. While this is not a large number, we immediately get a demonstration of the explicitness problem.

We calculate the mean and standard deviation ($\sigma$) for each instance: 2 have $\sigma > 4.0$, 6 have $\sigma > 3.0$, 14 have $\sigma > 2.0$, 30 have $\sigma > 1.0$. We select 7 examples of `per:LOC of residence` for discussion in Table 7.3. This contains both high confidence examples with low standard deviation (#1) and with significantly outlying scores (a score of 3 in #2), and instances which either are all of middle confidence but low deviation (#4 and #6), or have wide deviation (#3, #5 and #7).

In discussion with annotators we find that, as in our previous analysis, annotation decisions vary greatly over concrete background knowledge (i.e. knowledge found in a typical KB), general commonsense knowledge (e.g. the probability that the founder of a company being resident in that company's location in #4) and the willingness of an annotator to make probabilistic inference from evidence present in text. #2 requires an probabilistic inference about candidate nomination. #3 requires, among other decisions about likelihood of having lived with one's family, an understanding of who was born in Hungary.[4]

In #5 an annotator scoring 2 did not know that Milwaukee was in Wisconsin, and later indicated that this knowledge would substantially change their annota-

[4]Interestingly, in the full document, this refers to Simmons' *mother*, not Simmons

| # | sentence | scores | $\sigma$ | mean |
|---|---|---|---|---|
| 1 | The motion was put forward by the Prime Minister in reaction to an announced motion by Bloc Québécois leader **Gilles Duceppe** that would recognise *Quebec* as a nation, but did not contain the words "in Canada" . | 8 7 8 10 8 | 1.10 | 8.2 |
| 2 | He received his diploma from **Theodore Frelinghuysen**, *New Jersey*'s first major-party vice-presidential candidate, who had run unsuccessfully with Henry Clay in 1844. | 9 7 3 9 9 | 2.61 | 7.4 |
| 3 | **Simmons**' father, Feri Witz, also Hungarian-born, remained in *Israel*, where he had one other son and three daughters. | 9 3 8 3 10 | 3.36 | 6.6 |
| 4 | Recently, **Redford** founded the Redford Center in *California* to use the arts to push issues like clean energy. | 6 6 6 6 6 | 0.00 | 6 |
| 5 | A member of one of Milwaukee's most prominent families, **Billy Mitchell** was probably the first person with ties to *Wisconsin* to see the Wright Brothers plane fly. | 2 8 7 6 2 | 2.83 | 5 |
| 6 | In *Detroit*, Michigan he was introduced to **Tamia** by singer Anita Baker. | 4 6 3 5 6 | 1.30 | 4.8 |
| 7 | –1999: Returns to *Indonesia* following **Suharto**'s ouster and begins a campaign for the imposition of Islamic law. | 2 9 5 8 2 | 3.27 | 5.2 |

Table 7.3: Selected casual annotation examples for `per:LOC of residence`. Sorted by mean annotator. Query entities are **bold**, fill entities are *italic*.

tion. We expect this lack of annotator knowledge to be a problem for annotation. While it can be difficult for annotators to exclude real-world knowledge that is not actually expressed in the text, it is impossible for them to include knowledge they don't have. It the case of #5, this knowledge may make a relatively strongly expressed fact appear to be quite weakly expressed. Annotators could look up every mentioned entity in existing KBs for more information about entities, but this would substantially slow down the annotation task, and is beyond how the task is typically defined. Additionally, this process would not cover commonsense knowledge that annotators may or may not actually have.

#6 and #7 again rely on willingness of annotators to make inference. Commonsense knowledge and willingness to make inference typically relate to probabilities.

In #7, an annotator must decide how likely it is that a person being *ousted* from a country (in the context of another person *returning* to the country) was a resident of that country. In this particular example, the additional political context suggests that the person may have been a local politician, but there is no guarantee and so the annotator has to make call based on probabilities on whether to annotate this example. Even in this case when it is not a binary decision, assigning a high or low confidence score is still very annotator dependent.

## 7.6 Structured annotation

This casual annotation experiment provided some useful insights into annotator decision making, but does not actually provide any more consistent annotation. We explore providing annotators with a more structured approach to make decisions about annotation confidence. Possibly, we can do this by breaking down examples into a series of small decisions which are easier to quantify, as a way of making individual annotation decisions simpler and providing additional scaffolding to capture context that annotators may miss. This ideally would include some way of incorporating or separating background knowledge.

Following this idea, we breakdown the examples in Table 7.3 as we did informally in Section 7.5, but try to formally structure this process, primarily by attempting to structure each decision as a series of simpler decisions.

As an example, we walk through our original example:

(25) **relation:** (Billy Mitchell, `per:LOC of residence`, Wisconsin)
**context:** A member of one of Milwaukee's most prominent families, *Billy Mitchell* was probably the first person with ties to *Wisconsin* to see the Wright Brothers plane fly.

| # | arg1 | rel | arg2 | type |
|---|---|---|---|---|
| 1 | prominent families | `possessed by` | Milwaukee | direct |
| 2 | Billy Mitchell | `(one of a) member of` | prominent families | direct |
| 3 | Billy Mitchell | `ties to` | Wisconsin | direct |
| 4 | Milwaukee | `located in` | Wisconsin | KB |

| # | inference | type |
|---|---|---|
| 5 | #4 implies (`residence`, *city*) | uncertain |
| 6 | #2 + #1 implies (`residence`, *city*) | uncertain |
| 7 | `city in state residence` implies `state residence` | certain |

Table 7.4: Formal breakdown of Example 25.

We identify simple facts or decisions that contain either entity, with the idea that we want the facts or decision points to be made up of short sequence of text or dependency path, to be mediated by a single noun or verb, and are directly extractable from text. These *direct* decisions we identify for Example 25 are #1-3 in Table 7.4. We identify that the KB-style knowledge in #4 may be useful. To extract the relation, we can use one of two uncertain inferences to connect Billy Mitchell to Milwaukee. Either #5, that `ties to` implies `residence` (combined with *certain* inference #7 over #4 that `residence` in a city implies `residence` in that city's state, and *uncertain* but likely possibility); or #6, that being a `member of` a prominent family of a city implies `residence` of that city. Of course, an annotator may combine these inferences to potentially have more confidence in the relation than either inference individually.

If we can get annotators to consider #1, #2, #3, #4 and #7, and assign expectation probabilities to #5 and #6 (or even derive these from a KB), then we might be some way to making annotation decisions more consistent. If we can some structuring the annotation process in this way we may be able expose elements that annotators would have otherwise overlooked.

In addition to the types of sub-relations and inference in Example 25, in working with examples in Table 7.3, we identify two other elements that annotators may use when making annotation decisions. The first of these is *topic*: knowing that there is a Quebec sovereignty movement (or even that there is a possibly that one could exist given real-world constraints) may help annotate the first instance, even if no actual sub-relations directly encode this context. The second element, which we now explore, is a concept of small compound decisions which appear to be part of the decision-making process. If we breakdown Example 26 we get the small decisions in Table 7.5:

(26) **relation:** (Elena Paparizou, `per:LOC of residence`, Greece)
**context:** In 2006, with Greece hosting the Eurovision Song Contest after Elena Paparizou's victory with the song "My number one", he once again tried to take part in the Greek final as a composer, this time for Anna Vissi.

Note that this is a particularly complex case, and requires several inference steps if an annotator is to be confident about an extraction. *Compound* decisions are those which are derived from other decisions, such as one decision relying on an event having already happened (e.g. #3). Confident annotation likely requires knowledge that a country that wins Eurovision hosts it the next year, that people typically compete for their country, and this is what the Greek final refers to. These more world knowledge aspects, #6 and #8, may not be very generalisable. However, this process does allow us to break down a complicated construction.

### 7.6.1 Annotation task

This breakdown may be possible, but it raises two questions. Does this sub-relation breakdown represent decisions users are actually making in general, or is there disagreement even at this low level? And, we can try to be comprehensive in this

| # | arg1 | rel | arg2 | type |
|---|---|---|---|---|
| 1 | Greece | `hosting` | Eurovision Song Contest | direct |
| 2 | Elena Paparizou | `victory with` | the song "My number one" | direct |
| 3 | #1 | `after` | #2 | compound |
| 4 | Elena Paparizou | `tried to take part in` | the Greek final | direct |
| 5 | #4 | `as a` | composer | compound |

| # | inference | type |
|---|---|---|
| 6 | `hosting` Eurovision requires `victory` | certain |
| 7 | #6 `victory` refers to #2 `victory` | uncertain |
| 8 | Elena Paparizou sounds like a Greek name | uncertain |

Table 7.5: Attempt at a formal breakdown of Example 26.

breakdown, but how do we know if we've missed background knowledge? Note that while this breakdown may be useful to present to annotators, it is not a process that can be easily automated. However, this process is useful at least for analysis of the annotation process.

To address these ideas for this breakdown, we design a small annotation task based on `per:LOC of residence`, with 10 examples derived from our initial annotation experiment (such as the examples in Table 7.3). We ask the same annotators to add the same confidence scores as earlier, but in this setting we profile annotators with a small set of comprehension-style questions to explore the sentence. The idea behind the questions is not to bias annotators in one way or another, but simply to explore the sentence to both clarify their decision-making process and to perhaps give us some indication of why they have made a particular annotation.

For example, for Example 27 we ask the questions in Table 7.6.

(27) **relation:** (John Howard, `per:LOC of residence`, Sydney)

**context:** Australian former Prime Minister *John Howard*, who will meet the

| questions | 1 | 2 | 3 |
|---|---|---|---|
| Your score (1-10) | 5 | 6 | 2 |
| Is the relation obvious upon reading the sentence? | no | no | no |
| Is the tribute in person? | yes | yes | maybe |
| Is the tribute event significant enough for John Howard to travel to Sydney? | probably | yes | maybe |
| Did you use any other information to make your annotation? | I tried not to, but knowing that John Howard has lived in Sydney probably made me even more literal when interpreting this sentence | no | no |
| Your final score (1-10) | 3 | 6 | 2 |
| Why did your score change? | the sentence doesn't really say one way or the other whether John Howard was residing in Sydney | - | - |

Table 7.6: Questions for Example 27 with annotator responses (annotators marked as 1, 2 and 3 in the table).

> runners when they reach New York, paid tribute to the group on Monday as they were farewelled in Sydney.

Annotator 1's change from 5 to 3 in this example is the biggest change in this experiment, with most annotations never changing score. Of the 30 annotations (3 annotators over 10 examples), 22 had scores which remained the same; 5 had an increase of 1; 2 a decrease of 1; and 1 a decrease of 2.

It is surprising that scores were not more varied after the questions were posed, despite scores still being quite different between annotators. This suggests that annotators may be confident in their own mental model, or that they have already fully comprehended the sentence when making annotation decisions. Identifying differences in the mental model of annotators is important for being able to more precisely structure the annotation task.

## 7.7 Ranking explicitness

To compare annotator decisions, we set up a final task focused on a single fact, as opposed to asking annotators to annotate a range of facts. For this task, we ask annotators to rank sentences which refer to the marriage of Prince William and Kate Middleton, specifically the relation (Prince William, `per:spouse`, Kate Middleton). We choose to use this relation as we expect all of our annotators to have reasonably similar world knowledge about these particular entities, and the entities and relation are relatively unambiguous.

We want to find relevant sentences that are similar in meaning and source domain. To select these sentences, we extract all sentences from ClueWeb12 that include OpenIE-style relations between William and Kate.[5] From these 227 instances, we select 10 sentences that have reasonably different forms: we want to make the task reasonably straightforward, and in any case do not expect meaningful differences between cases like Examples 28 and 29.

(28) Prince William and Kate Middleton were pronounced husband and wife, and that's all anyone can seem to talk about today.

(29) Prince William and Kate Middleton were pronounced husband and wife Friday, after five months of breathless hype and anticipation for Britain's royal wedding, a ceremony that was expected to be watched by as many as two billion people.

We have four annotators rank these sentences by how confident they are that the sentence expresses the relation. The sentences we select are in Table 7.7, ordered by annotator 1's ranking (sentences were presented to annotators in random order). These are mostly sentences where the explicitness is unclear: while #8 explicit ex-

[5] openie.allenai.org/search?arg1=William&rel=&arg2=Kate&corpora=cw

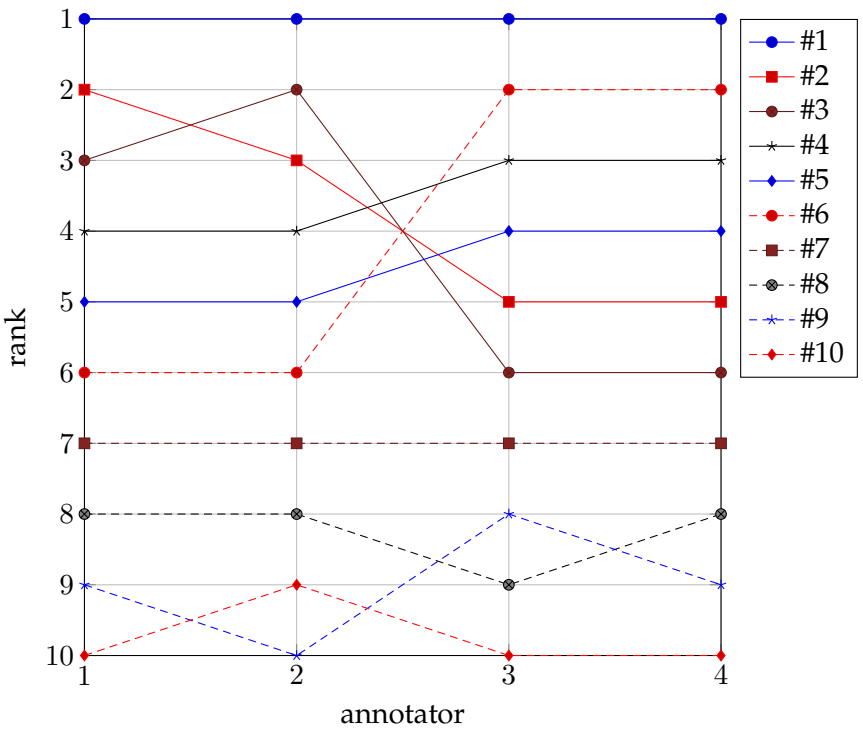



Figure 7.2: Visualisation of sentence re-ranking annotation for four annotators. Each line represents a sentence from Table 7.7.

presses the relation, and #10 doesn't really express the relation at all, the remainder fall somewhere in between.

Annotator ranks are visualised in Figure 7.2. Sentences #1 and #7 are consistently assigned the same rank. #8, #9 and #10 are of consistently low annotation. Interestingly, annotator 2 does assigns #9 to the lowest rank, instead of #10. It appears that this annotator has taken the indication of some kind of relationship as of higher importance than the context of both entities being at a wedding. However, they have higher confidence in #8 which would appear to require the same particular probabilistic inference.

#4 and #5 are similarly ranked, this is not surprising, as they both hinge on Duke and Duchess of Cambridge. #2 and #3 are also given similar ranks, again likely determined a single phrase, in this case exchange their vows. The biggest disagreement, is the relative position of these two sentences with respect to #6. It is

| # | sentence |
|---|---|
| 1 | Kate Middleton married Prince William on 29 April 2011 at Westminster Abbey, London, England. |
| 2 | The throne room at Buckingham Palace is being turned into a post-wedding relaxation zone after William and Kate exchange their vows for revelers who need to recharge during the day and evening festivities. |
| 3 | Cheers erupted among the hundreds gathered in the pre-dawn darkness in Times Square some decked out in wedding dresses as William and Kate exchanged their vows. |
| 4 | It's hard to believe that a whole year has passed since Prince William and Kate Middleton became Duke and Duchess of Cambridge at London s Westminster Abbey. |
| 5 | Prince William and Kate Middleton are now the Duke and Duchess of Cambridge. |
| 6 | At 11 am on the 29th April 2011, Kate Middleton walked down the aisle with Prince William. |
| 7 | Kate Middleton did her own make-up for the wedding because she wanted William to see her how he always will, not how someone made her up to look like. |
| 8 | I LOVE that moment when Kate Middleton walks down the aisle and Prince Harry turns back to look and tells Prince William "Wait 'til you see her !". |
| 9 | As Kate Middleton walked down the aisle, Prince William kept his gaze dead-ahead. |
| 10 | Kate Middleton made her first solo appearance while Prince William is in the Falklands, wearing an affordable gray Jesire dress and four-inch heels . |

Table 7.7: Sentences to rank for (Prince William, `per:spouse`, Kate Middleton).

interesting to note that we might expect this to be similar to #9, but walking down the aisle *with* the other entity is of particular importance. It appears that annotators 1 and 2 weight exchange their vows higher than walking down the aisle with. Ultimately, this ranking appears to be a result of annotator willingness to accept linguistic inference, i.e. use non-literal language as evidence for expression of a relation. There are blocks of certainty over which there is not much disagreement—although there may be substantial variation within these small blocks—based on particular probabilistic factors in context.

## 7.8 Discussion

These exploratory annotations have demonstrated both the effect of a lack of explicitness requirements in SF and RE annotation, and the multitude of ways annotators interpret the explicitness of a sentence. Use of world knowledge, commonsense knowledge, willingness to make inference, and interpretation of probabilistic components of context all influence annotator confidence. This is noticeable often in complex sentences, but also in quite simple constructions as annotators make substantially different decisions based on these factors.

These differences are somewhat due to the abstract nature of the TAC SF task. We note that TAC has notionally moved away from Wikipedia as a KB in later years, but the definition (and concept) of slots as well as the overall structure of the task is certainly derived from Wikipedia, and if anything moving away from Wikipedia further removes TAC SF from a real-world grounded application. Whatever the case, annotators do not particularly have a target in mind when annotating or adjudicating isolated instances.

The question is then, what can be done about this issue? Full annotation of a dataset with explicitness in mind may be worthwhile, and this would allow systems to target particular levels of explicitness. Such a dataset would likely be very tied to a specific application, as a schema is tied to a particular application and this affects measures of explicitness simply by how abstract relation definitions are. What may be more immediately useful is to expand the evaluation process of TAC: instead of simply assigning a simple mark for each instance, if annotators could provide confidence in the certainty of an instance—or provide extra reasons for accepting or rejecting particular results—this could go along way to better defining where justifications are correct. This could be done as analysis after the evaluation: offsets are provided with the results, and so a different set of annotators could analyse these datasets and potentially derive a clear explicitness target.

In the long term, much of this comes down to definition of the problem. At the very least, task designers need to discuss explicitness requirements, and provide detail as part of a task description. TAC specifically needs to incorporate more examples of instances that are too vague to be extracted, and a better defined boundary of what is explicit and what is not explicit enough for each slot.

There will always be edge cases in a schema, but at the moment the edges make up many of the instances being considered instead of being a few outliers. This could be helped by being driven by real-world end goals. If the end goal is to have as high coverage as possible, then annotators could be allowed to use real-world knowledge via web search or similar to make decisions. This could be provided as part of the annotation process: annotators could be provided with KB information about all entities in a particular in context, so they definitely know that Milwaukee is in Wisconsin. At a more complex level, tasks could become even more focused on individual entities, and require use of any available resources on the web to make annotation decisions: this would make annotation a costly process—structuring annotation of instances to allow for external information to be used as justification is a huge task—but this would allow for more confident annotation for large-scale KB completion.

These are possibilities for future tasks. TAC SF continues to evolve in positive ways, and is still a fantastic platform for framing KBP research. Many of these difficulties relate back to a lack of recall. Currently, annotations sacrifice precision even in training data to gain recall, and low-cost, high-recall, low-precision, large-scale annotations allow for recall to be gained and are useful sources of data. However, if we can better focus annotation tasks on a particular explicitness end goal, we may be able to reduce the amount of data required or enable annotators to be more efficient in producing useful training examples. An interesting future task would be to annotate a new data set following the principles in this chapter,

and existing approaches evaluated on this data set to measure in the impact of these principles on the RE task.

## 7.9 Summary

There is substantial disagreement in SF annotation, as well as in RE more broadly. As opposed to other semantic tasks, deciding between positive labels does not appear to be a key difficultly. Instead, the primary difficultly seem to be identifying whether a slot fill exists at all. Deciding whether a relation is explicitly expressed in text is central to SF. While it is typically straightforward to identify that an extraction *may* be possible, is difficult for human annotators to decide whether a justification actually exists.

Our initial annotation task provides our first contribution, and an indication of the scope of this explicitness problem. For this task, we require that slot fills are only annotated if they meet strict explicitness requirements. This is driven by a substitutability rule of thumb, where we ask annotators to exclude knowledge about specific entities but still make extractions relevant to the type of entities. This initial experiment motivates the rest of this analysis, as there is substantial disagreement on whether to assign a `no relation` label: a Cohen's kappa of $\kappa = 0.64$, where the remainder of annotation decisions has $\kappa = 0.92$.

This initial task is small, but we use this to motivate an exploration of this issue. The remainder of the chapter contributes different ways to frame the concept of explicitness, and offers a very in-depth look at the evidence annotators are using to make annotation decisions. We move beyond a harsh strictness definition, and after finding that degrees of certainty are hard to annotate reliably and do not appear to be useful, we move to annotator confidence. These annotation experiments reinforce the idea that confidence is inconsistent, to a 7-point difference on a 10-point scale in the most disagreed cases.

In our final annotation task, where we ask annotators to rank sentences that all express the same fact, by the confidence that they express this fact. We find annotators tend to be internally consistent, but rank different types of evidence differently, e.g. how willing they are to make inference from non-literal text.

We use this to motivate ideas on designing guidelines for those evaluating SF and RE, and hope that this work will encourage better defined annotation tasks, particularly the formal definition explicitness suitable for particular real-world applications. We hope that this will lead to more consistent data and evaluations.

# 8 Conclusion

Slot filling (SF) is a critical task for automatically constructing knowledge bases (KBs). KBs are a hugely valuable resource of information, and a structured format makes key facts within free text more easily accessible, and provides a consistent presentation of information. More importantly, a structured format makes data available for further machine processing.

These structured KBs support a wide variety of valuable applications. This includes web search; sophisticated question answering (which is made widely available within digital personal assistants); and for fact checking of news articles and other documents. Domain-specific KBs allow for further applications. Biomedical NLP uses structured data to identify trends in biological events, supporting medical research. Financial applications use data derived from news articles and company documents to support both rapid decision making and mining of long-term trends in financial markets. Personal productivity applications include using information extracted from emails, such as extracting event details into a calendar, or contact details from emails.

KBs need to have huge amounts of data is they are to be useful for any particular application. Human curation of facts into a structured format is is limited by human response times, reading times, and ability to collate large amounts of source data (documents such as news articles and web pages) into a set of facts. Humans cannot curate structured facts from more than a relatively small number of documents, particularly when response times are critical (as for financial applic-

ations) or where large amounts of data are required (as for general purpose QA). Automatic processing is not limited in this way, and is used to process these large amounts of unstructured data into a structured format, whether fully automatically or as support to human curators who make final adjudications on what should be extracted. In turn, this data can be used in these sophisticated downstream applications.

## 8.1 Thesis overview

This thesis has considered *slot filling* (SF). A *slot* is a named attribute, such as `per:city of birth`. A *fill* is a value of an attribute for a given entity, e.g. for the entity Mia Farrow the fill for `per:city of birth` is Los Angeles. Slot filling involves extraction of these attributes of entities from a large corpus of documents, for the purpose of creating or expanding a knowledge base (KB). Relation extraction (RE) is a core component of SF, and SF is typical framed as a query-oriented RE task. However, we have reflected on how SF is a much larger task, and a substantial amount of error in the task actually occurs outside this RE component. This is most notable in the NLP pipeline, when extraction of candidate fills results in a substantial amount of recall loss, before RE even begins. We have argued that this recall is a major limiter on performance.

How to represent relations is a major consideration for the RE component of SF. A representation needs to be discriminative enough for complex relations, but needs to be not too sparse otherwise it will not be useful. We have surveyed how the definition of this representation has influenced SF, and explored in detail how different specificity of the feature space divides representations of particular relation contexts.

We explored defining a representation of the task which reflects the behaviour and assumptions we want to model, in the form of a graph-based label propagation

approach. We explored a number of facets of defining this model. This gave us insight into the issue of representing relations, as the graph model allows us to get a measure of the impact of sparsity on both training and test data as well as the distribution of data as a whole.

The last issue we consider is the definition of explicitness: how explicit slots and relations need to be to be extracted. This is a substantial problem that has not been addressed, and results in a substantial amount of disagreement when trying to annotate data. Defining a standard of explicitness is ultimately defined by a downstream application, and is critical for consistent evaluation.

In this thesis, we have analysed an number of problems in the broad task of SF, ranging from significant loss of recall in TAC systems, to particular issues in the TAC task definition and SF and RE more widely.

## 8.2 Contributions

In Chapter 2, we detailed the task of TAC KBP slot filling. We established the foundation for the rest of this thesis, and identified a number of ways in which SF is perhaps more difficult than the related task of RE, and we begin to detail the differences between the two tasks. We continued this in Chapter 3, where we detailed approaches to RE, considering how relations are represented by contextual features. We surveyed literature for RE, how this task has been defined, and how the definition of this space has influenced SF. Additionally, we surveyed approaches to SF and how these approaches incorporate RE techniques, exploring some of the general difficulties of the task.

In Chapter 4, we contributed a detailed analysis of recall loss in SF systems. We argued that recall specifically is a major limiter on SF system performance. We precisely analysed where typical SF systems lose recall, and found that a substantial amount of loss occurs early in the SF pipeline, before RE approaches are applied.

We show reasons for low recall upper bounds: systems lose 12% recall by simply using NER to extract candidates, and lose an additional 8–24% recall dependency on what type of coreference resolution is used. We find that, using maximum recall bootstrapping, 39% of test slots fills are reachable from the training data, limited by an upper bound on non-unique paths of 43%, again highlighting that feature sparsity is of key concern for SF. We provided guidance to designers of systems in accounting for this loss. We expected this reachability technique would be potentially useful as an extraction approach, if sensible constraints were applied.

In Chapter 5, we follow up on this idea, designing a graph representation for SF and evaluating a label propagation approach to slot filling. We focus on creating a graph modelling behaviour and assumptions about the task derive from both the previous analysis and experimentation with the graph structure. Our baseline naïve slot filling pipeline results in an F1 of 9%; adding a set of rule-based filters over our graph structure slightly increased F1 to 10%. Actually integrating label propagation in the form of Modified Adsorption (MAD) increased F1 to 15%, and substantially increase the number of seed nodes in the graph, as well as adding a negative `no relation` label, saw F1 increase to 21%. Finally, we identify that allowing all slots to compete in the graph (as is standard for label propagation) limits performance in the task. For example, `per:parents` and `per:children` are inverses, and this should be accounted for in propagation. We categorise these relationships between slots, and map this categorisation to an interaction matrix. We then use this matrix to define a modification of MAD, which allows slots to interact in different ways as per constraints defined by the matrix. This gives us our best F1 of 23%.

We further profiled the graph model and results in Chapter 6. Profiling the graph, we found 12-16% of nodes are not included in any of the main typed subgraphs, instead being in small, disconnected subgraphs without any seed nodes. To better connect our graph, we added a trigger word feature, drastically

increasing connectivity in the graph, increasing the recall upper bound by 7%. We then turned our attention to precision errors, and identify that while dependency paths are discriminative, they are actually not discriminative enough for a number of instances. To account for this we added additional modifier features to the graph. We found that a large number of errors occur in very close proximity to seeds, and identify that this is a major concern for propagation, as very high-degree nodes often have a large number of conflicting attached seeds which are then propagated.

This seed error analysis led us to consider issues in annotation, particularly in how explicit relations need to be to be valid for extraction. Consistent annotation is critical for SF, and in Chapter 7 we undertook multiple annotation experiments that consider annotation differences. We identify that relation explicitness, as opposed to disagreement between labels, is of key concern. We analyse how this disagreement results from differences in world knowledge and willingness of annotators to make probabilistic inference. We propose a number of options for incorporating explicitness information in designing future annotation tasks, particularly in regards to considering explicitness in defining schema driven by real-world applications.

## 8.3 Future work

Firstly, we address elements of the thesis that could be directly improved. Many of the chapters in this thesis provide avenues for future work. Incorporating coreference resolution information in the graph representation in Chapters 5 and Chapter 6 is a promising extension, by linking mention pair nodes to entity pair nodes via coreference information. Named entity linking could also be used for the same purpose. In our experiments, adding coreferential mentions in this way resulted in the graph increasing to a size that was inconvenient to process, but with more resources or a more approximate representation this would be worth-

while, particularly for increasing recall. This would likely be a straightforward way for future work to improve on these results. Additional resources or approximate matching—perhaps by merging similar mention pair nodes or pruning low-frequency features—could also enable vastly increasing the size of the corpus. This would help reduce the disconnectedness by making particular contexts more frequent. Again, scaling the graph in this way is limited by available resources.

While we have extensively considered the definition of nodes in Chapter 5, we only considered normalised co-occurrence counts for edge weights. This is fairly standard for label propagation approaches, and how to weight edges is not something that has been experimented with in any great detail. However, we did not explore it at all here. Notably, our graph uses a number of different types of nodes, and experimenting with different edge weights across different types may allow for better modelling of this similarity. This possibly includes modifying edge weights based on the direction of propagation, e.g. it may be relatively more useful to propagate from a mention pair node to an entity pair node than vice versa. This could also potentially be integrated as part of a modification to the label propagation algorithm.

Next, we address ideas from our work that could be interesting avenues for future work. In Chapter 5, we modelled the interaction between slots as part of label propagation. The weights we defined for this interaction were derived from the definitions for slots, and were limited to hard constraints. However, these weights could be learnt, particularly for slots which are correlated but don't have any strict interactions (for which we did not include values in the matrix).

We experimented with adding both more general and more discriminative features to the graph in Chapter 6. While the general features were informative, adding all features to the graph without an informed selection process resulted in the discriminative features having little effect: the general features appeared to dominate the graph. Adding in all of these features was intended to reflect the

data as closely as possible. However, in this case the MAD algorithm was not able to leverage these additional features. Modifying MAD to better account for this kind of node distribution could be useful, either by adding one-direction propagation across particular edges, or adapting the entropy calculation to handle co-occurring features. A more sophisticated selection process for adding these feature nodes is a potential direction for future work. Removal of overly general features, entirely replacing them with more discriminative features, is possibly a better model. This would allow for more discriminative features to have greater influence over the graph, at the same time allowing a fallback to somewhat more general features. It would be interesting to explore using new semantic parsing techniques, such as Abstract Meaning Representation (LDC, 2017), to produce a different set of features for the graph.

Considering explicitness in task definition is important for future tasks, and much can be done to expand on our exploratory work in this space in Chapter 7. The experiment where we ask annotators to rank sentences referring to the same fact forms the most promising starting point. Expanding the scope of this experiment—with more facts across relation types, sentences, and annotators—will help to define the extent of the types of disagreements between annotators. Categorising larger types of disagreements should help to better define annotation tasks. TAC specifically should incorporate more examples of vague examples of fills as part of the schema. This would be helped by including a measure of annotator confidence in explicitness as part of the adjudication process, even if this were only useful for analysis at first (i.e. not used as part of scoring). Additionally, evaluating different explicitness requirements for different downstream applications would give a quantitative measure of annotation differences across these applications.

Finally, we consider future work that applies to the field more broadly. Our breakdown of the human process for annotating instances in Chapter 7 was useful for exploring the space, but was performed without a formal set of assumed

knowledge for annotators. This provides us with a few opportunities. The first is to experiment with including KB knowledge as part of the annotation process: e.g. provide a city's state or province to annotators when annotating contexts containing that city. For more abstract inference or probabilistic decisions, providing annotators with some form of aggregate statistics over a KB or commonsense knowledge-style statements (Angeli and Manning, 2014) (e.g. statements like it is likely the founder of a company lives in the country the company is based) may provide for a more consistent annotation.

## 8.4 Summary

This thesis has investigated the key difficulties in slot filling, particularly considering downstream use of KBs ultimately produced by SF. We have discussed the place that RE has within SF. We have found that the extraction of candidate fills results in a substantial amount of recall loss early in the pipeline, and have argued that this recall loss is a major limiter on performance. We explored defining a representation of the task which reflects the behaviour and assumptions we want to model, in the form of a graph-based label propagation approach. We added features to drastically increase the connectedness and discriminability of the graph. Finally, we considered the definition of explicitness: how explicit mentions need to be to qualify for extraction. This is a substantial problem that has not been addressed, and causes problems for annotation consistency, data and evaluation. A standard of explicitness is ultimately required by a downstream application, and is critical for consistent evaluation.

Slot filling is fundamental to enabling many knowledge-based applications. In this work, we have explored critical concerns for applying slot filling to these applications. We have analysed an number of problems in the task, ranging from substantial recall loss, to particular issues of explicitness in the task definition itself.

In total, this work has provided critical analysis for system designers seeking to apply slot filling techniques to challenging real-world problems.

# Bibliography

Heike Adel, Benjamin Roth, and Hinrich Schütze. 2016. Comparing convolutional neural networks to traditional models for slot filling. In *Proceeding of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 828–838.

Alicia Ageno, Pere Ramon Comas, Ali Naderi, Horacio Rodríguez, and Jorge Turmo. 2013. The TALP participation at TAC-KBP 2013. In *Proceedings of the 2013 Text Analysis Conference*.

Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting Relations from Large Plain-text Collections. In *Proceedings of 5th ACM Conference on Digital Libraries*, pages 85–94.

Eneko Agirre, Angel X. Chang, Daniel S. Jurafsky, Christopher D. Manning, Valentin I. Spitkovsky, and Eric Yeh. 2009. Stanford-UBC at TAC-KBP. In *Proceedings of the 2009 Text Analysis Conference*.

Enrique Alfonseca, Katja Filippova, Jean-Yves Delort, and Guillermo Garrido. 2012. Pattern Learning for Relation Extraction with a Hierarchical Topic Model. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, pages 54–59.

Gabor Angeli, Arun Chaganty, Angel Chang, Kevin Reschke, Julie Tibshirani, Jean Wu, Osbert Bastani, Keith Siilats, and Christopher D. Manning. 2013. Stanford's 2013 KBP System. In *Proceedings of the 2013 Text Analysis Conference*.

Gabor Angeli and Christopher D. Manning. 2014. NaturalLI: Natural Logic Inference for Common Sense Reasoning. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 534–545.

Gabor Angeli, Julie Tibshirani, Jean Y. Wu, and Christopher D. Manning. 2014. Combining Distant and Partial Supervision for Relation Extraction. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1556–1567.

Chinatsu Aone, Lauren Halverson, Tom Hampton, and Mila Ramos-Santacruz. 1998. SRA: Description of the IE2 system used for MUC-7. In *Proceedings of the 7th Message Understanding Conference*, pages 1–14.

Shumeet Baluja, Rohan Seth, D. Sivakumar, Yushi Jing, Jay Yagnik, Shankar Kumar, Deepak Ravichandran, and Mohamed Aly. 2008. Video Suggestion and Discovery for Youtube: Taking Random Walks Through the View Graph. In *Proceedings of the 17th International Conference on World Wide Web*, pages 895–904.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open Information Extraction from the Web. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, pages 2670–2676.

Michele Banko and Oren Etzioni. 2008. The Tradeoffs Between Open and Traditional Relation Extraction. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pages 28–36.

Dan Bikel, Vittorio Castelli, Radu Florian, and Ding-jung Han. 2009. Entity Linking and Slot Filling through Statistical Processing and Inference Rules. In *Proceedings of the 2009 Text Analysis Conference*.

Avrim Blum and Tom Mitchell. 1998. Combining Labeled and Unlabeled Data with Co-training. In *Proceedings of the 11th Conference on Computational Learning Theory*, pages 92–100.

Sergey Brin. 1998. Extracting Patterns and Relations from the World Wide Web. In *Selected papers from the International Workshop on The World Wide Web and Databases*, pages 172–183.

Peter F. Brown, Peter V. Desouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479. MIT Press, Cambridge, MA, USA.

Razvan C. Bunescu and Raymond J. Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 724–731.

Razvan C. Bunescu and Raymond J. Mooney. 2007. Learning to Extract Relations from the Web using Minimal Supervision. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 576–583.

Lorna Byrne and John Dunnion. 2010. UCD IIRG at TAC 2010 KBP Slot Filling Task. In *Proceedings of the 2010 Text Analysis Conference*.

Lorna Byrne and John Dunnion. 2011. UCD IIRG at TAC 2011. In *Proceedings of the 2011 Text Analysis Conference*.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Toward an Architecture for Never-Ending Language Learning. In *Proceedings of the 24th Conference on Artificial Intelligence*, pages 1306–1313.

Vittorio Castelli, Radu Florian, and Ding-jung Han. 2010. Slot Filling through Statistical Processing and Inference Rules. In *Proceedings of the 2010 Text Analysis Conference*.

Daniel Chada, Christian Aranha, and Carolina Monte. 2010. An Analysis of The Cortex Method at TAC 2010 KBP Slot-Filling. In *Proceedings of the 2010 Text Analysis Conference*.

Eugene Charniak. 2000. A Maximum-entropy-inspired Parser. In *Proceedings of the 1st Conference of the North American Chapter of the Association for Computational Linguistics*, pages 132–139.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine N-best Parsing and MaxEnt Discriminative Reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 173–180.

Jinxiu Chen, Donghong Ji, Chew Lim Tan, and Zhengyu Niu. 2006. Relation extraction using label propagation based semi-supervised learning. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 129–136.

Zheng Chen, Suzanne Tamang, Adam Lee, Xiang Li, Wen-Pin Lin, Matthew Snover, Javier Artiles, Marissa Passantino, and Heng Ji. 2010. CUNY-BLENDER TAC-KBP2010 Entity Linking and Slot Filling System Description. In *Proceedings of the 2010 Text Analysis Conference*.

Nancy Chinchor and Elaine Marsh. 1998. MUC-7 Information Extraction Task Definition. In *Proceedings of the 7th Message Understanding Conference*, pages 359–367.

Laura Chiticariu, Yunyao Li, and Frederick Reiss. 2013. Rule-Based Information Extraction is Dead! Long Live Rule-Based Information Extraction Systems! In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 827–832.

Michael Collins and Nigel Duffy. 2002. Convolution Kernels for Natural Language. In *Advances in Neural Information Processing Systems 14*, pages 625–632. MIT Press, Cambridge, MA, USA.

Michael Collins and Yoram Singer. 1999. Unsupervised Models for Named Entity Classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 100–110.

Mark Craven and Johan Kumlien. 1999. Constructing Biological Knowledge Bases by Extracting Information from Text Sources. In *Proceedings of the 7th International Conference on Intelligent Systems for Molecular Biology*, pages 77–86.

D. Alan Cruse. 1986. *Lexical Semantics*. Cambridge University Press, Cambridge, UK.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency Tree Kernels for Relation Extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*.

James R. Curran, Tara Murphy, and Bernhard Scholz. 2007. Minimising semantic drift with mutual exclusion bootstrapping. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, page 172–180.

Tim Dawborn and James R. Curran. 2014. docrep: A lightweight and efficient document representation framework. In *Proceedings of the 25th International Conference on Computational Linguistics: Technical Papers*, pages 762–771.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford Typed Dependencies Representation. In *Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8.

Cıcero N. dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. In *Proceedings of the 53rd Annual*

*Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 626–634.

Joe Ellis, Jeremy Getman, Justin Mott, Xuansong Li, Kira Griffit, Stephanie M. Strassel, Robert Parker, and Jonathan Wright. 2013. Linguistic Resources for 2013 Knowledge Base Population Evaluations. In *Proceedings of the 2013 Text Analysis Conference*.

Joe Ellis, Xuansong Li, Kira Griffit, Stephanie M. Strassel, and Jonathan Wright. 2012a. Linguistic Resources for 2012 Knowledge Base Population Evaluations. In *Proceedings of the 2012 Text Analysis Conference*.

Joe Ellis, Heather Simpson, Kira Griffit, Hoa Trang Dang, Ralph Grishman, Heng Ji, Catherine DePrince, Jeremy Getman, and Thomas Riese. 2012b. TAC KBP Slots version 2.4. Linguistic Data Consortium, Philadelphia, PA, USA. http://www.nist.gov/tac/2012/KBP/task_guidelines/TAC_KBP_Slots_V2.4.pdf.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545.

Jenny Rose Finkel, Trond Grenager, and Christopher D. Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370.

Ryan Gabbard, Marjorie Freedman, and Ralph Weischedel. 2011. Coreference for Learning to Extract Relations: Yes, Virginia, Coreference Matters. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, pages 288–293.

Sanyuan Gao, Yichao Cai, Si Li, Zongyu Zhang, Jingyi Guan, Yan Li, Hao Zhang, Weiran Xu, and Jun Guo. 2010. PRIS at TAC2010 KBP Track. In *Proceedings of the 2010 Text Analysis Conference*.

Joseph E. Gonzalez, Yucheng Low, Haijie Gu, Danny Bickson, and Carlos Guestrin. 2012. PowerGraph: Distributed Graph-parallel Computation on Natural Graphs. In *Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation*, pages 17–30.

Ralph Grishman. 2013. Off to a Cold Start: NY University's 2013 Knowledge Base Population Systems. In *Proceedings of the 2013 Text Analysis Conference*.

Ralph Grishman and Bonan Min. 2010. NY University KBP 2010 Slot-Filling System. In *Proceedings of the 2010 Text Analysis Conference*.

Zhou GuoDong, Su Jian, Zhang Jie, and Zhang Min. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 427–434.

Marti A. Hearst. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In *Proceedings of the 14th International Conference on Computational Linguistics - Volume 2*, pages 539–545.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based Weak Supervision for Information Extraction of Overlapping Relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 541–550.

Raphael Hoffmann, Congle Zhang, and Daniel S. Weld. 2010. Learning 5000 Relational Extractors. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 286–295.

Heng Ji and Ralph Grishman. 2011a. Knowledge base population: successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 1148–1158.

Heng Ji and Ralph Grishman. 2011b. Knowledge Base Population: Successful Approaches and Challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 1148–1158.

Heng Ji, Ralph Grishman, and Hoa Trang Dang. 2011. Overview of the TAC 2011 Knowledge Base Population Track. In *Proceedings of the 2011 Text Analysis Conference*.

Xu Jian, Hector Liu, Qin Lu, Patty Liu, and Chen Chen Wang. 2011. PolyUCOMP in TAC 2011 Entity Linking and Slot-Filling. In *Proceedings of the 2011 Text Analysis Conference*.

Jing Jiang and Chengxiang Zhai. 2007. A systematic exploration of the feature space for relation extraction. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 113–120.

Nanda Kambhatla. 2004. Combining Lexical, Syntactic, and Semantic Features with Maximum Entropy Models for Extracting Relations. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics on Interactive Poster and Demonstration Sessions*, pages 178–181.

Jin-Dong Kim, Jung-jae Kim, Xu Han, and Dietrich Rebholz-Schuhmann. 2015. Extending the evaluation of Genia Event task toward knowledge base construction and comparison to Gene Regulation Ontology task. *BMC Bioinformatics*, 16(10):1–13. BioMed Central, London, United Kingdom.

Bryan Kisiel, Justin Betteridge, Matt Gardner, Jayant Krishnamurthy, Ndapa Nakashole, Mehdi Samadi, Partha Talukdar, Derry Wijaya, and Tom Mitchell. 2013. CMUML System for KBP 2013 Slot Filling. In *Proceedings of the 2013 Text Analysis Conference*.

Dan Klein and Christopher D. Manning. 2003. Accurate Unlexicalized Parsing. In *In Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430.

Jonathan K. Kummerfeld, David Hall, James R. Curran, and Dan Klein. 2012. Parser Showdown at the Wall Street Corral: An Empirical Investigation of Error Types in Parser Output. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1048–1059.

Ni Lao, Tom Mitchell, and William W. Cohen. 2011. Random Walk Inference and Learning in a Large Scale Knowledge Base. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 529–539.

LDC. 1995. North American News Text Corpus. LDC95T21. Linguistic Data Consortium, Philadelphia, PA, USA.

LDC. 2002. Annotation Guidelines for Relation Detection and Characterization (RDC) version 3.6. Linguistic Data Consortium, Philadelphia, PA, USA. https://www.ldc.upenn.edu/sites/www.ldc.upenn.edu/files/rdc-guidelines-v3.6.pdf.

LDC. 2004. Annotation Guidelines for Relation Detection and Characterization (RDC) version 4.3.2. Linguistic Data Consortium, Philadelphia, PA, USA. https://catalog.ldc.upenn.edu/docs/LDC2005T09/guidelines/EnglishRDCV4-3-2.PDF.

LDC. 2005. ACE (Automatic Content Extraction) English Annotation Guidelines for Relations version 5.8.3. Linguistic Data Consortium, Phil-

adelphia, PA, USA. https://www.ldc.upenn.edu/sites/www.ldc.upenn.edu/files/english-relations-guidelines-v5.8.3.pdf.

LDC. 2009. TAC KBP Reference Knowledge Base. LDC2009E58. Linguistic Data Consortium, Philadelphia, PA, USA.

LDC. 2017. Abstract Meaning Representation (AMR) Annotation Release 2.0. LDC2017T10. Linguistic Data Consortium, Philadelphia, PA, USA.

Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2011. Stanford's multi-pass sieve coreference resolution system at the CoNLL-2011 shared task. In *Proceedings of the 15th Conference on Computational Natural Language Learning: Shared Task*, pages 28–34.

John Lehmann, Sean Monahan, Luke Nezda, Arnold Jung, and Ying Shi. 2010. LCC Approaches to Knowledge Base Population at TAC 2010. In *Proceedings of the 2010 Text Analysis Conference*.

Si Li, Sanyuan Gao, Zongyu Zhang, Xinsheng Li, Jingyi Guan, Weiran Xu, and Jun Guo. 2009. PRIS at TAC 2009: Experiments in KBP Track. In *Proceedings of the 2009 Text Analysis Conference*.

Xuansong Li, Joe Ellis, Kira Griffit, Stephanie M. Strassel, Robert Parker, and Jonathan Wright. 2011. Linguistic Resources for 2011 Knowledge Base Population Evaluation. In *Proceedings of the 2011 Text Analysis Conference*.

Xuansong Li, Jeremy Getman, and Stephanie Strassel. 2014. Overview of Linguistic Resources for the TAC KBP 2014 Evaluations: Planning, Execution, and Results. In *Proceedings of the 2014 Text Analysis Conference*.

Xuansong Li, Jeremy Getman, Stephanie Strassel, and Heather Simpson. 2013. TAC KBP 2013 Assessment. Linguistic Data Consortium, Philadelphia, PA, USA. http://surdeanu.info/kbp2013/TAC_KBP_2013_Assessment_Guidelines_V1.3.pdf.

Yan Li, Sijia Chen, Zhihua Zhou, Jie Yin, Hao Luo, Liyin Hong, Weiran Xu, Guang Chen, and Jun Guo. 2012. PRIS at TAC2012 KBP Track. In *Proceedings of the 2012 Text Analysis Conference*.

Dekang Lin and Patrick Pantel. 2001. DIRT - Discovery of Inference Rules from Text. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 323–328.

Jianhua Lin. 1991. Divergence Measures Based on the Shannon Entropy. *IEEE Transactions on Information Theory*, 37(1):145–151. IEEE Press Piscataway, NJ, USA.

Thomas Lin, Mausam, and Oren Etzioni. 2010. Commonsense from the Web: Relation Properties. In *2010 AAAI Fall Symposium Series*, pages 70–75.

Thomas Lin, Mausam, and Oren Etzioni. 2012a. Entity Linking at Web Scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pages 84–88.

Thomas Lin, Mausam, and Oren Etzioni. 2012b. No Noun Phrase Left Behind: Detecting and Typing Unlinkable Entities. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 893–903.

Christopher Malon, Bing Bai, and Kazi Saidul Hasan. 2012. Slot-Filling by Substring Extraction at TAC KBP 2012 (Team Papelo). In *Proceedings of the 2012 Text Analysis Conference*.

Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, and Oren Etzioni. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534.

Tara McIntosh and James R. Curran. 2008. Weighted mutual exclusion bootstrapping for domain independent lexicon and template acquisition. In *In Proceedings of the Australasian Language Technology Association Workshop*, pages 97–105.

Tara McIntosh and James R. Curran. 2009. Reducing Semantic Drift with Bagging and Distributional Similarity. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, pages 396–404.

Paul McNamee and Hoa Dang. 2009. Overview of the TAC 2009 Knowledge Base Population Track. In *Proceedings of the 2009 Text Analysis Conference*.

Paul McNamee, Mark Dredze, Adam Gerber, Nikesh Garera, Tim Finin, James Mayfield, Christine Piatko, Delip Rao, David Yarowsky, and Markus Dreyer. 2009. HLTCOE Approaches to Knowledge Base Population at TAC 2009. In *Proceedings of the 2009 Text Analysis Conference*.

Scott Miller, Michael Crystal, Heidi Fox, Lance Ramshaw, Richard Schwartz, Rebecca Stone, Ralph Weischedel, and the Annotation Group. 1998. BBN: Description of the SIFT System as Used for MUC-7. In *Proceedings of the 7th Message Understanding Conference*.

Bonan Min and Ralph Grishman. 2012. Challenges in the Knowledge Base Population Slot Filling Task. In *Proceedings of the Eight International Conference on Language Resources and Evaluation*, pages 1148–1158.

Bonan Min, Xiang Li, Ralph Grishman, and Ang Sun. 2012a. NY University 2012 System for KBP Slot Filling. In *Proceedings of the 2012 Text Analysis Conference*.

Bonan Min, Shuming Shi, Ralph Grishman, and Chin-Yew Lin. 2012b. Ensemble semantics for large-scale unsupervised relation extraction. In *Proceedings of*

*the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1027–1037.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant Supervision for Relation Extraction Without Labeled Data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, pages 1003–1011.

Thahir Mohamed, Estevam Hruschka, and Tom Mitchell. 2011. Discovering Relations between Noun Categories. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1447–1455.

Raymond J. Mooney and Razvan C. Bunescu. 2006. Subsequence Kernels for Relation Extraction. In *Advances in Neural Information Processing Systems 18*, pages 171–178. MIT Press, Cambridge, MA, USA.

Alexander A. Morgan, Lynette Hirschman, Marc Colosimo, Alexander S. Yeh, and Jeff B. Colombe. 2004. Gene Name Identification and Normalization Using a Model Organism Database. *Journal of Biomedical Informatics*, 37(6):396–410. Elsevier Science, San Diego, CA, USA.

Truc-Vien T. Nguyen and Alessandro Moschitti. 2011a. End-to-end Relation Extraction Using Distant Supervision from External Semantic Repositories. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, pages 277–282.

Truc-Vien T. Nguyen and Alessandro Moschitti. 2011b. Joint distant and direct supervision for relation extraction. In *Proceedings of the Fifth International Joint Conference on Natural Language Processing*, pages 732–740.

Truc-Vien T. Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. 2009. Convolution Kernels on Constituent, Dependency and Sequential Structures for Relation

Extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1378–1387.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank Citation Ranking: Bringing Order to the Web. Technical report, 1999-66, Stanford InfoLab, Stanford, CA, USA.

Glen Pink and James R. Curran. 2014. SYDNEY at TAC 2014. In *Proceedings of the 2014 Text Analysis Conference*.

Glen Pink, Joel Nothman, and James R. Curran. 2014. Analysing recall loss in named entity slot filling. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 820–830.

Glen Pink, Will Radford, Will Cannings, Andrew Naoum, Joel Nothman, Daniel Tse, and James R. Curran. 2013. SYDNEY_CMCRC at TAC 2013. In *Proceedings of the 2013 Text Analysis Conference*.

Barbara Plank and Alessandro Moschitti. 2013. Embedding Semantic Similarity in Tree Kernels for Domain Adaptation of Relation Extraction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1498–1507.

Arpana Rawal, Ani Thomas, M K Kowar, and Sanjay Sharma. 2013. ARPAN-IBIT_DURG: KBP English Slot-filling Task Challenge. In *Proceedings of the 2013 Text Analysis Conference*.

Sebastian Riedel, Limin Yao, Benjamin M. Marlin, and Andrew McCallum. 2013. Relation Extraction with Matrix Factorization and Universal Schemas. In *Proceedings of 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 74–84.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling Relations and Their Mentions Without Labeled Text. In *Proceedings of the 2010 European Conference on Machine Learning and Knowledge Discovery in Databases: Part III*, pages 148–163.

Benjamin Roth, Tassilo Barth, Grzegorz Chrupała, Martin Gropp, and Dietrich Klakow. 2014. RelationFactory: A Fast, Modular and Effective System for Knowledge Base Population. In *Proceedings of 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 89–92.

Benjamin Roth, Tassilo Barth, Michael Wiegand, and Dietrich Klakow. 2013a. A Survey of Noise Reduction Methods for Distant Supervision. In *Proceedings of the 2013 Workshop on Automated Knowledge Base Construction*, pages 73–78.

Benjamin Roth, Tassilo Barth, Michael Wiegand, Mittul Singh, and Dietrich Klakow. 2013b. Effective Slot Filling Based on Shallow Distant Supervision Methods. In *Proceedings of the 2013 Text Analysis Conference*.

Benjamin Roth, Grzegorz Chrupała, Michael Wiegand, Mittul Singh, and Dietrich Klakow. 2012. Generalizing from Freebase and Patterns using Cluster-Based Distant Supervision for KBP Slot-Filling. In *Proceedings of the 2012 Text Analysis Conference*.

Robert P. Schumaker and Hsinchun Chen. 2009. Textual analysis of stock market prediction using breaking financial news: The AZFin text system. *ACM Transactions on Information Systems*, 27(2). Association for Computing Machinery, NY, NY, USA.

Yusuke Shinyama and Satoshi Sekine. 2006. Preemptive Information Extraction Using Unrestricted Relation Discovery. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 304–311.

Sameer Singh, Limin Yao, David Belanger, Ari Kobren, Sam Anzaroot, Michael Wick, Alexandre Passos, Harshal Pandya, Jinho Choi, Brian Martin, and Andrew McCallum. 2013. Universal Schema for Slot Filling and Cold Start: UMass IESL at TACKBP 2013. In *Proceedings of the 2013 Text Analysis Conference*.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2004. Learning Syntactic Patterns for Automatic Hypernym Discovery. In *Advances in Neural Information Processing Systems 17*, pages 1297–1304.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A Core of Semantic Knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, pages 697–706.

Ang Sun, Ralph Grishman, and Satoshi Sekine. 2011a. Semi-supervised Relation Extraction with Large-scale Word Clustering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 521–529.

Ang Sun, Ralph Grishman, Wei Xu, and Bonan Min. 2011b. NY University 2011 System for KBP Slot Filling. In *Proceedings of the 2011 Text Analysis Conference*.

Mihai Surdeanu. 2013. Overview of the TAC2013 Knowledge Base Population Evaluation: English Slot Filling and Temporal Slot Filling. In *Proceedings of the 2013 Text Analysis Conference*.

Mihai Surdeanu, Sonal Gupta, John Bauer, David McClosky, Angel X. Chang, Valentin I. Spitkovsky, and Christopher D. Manning. 2011. Stanford's Distantly-Supervised Slot-Filling System. In *Proceedings of the 2011 Text Analysis Conference*.

Mihai Surdeanu and Heng Ji. 2014. Overview of the English Slot Filling Track at the TAC2014 Knowledge Base Population Evaluation. In *Proceedings of the 2014 Text Analysis Conference*.

Mihai Surdeanu, David McClosky, Julie Tibshirani, John Bauer, Angel X. Chang, Valentin I. Spitkovsky, and Christopher D. Manning. 2010. A Simple Distant Supervision Approach for the TAC-KBP Slot Filling Task. In *Proceedings of the 2010 Text Analysis Conference*.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance Multi-label Learning for Relation Extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465.

Kumutha Swampillai and Mark Stevenson. 2011. Extracting Relations Within and Across Sentences. In *Proceedings of Recent Advances in Natural Language Processing*, pages 25–32.

Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. 2012. Reducing Wrong Labels in Distant Supervision for Relation Extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, pages 721–729.

Partha Pratim Talukdar and Koby Crammer. 2009. New Regularized Algorithms for Transductive Learning. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 442–457.

Partha Pratim Talukdar and Fernando Pereira. 2010. Experiments in Graph-based Semi-supervised Learning Methods for Class-instance Acquisition. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1473–1481.

Partha Pratim Talukdar, Joseph Reisinger, Marius Paşca, Deepak Ravichandran, Rahul Bhagat, and Fernando Pereira. 2008. Weakly-supervised Acquisition of Labeled Class Instances Using Graph Random Walks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 582–590.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich Part-of-speech Tagging with a Cyclic Dependency Network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 173–180.

Vasudeva Varma, Praveen Bysani, Kranthi Reddy, Vijay Bharat, Santosh GSK, Karuna Kumar, Sudheer Kovelamudi, Kiran Kumar N, and Nitin Maganti. 2009. IIIT Hyderabad at TAC 2009. In *Proceedings of the 2009 Text Analysis Conference*.

Vasudeva Varma, Praveen Bysani, Kranthi Reddy, Vijay Bharat Reddy, Sudheer Kovelamudi, Srikanth Reddy Vaddepally, Radheshyam Nanduri, Kiran Kumar N, Santosh GSK, and Nitin Maganti. 2010. IIIT Hyderabad in Guided Summarization and Knowledge Base Population. In *Proceedings of the 2010 Text Analysis Conference*.

Ngoc Thang Vu, Heike Adel, Pankaj Gupta, and Hinrich Schütze. 2016. Combining recurrent and convolutional neural networks for relation classification. In *Proceeding of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 534–539.

Richard C. Wang and William W. Cohen. 2008. Iterative Set Expansion of Named Entities Using the Web. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pages 1091–1096.

Yafang Wang, Maximilian Dylla, Marc Spaniol, and Gerhard Weikum. 2012. Coupling Label Propagation and Constraints for Temporal Fact Extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, pages 233–237.

Yafang Wang, Bin Yang, Lizhen Qu, Marc Spaniol, and Gerhard Weikum. 2011. Harvesting Facts from Textual Web Sources by Constrained Label Propagation. In

*Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, pages 837–846.

Fei Wu and Daniel S. Weld. 2007. Autonomously Semantifying Wikipedia. In *Proceedings of the 16th ACM Conference on Conference on Information and Knowledge Management*, pages 41–50.

Fang Xu, Stefan Kazalski, Grzegorz Chrupala, Benjamin Roth, Xujian Zhao, Michael Wiegand, and Dietrich Klakow. 2011. Saarland University Spoken Language Systems Group at TAC KBP 2011. In *Proceedings of the 2011 Text Analysis Conference*.

Sheng Xu, Chunxia Zhang, Zhendong Niu, Rongyue Mei, Junpeng Chen, Junjiang Zhang, and Hongping Fu. 2013. BIT's Slot-Filling Method for TAC-KBP 2013. In *Proceedings of the 2013 Text Analysis Conference*.

Roman Yangarber and Ralph Grishman. 1998. NYU: Description of the Proteus/PET System as Used for MUC-7 ST. In *Proceedings of the 7th Message Understanding Conference*.

Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. 2011. Structured Relation Discovery Using Generative Models. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1456–1466.

David Yarowsky. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, pages 189–196.

Alexander Yates and Oren Etzioni. 2009. Unsupervised Methods for Determining Object and Relation Synonyms on the Web. *Journal of Artificial Intelligence Research*, 34:255–296. AAAI Press, Palo Alto, CA, USA.

Dian Yu and Heng Ji. 2016. Unsupervised person slot filling based on graph mining. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 44–53.

Dian Yu, Haibo Li, Taylor Cassidy, Qi Li, Hongzhao Huang, Zheng Chen, and Heng Ji. 2013. RPI-BLENDER TAC-KBP2013 Knowledge Base Population System. In *Proceedings of the 2013 Text Analysis Conference*.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel Methods for Relation Extraction. *Journal of Machine Learning Research*, 3:1083–1106. JMLR.org.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of the 25th International Conference on Computational Linguistics*, pages 2335–2344.

Min Zhang, Jie Zhang, Jian Su, and Guodong Zhou. 2006. A Composite Kernel to Extract Relations Between Entities with Both Flat and Structured Features. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 825–832.

Xingxing Zhang, Jianwen Zhang, Junyu Zeng, Jun Yan, and Zheng Chen. 2013. Towards Accurate Distant Supervision for Relational Facts Extraction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 810–815.

Shubin Zhao and Ralph Grishman. 2005. Extracting Relations with Integrated Information Using Kernel Methods. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 419–426.

Guodong Zhou, Min Zhang, Donghong Ji, and Qiaoming Zhu. 2007. Tree kernel-based relation extraction with context-sensitive structured parse tree inform-

ation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 728–736.

Xiaojin Zhu and Zoubin Ghahramani. 2002. Learning from labeled and unlabeled data with label propagation. Technical report, CMU-CALD-02, Carnegie Mellon University, Pittsburgh, PA, USA.