UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

MITJA TRAMPUŠ

Semantični pristopi h konstrukciji domenskih predlog in odkrivanju mnenj iz naravnega besedila

DOKTORSKA DISERTACIJA

Mentorica:
prof. dr. Dunja Mladenić

Somentor:
prof. dr. Janez Demšar

Ljubljana, 2015

UNIVERSITY OF LJUBLJANA

FACULTY OF COMPUTER AND INFORMATION SCIENCE

MITJA TRAMPUŠ

Semantic approaches to domain template construction and opinion mining from natural language

DOCTORAL THESIS

Advisor:
prof. dr. Dunja Mladenić

Co-advisor:
prof. dr. Janez Demšar

Ljubljana, 2015

Povzetek

Semantični pristopi h konstrukciji domenskih predlog in odkrivanju mnenj iz naravnega besedila

Večina algoritmov za rudarjenje besedil je danes zasnovana na leksikalnih predstavitvah vhodnih podatkov, npr. z vrečo besed (angl. *bag of words*). Ena od možnih alternativ je, da tekst najprej pretvorimo v semantično predstavitev, ki je *strukturirana* in uporablja le *vnaprej definirane oznake*, npr. koncepte iz leksikona. Ta disertacija preučuje uporabnost pristopov, osnovanih na tovrstni predstavitvi, in sicer na primeru dveh problemov s področja analize množic dokumentov: odkrivanje skupne strukture v vhodnih dokumentih (*konstrukcija domenskih predlog*, angl. *domain template construction*) ter podpora odkrivanju mnenjskih razlik (*rudarjenje mnenj*, angl. *opinion mining*) v vhodnih dokumentih.

V disertaciji se najprej posvetimo možnostim za pretvorbo naravnega besedila v semantično predstavitev. Predstavimo in primerjamo dve novi metodi, ki se med seboj razlikujeta po kompleksnosti in izrazni moči. Prva metoda, izkaže se za bolj obetavno, temelji na skladenjski razčlembi teksta (angl. *dependency parse tree)*, poenostavljeni v preproste semantične okvirje (*semantic frames*) z atributi, poravnanimi na WordNet. Druga metoda strukturira besedilo v semantične okvirje z uporabo tehnike označevanja semantičnih vlog (*semantic role labeling*) in poravna podatke na ontologijo Cyc.

Z uporabo prve od teh dveh metod vpeljemo in evalviramo dve metodi za konstrukcijo domenskih predlog iz dokumentov iz posamezne domene (npr. poročila o bombnih napadih). Predlogo definiramo kot množico ključnih atributov (npr. napadalec, število žrtev, ...). Ključna ideja obeh metod je, da generirata takšne posplošene semantične okvirje, da so njihove bolj specifične instance (kot jih defninira WordNet hierarhija podpomenk) pogoste v vhodnem tekstu. Vsak od takšnih okvirjev nam predstavlja atribut domenske predloge. Dosežemo rezultate, ki so po točnosti vsaj na nivoju sodobnih obstoječih metod, pri tem pa atribute predlog tudi natančno tipovno omejimo, česar konkurenčne metode ne omogočajo.

V zadnjem večjem sklopu vpeljemo in predstavimo programski sistem za izpostavljanje mnenjskih razlik v novicah. Za poljuben dogodek uporabniku prikažemo nabor znanih člankov o dogodku ter omogočimo navigacijo na podlagi treh semantičnih atributov: čustvo, tematika in geografsko poreklo. Rezultata navigacije sta množica relevantnih dokumentov, ki jih dinamično uredimo glede na uporabnikov fokus, ter fokusiran povzetek teh člankov, zgrajen v realnem času. Povzetek je zgrajen z novo metodo, temelječo na zgoraj omenjeni predstavitvi teksta s semantičnimi okvirji. Uporabniška študija celotnega sistema pokaže pozitivne rezultate.

Ključne besede: odkrivanje znanj iz podatkov, odkrivanje znanj iz besedila, ontologije, procesiranje naravnega jezika

Abstract

Semantic approaches to domain template construction and opinion mining from natural language

Most of the text mining algorithms in use today are based on lexical representation of input texts, for example bag of words. A possible alternative is to first convert text into a semantic representation, one that captures the text content in a *structured* way and using only a set of *pre-agreed labels*. This thesis explores the feasibility of such an approach to two tasks on collections of documents: identifying common structure in input documents ("*domain template construction*"), and helping users find differing opinions in input documents ("*opinion mining*").

We first discuss ways of converting natural text to a semantic representation. We propose and compare two new methods with varying degrees of target representation complexity. The first method, showing more promise, is based on dependency parser output which it converts to lightweight semantic frames, with role fillers aligned to WordNet. The second method structures text using Semantic Role Labeling techniques and aligns the output to the Cyc ontology.

Based on the first of the above representations, we next propose and evaluate two methods for constructing frame-based *templates* for documents from a given domain (e.g. bombing attack news reports). A template is the set of all salient attributes (e.g. attacker, number of casualties, ...). The idea of both methods is to construct abstract frames for which more specific instances (according to the WordNet hierarchy) can be found in the input documents. Fragments of these abstract frames represent the sought-for attributes. We achieve state of the art performance and additionally provide detailed type constraints for the attributes, something not possible with competing methods.

Finally, we propose a software system for exposing differing opinions in the news. For any given event, we present the user with all known articles on the topic and let them navigate them by three semantic properties simultaneously: sentiment, topical focus and geography of origin. The result is a dynamically reranked set of relevant articles and a near real time focused summary of those articles. The summary, too, is computed from the semantic text representation discussed above. We conducted a user study of the whole system with very positive results.

Keywords: data mining, text mining, ontologies, natural language processing

Izjava o avtorstvu

Spodaj podpisani Mitja Trampuš z vpisno številko 63040301 sem avtor doktorske disertacije z naslovom *Semantic approaches to domain template construction and opinion mining from natural language*. S svojim podpisom zagotavljam, da:

- sem doktorsko disertacijo izdelal samostojno pod vodstvom mentorice prof. dr. Dunje Mladenić in somentorstvom prof. dr. Janeza Demšarja;
- so elektronska oblika doktorske disertacije, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko doktorske disertacije;
- in soglašam z javno objavo elektronske oblike doktorske disertacije v zbirki *Dela FRI*.

V Ljubljani, maja 2015. Podpis avtorja:

Acknowledgements

First and foremost, thank you to Dunja Mladenić, my mentor, and Marko Grobelnik, an informal but no less important advisor, for letting me explore machine learning and guiding me along the way, and for showing me the more earthly aspects of academia like the importance of presenting yourself well. The strictly academic support is however eclipsed by the personal support, understanding, selflessness and trust that they showed from the very beginning to the very end. I will be lucky to ever again get to work in such a familial atmosphere and with such supervisors.

Thanks to Janez Demšar, the co-mentor, for showing me how easy it is to think yourself into a bubble when not seeking feedback outside of your regular environment, and for bursting the bubble on occasion. Janez was also my all-important tie to the faculty, with which I grew more distant working at Jozef Stefan Institute than I would have liked.

On a very pragmatic note, thanks to our faculty's administrative staff and Zdenka Velikonja in particular. They were enormously helpful and patient in helping me navigate the ofttimes muddy waters of grad school's formal processes.

Through the years in which this thesis was directly or indirectly formed, a number of collaborators helped in various ways, most commonly by providing reusable software modules. Specifically, Tadej Štajner developed the sentiment detection module (Section 5.2.5) and is the primary author of Enrycher (Section 2.4.5). Luka Stopar developed the framework that supports the web version of the application, Janez Brank developed the clustering module (Section 2.4.5) and Blaž Novak co-developed the NewsFeed (Section 2.4) with me. Tomaž Hočevar implemented the baseline for the evaluation of webpage cleartext extraction. Daniele Pighin conducted the bulk of DiversiNews evaluation with the help of anonymous crowdsourced workers. Delia Rusu, Marko Grobelnik and Enrique Alfonseca participated in the early stages of the DiversiNews application, helped define the goals and supported the collaboration of everyone involved. Primož Škraba provided useful advice on deriving domain templates (Chapter 4) and other topics; his breadth of technical knowledge is inspiring.

The above people contributed more to my development than the occasional software module, of course. I have shared many pleasant and informative conversations with them, along with Lorand Dali, Blaž Fortuna, Janez Starc, Andrej Muhič, Aljaž Košmerlj, Lan Žagar, Lovro Šubelj, Ruben Sipoš and many other colleagues and friends both inside and outside the department, with conversation topics ranging from the newest in machine learning to bashing the cafeteria.

Last but very important, no small thanks go to everybody close to me – mom, dad, Matija, Maja, babi, dedi, babi Lina, and friends both data-mining and non-data-mining[1] – for making the years leading to this thesis happy and enjoyable outside work as well, and for tolerating me in the moments when I let any PhD-induced frustrations leak outside their rightful domicile. Be it known that Šapa the dog handled it particularly gracefully.

Work on this thesis was supported in part by the Slovenian Research Agency and the European Commission under PASCAL2 (IST-NoE-216886), ACTIVE (IST-2008-215040), RENDER (FP7-257790) and XLIKE (FP7-ICT-288342-STREP). Thanks to the funding agencies that made the work possible, and to the project collaborators who provided use helpful suggestions and comments or contributed otherwise.

Thanks also to Mr Obama and Ms Merkel, the quintessential protagonists of sample sentences in NLP, for staying in power and keeping those sentences relevant throughout my grad studies.

[1] A special friend included.

Contents

List of acronyms

AI Artificial Intelligence

AUC Area Under the Responder Operator Curve (ROC)

CCA Canonical Correlation Analysis

CRF Conditional Random Field

CSS Cascading Style Sheets

CSV Comma-Separated Values

CT Charateristic Triplet

DAG Directed Acyclic Graph

DB DataBase

DOM Document Object Model

FVM Frequent Verb Modifier

HMM Hidden Markov Model

HTML HyperText Markup Language

HTTP HyperText Transfer Protocol

IC Information Content

IDF Inverse Document Frequency

IE Information Extraction

KB Knowledge Base

MDS MultiDimensional Scaling

ML Machine Learning

MUC Message Understanding Contest

NER Named Entity Recognition / Named Entity Resolution

NL Natural Language

NLP Natural Language Processing

NN NouN

NP Noun Phrase

POS Part Of Speech

PP PrePosition

RSS Really Simple Syndication

SDP Simplified Dependency Parses

SRL Semantic Role Labeling

SVM Support Vector Machine

SVO Subject-Verb-Object

TAC Text Analysis Contest

TF Text Frequency

TLD Top-Level Domain

UI User Interface

VP Verb Phrase

WN WordNet

WSD Word Sense Disambiguation

XML eXtensible Markup Language

Chapter 1

Introduction

Written word is one of the most important human means of communication and dissemination of knowledge; so important, in fact, that we equate the beginning of civilization with the invention of writing. The ease of knowledge dissemination increased dramatically with Gutenberg's invention of the printing press, and recently again with ubiquitous internet access. It is becoming easier and easier to both consume and produce text, and unlike spoken word, this data is much less ephemeral and is often preserved for years or even hundreds of years. As a result, the total amount of textual data available to us is climbing rapidly, which brings about the need for us to be able to process, analyze, summarize, link, organize and make sense of text automatically or semi-automatically. Without such methods, a large part of our collective knowledge goes unobserved and unexploited due to the limited processing bandwidth of the human brain. Thus, the research discipline of *text mining* evolved, dealing with automated ways of processing text.

Another type of data that visibly gained prominence with the advent of computers is *semantic data*. This is data in a structured form, presented using a pre-agreed set of labels that are related to the real world. Reuse of labels across applications is strongly encouraged. This makes the data more easily interpretable and interoperable and comparable. In particular, it supports integration of data coming from various sources. As we are accumulating increasing amounts of data, this ability is becoming more and more important. The most common use case is to merge application-specific data with *background knowledge*, a database that encodes knowledge of broader interest and is often (though not necessarily) more static in nature. This background knowledge provides context in which we can more easily interpret and "understand" the core data. For example, knowing the recipe for a dish tells us quite a lot about the dish, but having access to an extensive database of common cooking ingredients (i.e. background knowledge) lets us infer a lot more about the dish — its nutritional value, potential risks to people with allergies and risks due to raw ingredients, the likely taste, country of origin, expected number of servings and so on. It is often easy for humans to take background knowledge for granted, because we consider a lot of it "common sense". Everybody knows that butter is fatty, China is a big country, $1 million is a high annual salary, and similar

facts. Computers do not, and that can hurt their reasoning powers.

A natural idea then is to try and bring the benefits of semantic approaches to methods for analysis of text data. Note that text is a typical example of *un*structured data without clearly defined or easily understood semantics. Machine learning and data mining methods that deal with text often represent the data as a bag of character or word n-grams and give up on "understanding" what those sequences of characters mean. In many applications, this yields good results, but it leaves us wondering: what if we were to semanticize at least some fragments of the text, i.e. find links between those fragments and background knowledge in the form of lexicons, encyclopedia entries, geographical databases and more? With *semantic approaches to text mining*, we represent text data with semantic attributes, i.e. with labels with known meanings, and attempt to solve text mining tasks using that representation. As we do so, we aim to exploit background knowledge to gain an advantage compared to bag of words or similar models.

Despite a surge in research activity on the intersection of semantics and text mining in the last five to ten years, there are still many unexplored scenarios to consider. In this thesis, we consider applying a *shallow, structured* semantic representation to two problems on a *collection* of documents. Almost any analysis of relationship(s) between a set of documents can be cast as a search for either *commonalities* or *differences* between those documents; we attempt to explore each of these two main groups of analyses with a representative task and a proposed novel solution to the task.

1.1 Thesis Overview

The work presented in this thesis traverses the whole pipeline of tasks from obtaining collections of documents, transforming them into a semantic representation and performing analyses on them. As discussed before, we focus on two analysis tasks over a set of documents, one that aims to discover the *commonalities* in the set of documents, and one that aims to highlight the *differences*.

Chapter 2. This chapter overviews existing work related to this thesis and the language resources available for developing semantic methods of dealing with text. We also discuss the acquisition of online news data used throughout the thesis; Section 2.4 describes how to do this robustly, reliably and at scale.

Chapter 3. The text collected from the internet is inherently mostly unstructured data. While we do use some of the metadata available directly in a structured form, what lies at the heart of the methods presented here is the idea of presenting the text itself semantically. The semantic representation we choose is that of *semantic frames*. The transformation of text into this form is presented in Chapter 3. The chapter also discusses possible variants of this representation, their advantages and weaknesses.

Chapter 4. Equipped with this representation, we discuss the task of constructing *domain templates*: given a set of documents from a single domain (e.g. reports

on bomb attacks), the goal is to automatically identify the set of attributes that characterize such documents (e.g. location, number of victims, perpetrator, ...). We present two methods for doing so, both based on representing text as a set of *semantic triplets* $\boxed{\text{concept}} \xrightarrow{\text{relation}} \boxed{\text{concept}}$ trivially derived from the semantic frames. Both methods are novel and have performance comparable to the state of the art while in addition providing type information for the identified attributes.

Chapter 5. In the search for differences within a set of documents, we present not an autonomous method but rather a system that helps human users identify and expose those differences more easily. In particular, our system lets users analyze clusters of news articles reporting on a single news event. We represent each article with structured, interpretable attributes like sentiment and geolocation of the publisher, and give the user controls to navigate articles based on these attributes. Because reading articles is still a time-consuming task, we also present the most relevant content to the user in the form of a summary. True to the theme of the thesis, the latter is constructed based on the semantic triplet representation of articles. The end result is a system that allows users to efficiently discover diversity and biases in media in a way not possible before.

Chapter 6 assembles the lessons learned in previous chapters into concluding remarks on the use of semantic text representations, and explicitly lists the original contributions to science.

1.2 Contributions Overview

The key contributions to science are listed in Section 6.1. In brief, however, they are:

- A new method for semantically representing text from "any" domain, with broader scope than supervised relation extraction algorithms but still sufficient accuracy. (Section 3.2)
- Two new methods for obtaining domain templates, evaluated against state of the art. (Sections 4.2, 4.3)
- An interface for exposing opinions in news, based on navigating along novel dimensions, validated in a user study. (Section 5.1)

Let us summarize the novelties in a more descriptive way as well.

In Chapter 3 we propose and evaluate several techniques for text semantization. While there is no shortage of related work (see Section 2.2), it mostly focuses on extracting a *small* number of semantic objects or relations with *high* precision and/or recall. There is a much smaller set of projects that valiantly attempt to extract a *high* number)"all") of objects and/or relations. As this is a much harder task, they focus on *precision* and sacrifice (sentence-level) recall, with the goal of aggregating the extracted information over a large dataset and reconstructing "common sense"

facts or other relations that are relatively pervasive throughout a set of analyzed documents. Our work also deals with general-purpose semantic representations (i.e. a large number of objects/relations), but sacrifices precision for *recall*, exploring if it is viable to semantically represent a single document well enough that it enables common text mining tasks, e.g. document similarity measurement. Prior work in this direction is very scarce, and little was known about the empirical limitations of current tooling and static resources. We demonstrate that it is possible to extract (shallowly) semantic representations with a balance of reasonable recall (most sentences generate at least one feature) and precision.

We "test-drove" the new representation on the little-researched task of domain template construction (Chapter 4) – only a few papers exist on the topic, and none of them employ structured data representations or background knowledge. As the task's output is inherently structured, we deemed it promising to devise an algorithm for it that uses the abovementioned semantic representation. The results confirmed our hypothesis: our method allows one to infer templates for a collection of documents, keeping the quality of the produced templates on par with prior state of the art, but unlike any prior work, also providing additional structure and type information for the templates.

Finally, we combined those same representations with additional semantic data and used them as the foundation of a news exploration system (Chapter 5). The innovation is on the system level rather than in individual data analysis components. To our knowledge, no existing system provides a comparable level of in-depth analysis for individual news events. It is now easier than before to understand the details of a controversial news story, its different aspects, and the viewpoints of various stakeholders.

Chapter 2

Background

2.1 Terminology and Notation

Before diving deeper, let us expand on some of the key terms and expressions used throughout the thesis. Some of them appear directly in the title, *Semantic approaches to domain template construction and opinion mining from natural language*, others just cannot be avoided when speaking of commonalities and differences in collections of news. Some deserve to be mentioned because they are specific to a narrower domain and not widely used (e.g. *role filler*), others are quite commonplace and used in a number of contexts (e.g. *story*), so we explain more precisely what we mean by them.

- **Semantic data** is a loosely defined term. While the dictionary definition – "*semantic* — Of or relating to meaning, especially meaning in language." – is clear, there is no unanimous definition of properties that a data representation should have to be deemed semantic. We use the adjective semantic to refer to data that is meaningful and interpretable without further human intervention as a merit of a rich context in which it has been placed. The context is typically ontological (e.g. the string "President Obama" can be given context by associating it with Obama's DBpedia page with its many relations and attributes) or structural (e.g. the string "Luke" in a list is meaningful if we also encode the fact that this is the list of 10 most frequent baby names in the US in 2013).

- Many of our experiments deal with news. We use the term **article** to refer to the text from a single news webpage and **story** to refer to the informally defined collection of articles that are reporting on the same **event**. Because there is a one-to-one correspondence between events (which happen in real life) and stories (which report on them), we sometimes use the two terms interchangeably.

- When abstracting away the set of common attributes for a collection of articles on related events (e.g. earthquakes), we present them in terms of recurring

roles (e.g. magnitude, location). The collection of all roles is called a **domain template**. Values that fill the roles (e.g. "3.4" for the magnitude) are **role fillers**. Note that the terminology in related work is highly varied; Table 2.1 contains the details.

- **Opinion** or **viewpoint** is a person's take on a topic. When the person authors a document (e.g. news article) on the topic, the opinion is reflected in aspect emphases, judgment statements, disposition towards subject matter and similar. A "common sense" definition suffices as we do not model opinions explicitly in our work; we instead model properties that are likely to correlate with opinions: sentiment, geographical provenance and topical focus.

Several methods in this thesis are based on a graph-like representation of documents, roles, and summaries, with labeled nodes denoting concepts and labeled edges denoting relations between them. We use the following notation:

- $\boxed{\text{Node}}$ for concepts extracted directly from documents, e.g. $\boxed{\text{Obama}}$.
- $\boxed{\textit{NodeType}}$ for generic, automatically inferred concepts, e.g. $\boxed{\textit{politician}}$.
- $\boxed{\text{Node}_1} \xrightarrow{\text{relation}} \boxed{\text{Node}_2}$ for relations.

Throughout the thesis, we use "quoted sans-serif text" to present (snippets of) actual input text, and **bolded text** to emphasize important points or concepts.

Additional terms and notations specific to individual sections are correspondingly introduced later on.

2.2 Related Work

The structure of this section closely follows the structure of the thesis as a whole – in the subsections, we group related work by the chapter to which it is the most pertinent.

Statement of authorship. A considerable portion of the work presented in this thesis has been published before, in the following papers:

[1] Trampuš M, Novak B. *Internals of an aggregated web news feed*, in Proc. of SiKDD 2012

[2] Trampuš M, Mladenić D. *High-Coverage Extraction of Semantic Assertions from Text*, in Proc. of SiKDD 2011

[3] Trampuš M, Mladenić D. *Constructing Event Templates from Written News*, in Proc. of WI/IAT 2009

[4] Trampuš M, Mladenić D, *Approximate Subgraph Matching for Detection of Topic Variations*, in Proc. of DiversiWeb 2011

[5] Trampuš M, Mladenić D. *Constructing Domain Templates from Text: Exploiting Concept Hierarchy in Background Knowledge*, in Information Technology and Control. Accepted, awaiting publication.

[6] Trampuš M, Fuart F, Berčič J, Rusu D, Stopar L, Štajner T. *(i)DiversiNews – a Stream-Based, On-line Service for Diversified News*, in Proc. of SiKDD 2013

[7] Trampuš M, Fuart F, Pighin D, Štajner T, Berčič J, Rusu D, Stopar L, Grobelnik M. *DiversiNews: Surfacing Diversity in Online News* in AI Magazine. Accepted, awaiting publication.

[8] Rusu D, Trampuš M, Thalhammer A. *Diversity-Aware Summarization*, a deliverable of the RENDER project

Full citations are available in the Bibliography section. Parts of the text in this thesis are taken verbatim from those publications. I declare that I am the first and principal author of all of those publications[1] and have consent from all the co-authors to re-publish here.

2.2.1 Semantic Representations of Text

Almost any formalization for semantically representing text can be recast as a collection of *relations*. The task of semanticizing text therefore reduces to that of *relation extraction*, a subfield of information extraction (IE). The field of semantic fact extraction is much less researched. In "standard" IE, the topic domain is constructed beforehand and remains fixed. There is a large body of IE research available; see e.g. [9] for a survey or the very active TAC (Text Analysis Conference) challenge [10]. Of even more interest are Open Information Extraction systems; "open" in the task name refers to the fact that these systems construct new concepts and relations on the fly. Of similar interest are systems that do not quite perform open IE but consider a very large number of predefined relations.

The first open IE system was TextRunner [11, 12]. TextRunner consider each noun phrase in a sentence as a possible entity and models binary relations with noncontiguous sequences of words appearing between two entities. For a candidate pair of entities, a sequence tagger (named O-CRF, based on conditional random

[1] With the exception of *Diversity-Aware Summarization*, where I am the sole author of its only section partially included in this thesis.

fields) decides for each word whether it is a part of the relation phrase or not. The system starts with a large number of heuristically labeled training examples, and has the possibility of bootstrapping itself by interchangeably learning relation phrases and entity pairs. TextRunner focuses on relations that can be expressed as verb phrases. It attempts to link entities to Freebase; the relations are always kept at the level of string sequences.

ReVerb [13] is the successor to TextRunner. Unlike TextRunner, it identifies potential relation phrases first, using a handcrafted regular expression over POS tags. All relations include a verb. If a relation phrase is surrounded by two noun phrases, the triple constitutes a candidate relation. Results are further refined by only keeping relation phrases that occur between multiple different noun phrases. Finally, the authors train a supervised model that assigns a confidence score to every relation. The model was trained on a small hand-labeled dataset but is independent of the relation phrase; the features are lexical and POS-tag based.

SOFIE [14] and its successor PROSPERA [15] are interesting in that they perform relation extraction simultaneously with alignment to the target ontology. The ontology is then also central to placing type constraints on relation candidates. For example, for `presidentOf(X,Y)` to hold, `X` has to be of type *Person*. Both systems use YAGO [16][2] as the ontology, restricting themselves to extracting Wikipedia entities and infobox relations.

O-CRF, ReVerb, PROSPERA and the majority of other related work is based on lexical and POS patterns. In contrast, Ollie [17] uses syntactical features derived from dependency parse trees. Ollie uses ReVerb to generate a seed set of relations; using those relations, it finds new sentences that contain the same words but different phrasing, and finally it learns link patterns in the dependency tree that connect the relation constituents. The patterns are in fact lexico-syntactical as the system allows constraints on the content of tree nodes that appear in the pattern. By using patterns of this kind, Ollie is able to find relations that are not expressed by nouns.

Another Open IE system using dependecy parse trees is "KNext-" [18]; the transformation of parse trees into the structured representation of choice is simply a matter of manual rules, not unlike in our SDP approach (Section 3.2). Its output tends towards the more heavily formal logic; for example, the fragment "those in the US" would be recognized as extraction-worthy and converted to $\exists x, y, z.$ $\textsc{thing-referred-to}(x) \wedge \textsc{country}(y) \wedge \textsc{exemplar-of}(z, y) \wedge \textsc{in}(x, z)$.

Also prominent is NELL, the Never Ending Language Learner [19, 20]. Not unlike SOFIE/PROSPERA, it relies on existing knowledge to provide constraints and hints during acquisition of new statements; however, the ontology in this case is being built by the system from scratch. NELL is unique in that it automatically proposes new categories, relations and even ontological rules. Here, we describe only candidate relation extraction from text. Each relation is seeded with a small number of samples, from which two cooperating subsystems mutually bootstrap themselves, also with the help of other subsystems (e.g. rule inference, learning entity types).

[2] A lightweight ontology built by cleaning wikipedia/DBpedia.

Coupled Pattern Learner (CPL) searches for frequently co-occurring lexical patterns between pairs of noun phrases, not unlike TextRunner. Also based on co-occurrence statistics, CSEAL learns HTML patterns that capture relations expressed as lists or tables on webpages.

A further very abridged but reference-rich overview can be found in a recent tutorial by Suchanek and Weikum [21].

The most established and successful projects of the above are KnowItAll (encompasses TextRunner, ReVerb, Ollie and more) and NELL. They both aim to keep learning through time, bootstrapping their precision and recall from previously acquired knowledge. Both have been running for several years, with the long-term goal of capturing and structuring as much of common-sense knowledge from the internet as possible. In fact, for most of the open IE systems above aim to extract universal truths, "web-scale information extraction" being a common keyphrase. Precision is crucial, particularly if bootstrapping is intended. Our requirements are a bit different in that we need semantic representations of a single piece of text in order to perform further computations on it; we therefore care primarily about the recall at the level of statements within an individual document, not about precision at the level of universally true statements as web-scale extraction systems do.

A very different but highly relevant take on semantic representations is provided by deep learning methods that have recently enjoyed a lot of popularity. These methods convert inputs (images, sound, ..., text) to low-dimensional vectors that carry a lot of semantics, but little to no formal structure. Mikolov et al.'s word2vec approach [22] acts on individual words and is one of the seminal papers in the area dealing with text. Even more closely related to our work are approaches that model whole sentences or paragraphs, based on various recursive or hierarchical neural net designs. One of the more prominent topologies here is the Dynamic Convolutional Neural Net [23]. Alternatively, the approach by Grefenstette et al. [24] maps text directly to a structured representation, though it requires training data in the form of sentence-parse pairs. The algoritm proceeds in two steps. In the first, a latent "interlingua" vector is computed using a simple word2vec-like network mapping sentences to their parses. In the second step, only the projection of sentences to the latent space is retained, and is in turn used as an input to training a generative recursive neural network that produces parses.

Semantic Role Labeling (SRL). There is a relatively large amount of existing work on automated SRL. The basic design of all prominent methods is unchanged since the first attempt by Gildea and Jurafsky [25] – a supervised learning approach on top of PropBank or FrameNet annotated data (see Section 2.3.2), with hand-constructed features from parse trees.

A basic preprocessing step is constituency parsing (although a few rare examples opt for chunking or other shallower methods [26]). This gives rise to most of the features; feature engineering was shown to be very important [27]. The problem

is then typically divided into frame selection, role detection, and role identification steps; all of them are almost always performed using classic ML techniques. Here, too, deep learning has recently brought improvements to state of the art; for example, Hermann and Das [28] improve the frame selection phase by augmenting the features set with word2vec-based description of the trigger word context.

The best insight into SRL is offered by various challenges [29, 30, 31]. More recently, methods have been proposed that perform sequence labeling directly [32, 33] and avoid the need for explicit deep parsing by using structured learning. Additional tricks can be employed outside the core learning method, for example using text rewriting to increase the training set size [34].

2.2.2 Topic Template Construction

The task of domain template construction has seen relatively little research activity. The majority of existing articles take a similar approach. They start by representing the documents as dependency parse trees, thus abstracting away some of the language variability and making pattern discovery more feasible. The patterns found in these trees are often further clustered to arrive at more general, semantic patterns or pattern groups. In the remainder of this section, we describe the most closely related contributions in more detail.

Several articles focus on a narrow domain and/or assume a large amount of domain-specific background knowledge. For example, Das et al. [35] analyze weather reports to extract patterns of the form "`[weather front type]` is moving towards `[compass direction]`." where they manually create rules (based on shallow semantic parsing roles and part-of-speech tags) for identifying instances of concepts such as `[compass direction]` and `[weather front type]`. Once these concepts are identified, they cluster verbs based on WordNet and then construct template patterns for each verb cluster independently; a pattern is every frequent subsequence of semantic roles within sentences involving verbs from the verb cluster. The idea is only partially transferable to the open domain; authors themselves point out that they rely on the formulaic language that is typical of weather reports.

The method by Shinyama and Sekine [36] makes no assumptions about the domain but does limit itself to discovering named-entity slots. It tags named entities and clusters them based on their surrounding context in constituency parse trees. The problem of data sparsity (a logical statement can be expressed with many natural language syntactic trees) is alleviated by simultaneously analyzing multiple news articles about a single news story – an approach also taken by our FGS method in Section 4.2. In the end, each domain slot is described by the set of its common syntactic contexts.

Filatova et al. [37] use a tf-idf-like measure to identify the top 50 verbs for the domain and extract all dependency parse trees in which those verbs appear. The trees are then generalized: every named entity is replaced with its type (person, location, organization, number). Frequent subtree mining is used on these trees to

identify all subtrees occurring more than a predetermined number of times. From the frequent trees, all the nodes except the verb and the slot node (i.e. the generalized named entity) are removed; the remainder represents a template slot. The approach is similar to several other papers; unlike those, it is also well evaluated, which is why we choose to compare against it. The method is unnamed; because it focuses on modifiers of frequent verbs, we refer to it as the Frequent Verb Modifier (FVM) method.

Chambers and Jurafsky [38, 39, 40] take a different approach: they first cluster verbs based on how closely together they co-occur in documents. For each cluster, they treat cluster verbs' modifiers (object, subject) as slots and further cluster them by representing each verb-modifier pair (e.g. `(explode,subj)`) as a vector of other verb-modifier pairs that tend to refer to the same noun phrase (e.g. `[(plant,obj), (injure,subj)]`). Both rounds of clustering observe a number of additional constraints omitted here. The method is also capable of detecting topics from a mixture of documents, positioning the work close to open information extraction. This article, too, is systematically evaluated; however, their three golden standard templates come from MUC-4[3] and have only 2, 3 and 4 slots, respectively, making the measurement noisy and less suitable for comparison among algorithms.

Finally, Qiu et al. [41] propose a method with more involved preprocessing. Unlike the other methods, which consume parse trees, this method operates on semantic frames coming from a Semantic Role Labeling (SRL) system. Within each document, the frames are connected into a graph based on their argument similarity and proximity in text. The frames across document graphs are clustered with an EM algorithm to identify clusters of frames that semantically likely represent the same template slot(s). This approach is interesting in that it is markedly different from the others; sadly, there is no quantitative evaluation of the quality of the produced templates and even the qualitative evaluation (= sample outputs) is scarce.

In contrast to our work, none of the above methods explore the benefits and shortcomings of using semantic background knowledge. However, a hierarchy/lattice of concepts, the very form of background knowledge employed by us, was recently successfully used in related tasks of constructing ontologies from relational databases in a data-centric fashion [42] and semiautomatic ontology building [43].

Note that almost all of the related work, like ours, concerns itself with newswire or similar well-written documents, allowing parsers to play a crucial role. For less structured texts, parsing results are of questionable quality if obtainable at all, and domain-specific approaches are needed. This was observed for example by Michelson and Knoblock [44] who automatically construct a domain template from craigslist ad titles, deriving for example a taxonomy of cars and their attributes. Their templates also significantly differ from all the approaches listed above in that they are not verb- or action-centric.

Our proposed method is unique in that it tightly integrates background knowl-

[3] A reference dataset provided in the scope of the 4th Message Understanding Conference (MUC) in 1992

edge into the template construction process; all existing approaches rely instead on contextual similarities to cluster words or phrases into latent slots. However, an approach similar to ours has been successfully used in a related and similarly novel task of event prediction [45]. Starting with events from news titles (e.g. "Tsunami hit Malaysia", "Tornado struck in Indonesia"), the authors employed background knowledge to derive generic events and compute likely causality relations between them, e.g. a "[natural disaster] hit [Asian country]" event predicts a "[number] people die in [Asian country]" event.

Topic template construction as feature selection. We can also view our task as a case of feature selection for the binary classification problem of deciding whether a given document belongs to the target domain. The templates we are looking for aim to abstract/summarize all that is characteristic of a particular domain. If we view individual components of the templates – slots and their context words – as features appearing in documents, the template for a domain is intuitively composed of the most discriminative features for classification into that domain.

There are, however, two specifics that need to be accounted for and which prevent us from directly applying feature selection techniques:

1. The template consists of a combination of features rather than individual features. In particular, context words and even whole small semantic subgraphs only contribute to the template in a sensible way if they help qualify a slot. Blindly applying feature selection results in many statements that, although topical, do not vary across documents, e.g. $\boxed{\text{attack}} \xrightarrow{\text{claim}} \boxed{\text{life}}$ for the bombing attack domain. While the presence or absence of this fact is interesting, it cannot be part of the template as defined in this thesis because neither "attack" not "life" represent slots that could be filled/specialized by individual documents.

2. More importantly, the features need to be considered in the context of their containing taxonomy, here WordNet. In particular, template slots do not appear in documents as-is; their specializations do.

The first issue is relatively easy to tackle with pre- or post-filtering for features that do not vary across documents. The second issue is essentially the problem of feature selection in the face of (here non-linearly) correlated features, which is usually attacked with the wrapper techniques of forward selection and backward elimination (i.e. iteratively adding and removing features) or other related methods.

We discuss a somewhat feature selection inspired approach in Section 4.3.

The terminology of template construction. The domain template construction task has so far been tackled by people coming from different backgrounds, using different names for the task itself and the concepts related to it. We collected the assorted terminology in Table 2.1. Our terminology mostly follows that of Filatova.

Qiu's is influenced by the early terminology introduced in the 1990s for Information Extraction tasks (where the domain templates were created by hand), e.g. at the Message Understanding Conference (MUC) [46]. Chambers's "roles" and "role fillers" are normally used with Semantic Role Labeling (SRL) [47]; interestingly, he does not use the SRL term "frame" for templates. Shinyama's naming choices are strongly rooted in relational databases.

2.2.3 Exposing Opinion Diversity

Our work in the area of opinion mining is applied to the domain of newswire, where opinions abound and the value of understanding their diversity is clear. There is existing research demonstrating that no single news provider can cover all the aspects of a story, as well as research into how to improve the situation with the help of tools similar to ours.

Opinion distribution in media. There is a large body of research associated with identifying, measuring and explaining media bias. Frequently, the research in this area focuses on diversity and biases along a single dimension, typically the political orientation (liberal vs. conservative). An et al. [48], for example, tracked Facebook users' patterns of sharing links to articles and confirmed that liberals were much more likely to share liberally-inclined articles and vice versa for conservatives.

Maier [49] surveyed several thousand news sources cited in newspapers and found factual or subjective disagreement between the sources and the citing articles in 61% of the articles. This shows that in order to get objective information, one should ideally have easy access to multiple articles on a story.

Voakes and Kapfer [50] analyzed the multiple news stories and found that the content diversity is on average substantially lower than the source diversity; in other words, simply reading a high *number* of sources does not necessarily provide diverse content. This suggests that diversity-aware news browsing systems should "understand" news on some level, be aware of its content and other attributes.

While DiversiNews, the tool we propose in Chapter 5, is effective at discovering diverse viewpoints in news, the incentive for such exploration still has to come from the user. A recent user study [51] evaluated what happens if the diversity is *forced* upon (or away from) the user. Test subjects were asked about their political preferences and then exposed to a collection of news that agreed with their preferences to varying extents. Two groups of users were discernible: one was happiest if all the articles agreed with their views, while the other was happiest when served a balanced mixture of news that both support and challenge their views. Although these users represented a minority, there clearly is a target audience for technologies that make diverse content more accessible.

Opinion-aware news browsing. While the work listed above is mostly descriptive in nature, there is also no lack of prescriptive research trying to provide solutions

This work	**Filatova** [37]	**Das** [35]	**Chambers** [40]	**Qiu** [41]	**Shinyama** [36]	**Example**
domain, topic	domain, topic	domain	domain	scenario	—	bombing attack
slot, property	slot	slot	role, slot	salient aspect, slot	relation	attacker
slot filler	slot filler	slot value	role filler	sample modifier	—	John Smith
pattern, triplet	slot structure	template	syntactic relation	—	basic pattern	$person \xrightarrow{\text{detonate}} \text{bomb}$
schema, domain/topic template	dom. template	—	narrative schema	scenario template	unrestricted relations	(all slots)

Table 2.1: Consolidation of terminology in related work. Following our terminology, the *domain* is what the input documents have in common. *Properties/slots* are the concepts we would like to discover. *Slot filler* is a specific value that can fill the slot; this is what algorithms have to abstract away to produce the slots. *Patterns* are the syntactic context of slots using which the algorithm identifies slots and usually also presents them to the user; their content and representation are highly algorithm-specific. The *domain template* is the collection of all patterns for a domain and is the final output of the algorithm.

that would ameliorate the current state of affairs. In his PhD thesis, Munson [52] suggests several visualizations of a user's browsing patterns, for example a graph of the prevalence of liberal-leaning articles among those read by the user. As the graph evolves through time, the user can track her reading habits, holding herself accountable to a balanced diet of opinions. This complements our work where the goal is not to *identify* a users need for balanced reporting, but rather to help her *satisfy* that need.

Very closely related to our work is NewsCube by Park et al. [53, 54], a system for news aggregation, processing and diversity-aware delivery. DiversiNews and NewsCube have a lot in common – they both choose to expose diversity through a standalone news portal, and a lot of the preprocessing work is therefore similar across the two systems. There are however notable differences in delivery. For one, NewsCube offers no interactive exploration but rather groups and ranks articles within a story in a fixed way that is hoped to offer maximally diverse information in one screenful. Secondly, NewsCube focuses on topical (or *aspect*, as they call it) diversity only.

Later work by the same authors extends the information presented by NewsCube with a more detailed characterization of biases and a novel data acquisition method. NewsCube 2.0 [55] is a browser add-on that allows users to *collaboratively tag* articles with the types of exhibited biases (e.g. omission of information, suggestive photo, subjective phrasing etc.) and place them on the "framing spectrum", i.e. decide how strongly liberal or conservative the article's outlook is. User input is then presented in the NewsCube interface.

Another noteworthy and much more mature news portal is the Europe Media Monitor [56] which aims to bring together viewpoints across languages. The website offers a number of news aggregation and analysis tools that track stories across time, languages and geographic locations. It also detects breaking news stories and hottest news topics. Topic-specific processing is used, for example, to monitor EU policy areas[4] and possible disease outbreaks [57].

In a similar vein, DisputeFinder [58] is a browser extension that lets users mark up disputable claims on web pages and point to claims to the contrary. The benefit comes from the collaborative nature of the tool: when browsing, the extension highlights known disputed claims and presents to the user a list of articles that support a different point of view.

In contrast to most of the work that focuses on political diversity, Zhang et al. [59] identified similar and diverse news sources in terms of the prevalent emotions they convey.

Mining diversity in other news modalities. News in the "traditional" form of articles is among the most amenable to analysis. For news in other forms (video, tweets), the promotion of diversity is mostly restricted to attempts at making the data collections more easily navigable.

[4] `http://emm.newsbrief.eu/`

Social Mention[5] is a social media search and analysis platform which aggregates different user generated content, providing it as a single information stream. The platform provides sentiment (positive, negative, and neutral), top keywords, top users or hashtags related to the aggregated content.

The Global Twitter Heartbeat [60] project performs real-time Twitter stream processing, taking into account 10% of the Twitter feed. The text of each tweet is analyzed in order to assign its location. A heat map infographic displays the tweet location, intensity and tone.

2.3 Language Resources

When representing information in a semantic form, high-quality language resources are of tantamount importance. Although unsupervised approaches to extracting semantics exist, most often we rely on previous work to provide help with mapping natural text to existing knowledge bases. The help comes in the form of labels within the knowledge bases themselves (KB concepts are associated with natural language words or phrases) or annotated corpora to serve as training data (i.e. collections of text that are already mapped to the KB, most often manually).

An equally important resource for dealing with natural text are the various linguistic tools that introduce some formal structure in text. Part of Speech (POS) taggers, chunkers, dependency and constituency parsers, named entity recognizers etc. fall into this category.

A comprehensive list of all important resources for dealing with natural text is well beyond the scope of this thesis. Instead, we briefly introduce the ones used in this thesis.

2.3.1 Cyc

Cyc [61] is a large ontology of "common sense knowledge", an encyclopedia (and more) in the form of first- and higher-order predicate logic. Cyc has been built mostly by hand by a team of ontologists since the 1980s. As a consequence, it has an exceptionally well worked-out upper layer (i.e. abstract concepts and rules); the completeness of lower levels (e.g. specific people or events) however is often lacking.

Concepts in Cyc are represented as `#$ConceptName` and relations as `#$relation` (note the capitalization!). A lisp-like syntax is used; for example, this is a Cyc statement asserting that Barack Obama is a US president:
`(#$isa #$BarackObama #$UnitedStatesPresident)`

Cyc's expansiveness and expressiveness is one of its biggest strengths but also weaknesses. Mapping knowledge onto Cyc is hard even manually [62], and fully automatic mapping is still far from solved in general, especially because there is a

[5] `http://www.socialmention.com`

dearth of Cyc-annotated training data. Links between Cyc concepts and English natural language are established in particular in the following three ways[6]:

- Concepts' *glosses*. The gloss of a concept is its highly technical, disambiguation-oriented description. For example, the gloss for `#$UnitedStatesPresident` is "A specialization of both `#$UnitedStatesPerson` and `#$President_HeadOfGovernmentOrHeadOfState`. Each instance of `#$UnitedStatesPresident` is a person who holds the office of President of the `#$UnitedStatesOfAmerica`."
- The `#$denotation` relation describes English "aliases" of a concept. For example, it holds that (`#$denotation #$UnitedStatesPresident` "Presidents of the US").
- Cyc's same-as connections to other ontologies with potentially richer lexical annotations, most notably WordNet. However, these connections tend to be automatically derived, so they introduce errors and have only partial coverage.

Importantly, Cyc comes with a powerful inference engine that can reason about facts that are only implicitly stated in the knowledge base.

2.3.2 FrameNet

FrameNet [63, 64] is a knowledge base built around the theory of *frame semantics*. In short, FrameNet is a formal set of action types and attributes for describing actions[7]. Each single action (e.g. drinking tea) is represented with its type (*Drinking*) and attributes (*liquid*="tea"). The set of action types and their associated attributes is fixed and carefully thought out – that is the main value of FrameNet, along with the annotated examples it provides.

An event type along with its attributes is called a *frame*. The attributes are called *roles*, and their values in a specific instantiation of a frame (i.e. in a specific sentence) are called *role fillers*. The structured representations of text presented in this thesis follow the frame semantics approach (albeit simplified), and we adopt the terminology as well.

There are 1020 frames, of which 540 have at least 40 annotated examples and 180 have at least 200. Each frame is also tagged with a list of trigger words (e.g. drink.v, drink.n, sip.v etc. for the Drinking frame). Every frame and every role is defined with a short natural-language definition. Frames are loosely connected with several relations, most notably generalization/specialization. For each pair of connected frames, the mapping between their roles is given as well.

2.3.3 WordNet

WordNet [65, 66] is a general-purpose inventory of concepts. Each concept in WordNet, called a *synset*, is represented by a short description and a collection of English

[6]This is a greatly simplified view on Cyc's natural language mechanisms.

[7]Primarily actions; also relations and objects, but their coverage is poorer and they are of less interest to our work.

words that can denote that concept. In contrast to Cyc (Section 2.3.1), WordNet is much shallower and centered around the English language; it strives to achieve good coverage of English words first, and philosophical and abstract concepts second.

Synsets are connected with a very limited set of relations. Of those, the one that has by far the highest coverage and is the most widely used is the hypernym/hyponym relation. For practical purposes, WordNet can therefore be treated simply as a taxonomy of concepts.

WordNet is primarily a *middle- to lower-level* knowledge base (or lightweight ontology), meaning it describes particularities rather than high-level philosophical concepts: for example, there is a concept for a "chair" in WordNet, but not one for "a non-transient movable physical object".

WordNet as a standard. WordNet has seen wide-spread use in many areas of text modeling. Notable alternative freely available general-purpose ontologies with a populated lower layer include: Wikipedia and the structured, cleaned-up incarnation of its infoboxes, DBpedia [67]; YAGO [16], which merges WordNet with Wikipedia; and Freebase [68], which also originated from Wikipedia but has since been extensively collaboratively edited. Note that all of these originate from either WordNet or Wikipedia; these two resources provide the de-facto standard enumerations of entities today.

A similar conclusion has been reached by Boyd-Graber et al. [69] who note that "WordNet has become the lexical database of choice for NLP".

2.3.4 GATE

GATE [70] is a relatively widely used natural language processing and text annotation framework. The architecture is plugin-based, and plugins exist for many NLP tasks, often simply conveniently wrapping existing state of the art tools. The core distribution includes tools for tokenization, POS (part of speech) tagging, lemmatization, parsing, and named entity recognition, among others.

ANNIE, the module for named entity recognition, was developed by the same research group as GATE and is one of the more prominent components of the framework. ANNIE is tuned to perform on newswire and achieves 80–90% precision and recall (depending on the dataset) on that domain [71].

2.3.5 Stanford Parser

The Stanford Parser [72] is one of the more popular and best performing freely available deep parsers. Its language model is an unlexicalized[8] probabilistic context-free grammar.

[8]Meaning that the model doesn't try to "remember" e.g. that when "fast" appears next to "track", "fast" tends to be an adjective, not an adverb, and it modifies "track".

The basic version of the Stanford parser produces *constituency parse trees* which marks words with POS-like tags (noun, verb, adjective etc.) to produce tree leaves, then recursively groups them according to which word modifies which other word (or word group).

The constituency parse tree can be used to derive a *dependency parse tree*, which is more semantic in nature. The leaves of a dependency parse tree are still words, but now connected with relations like *direct object* and *determiner*. In the case of the Stanford parser, this transformation is achieved with a set of non-deterministic hand-crafted rules [73].

The performance of parsers is measured by micro-averaging the performance on *(typed) attachment* – for each tree node, how well does the algorithm predict what its parent node should be, and what is its relation to the parent? For the Stanford parser suite, the constituency parser achieves attachment F_1 of 86.3% [72] and the dependency parser that of 84.2% [74].

2.3.6 GeoNames

GeoNames[9] is a freely available geographical database of about 3 million geographical entities with over 10 million names – many places have alternate names. For each place, it contains the its type, geographical coordinates, elevation, population etc.

Though not a language resource in the strictest sense of the world, we use GeoNames in our work to perform *geocoding* – mapping human-readable, English place names (countries, cities, addresses) to the corresponding geographical coordinates. This is a rudimentary form of text "understanding".

2.4 News Data

The methods described in this thesis fall in the broader scope of text mining. To develop, test and evaluate them, we needed a suitably large collection of text data. We settled on using web news as the data source, as they are written in a clean language (unlike blogs or microblogs), virtually unlimited in size (unlike static datasets), diverse in writing style and topic coverage, and freely available. As an added benefit, current news concern us all, they are a relatable and relevant test polygon.

As a result, we developed Newsfeed [1], a substantial piece of infrastructure for acquisition and pre-processing of news from the internet which we present in this section.

Note on authorship and scope. NewsFeed was developed in collaboration with Blaž Novak. His work is essential to its functioning – it deals with efficient and robust downloading of the content. In this section, we greatly simplify or even omit

[9] `http://www.geonames.org`

the description of many of his contributions, focusing instead on the processing parts that more directly influence the work in the later chapters. Note that this section is therefore not a complete or reference description of the system. NewsFeed includes additional components not mentioned here that were successfully used and continue to be used in a range of research projects by people in our department and beyond.

2.4.1 Overview

NewsFeed is a news aggregator that provides a real-time aggregated stream of textual news items, with metadata normalized to a common format and the text content cleared of markup. The pipeline performs the following main steps:

1. Periodically crawls a list of RSS feeds and a subset of Google News and obtains links to news articles
2. Downloads the articles, taking care not to overload any of the hosting servers
3. Parses each article to obtain
   (a) Potential new RSS sources, to be used in step (1)
   (b) Cleartext version of the article body
4. Enriches the articles with a series of external services
5. Expose the stream of cleartexted, annotated news articles to end users.

2.4.2 Data Characteristics

2.4.2.1 Sources

As of early 2014, the crawler actively monitors about 250 000 feeds from 55 000 hostnames. The list of sources is constantly being changed – stale sources get removed automatically, new sources get added from crawled articles. In addition, we occasionally manually prune the list of sources using simple heuristics as not all of them are active, relevant or of sufficient quality. The feed crawler has inspected about 1 100 000 RSS feeds in its lifetime. The list was bootstrapped from publicly available RSS compilations. The sources are not limited to any particular geography or language.

Besides the RSS feeds, we use Google News (`news.google.com`) as another source of articles. We periodically crawl the US English edition and a few other language editions, randomly chosen at each crawl. As news articles are later parsed for links to RSS feeds, this helps diversify our list of feeds while keeping the quality high.

We also support additional news sources with custom crawling methods. In the scope of past and ongoing research projects, we have integrated into this platform private news feeds from Slovenska Tiskovna Agencija (STA), Bloomberg, Associated French Press (AFP), Deutsche Presse-Agentur (DPA), Telegrafska agencija nove Jugoslavije (TANJUG), Austria Presse Agentur (APA), Hrvatska izvještajna novinska agencija (HINA), Agenzia Nazionale Stampa Associata (ANSA), Associated Press

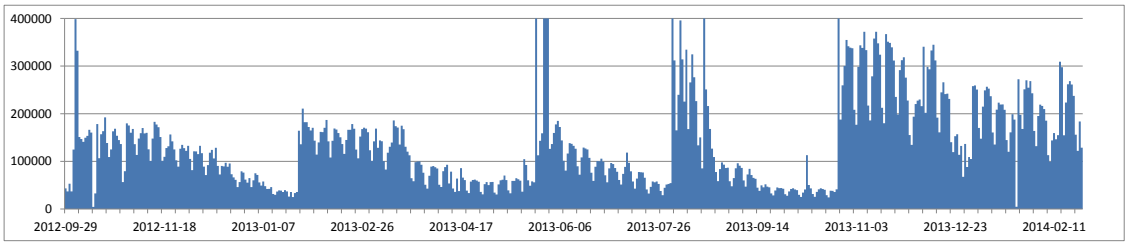


Figure 2.1: The daily number of downloaded articles from late 2012 to early 2014. A weekly pattern is nicely observable. The large-scale sawtooth pattern (large jump followed by exponential decay) is a consequence of occasional batch expansions of the RSS feed list, followed by gradual automatic weeding out of the poorly performing feeds.

(AP) and more. The contents of these feeds are commercially sensitive and needed to be made available only to a few people, so NewsFeed also implements a granular access control system.

We also ingest the public, 1% uniform sample of the Twitter stream and make it available in the same format as all other news. However, tweets skip almost all preprocessing steps for performance reasons. We also do not use them in methods described in this thesis, so all other paragraphs refer exclusively to non-Twitter data.

2.4.2.2 Data Volume

The crawler currently downloads 150 000 to 250 000 news articles per day which amounts to roughly several articles per second. Since May 2008, about 160 000 000 articles have been downloaded. See Figure 2.1 for the daily number of downloaded articles over an extended period of time.

We have observed that the problem with acquiring more data lies mostly with finding news sources of sufficient quality, rather than with scaling the system. Even with current data, it is often desirable to work only on higher-quality sources (e.g. without blogs), which cuts the volume by about 50%. The lack of a more fine-grained and automatically-updating quality control subsystem is currently a limitation of NewsFeed. We do disable feeds that are often offline or provide no new content for a substantial amount of time.

The median and average article body lengths are 1750 and 2200 characters, respectively.

2.4.2.3 Language Distribution

The downloading pipeline is agnostic with regards to the language of the articles it downloads. However, some languages are naturally better represented or more discoverable via RSS. Currently, 36 languages reach an average daily volume of 200 articles or more. English is the most frequent, representing roughly half of the

articles. German, Spanish, French and Chinese are represented by 3 to 10 percent of the articles. Table 2.2 gives a more detailed breakdown.

English	49.05%	Arabic	1.12%	Serbian	0.28%
German	9.19%	Finnish	0.82%	Catalan	0.27%
Spanish	8.04%	Romanian	0.73%	Ukrainian	0.27%
French	4.93%	Korean	0.67%	Slovak	0.26%
Chinese	4.22%	Croatian	0.67%	Hebrew	0.23%
Italian	2.74%	Tamil	0.61%	Persian	0.21%
Russian	2.51%	Norwegian	0.55%	Danish	0.21%
Swedish	2.15%	Greek	0.52%	Bulgarian	0.19%
Dutch	2.09%	Hungarian	0.48%	Latvian	0.14%
Turkish	1.69%	Slovenian	0.47%	Vietnamese	0.10%
Japanese	1.31%	Polish	0.44%		
Portuguese	1.26%	Czech	0.42%		

Table 2.2: Relative volume of languages in NewsFeed output; languages with less than 0.10% of total volume are omitted. In absolute terms, 0.10% corresponds to very roughly 200 articles per day.

2.4.2.4 Responsiveness

We poll the RSS feeds at varying time intervals from 5 minutes to 12 hours depending on the feed's past activity. Google News is crawled every two hours. Precautions are taken not to overload any news source with overly frequent requests.

Based on articles with known time of publication, we estimate 70% of articles are fully processed by our pipeline within 3 hours of being published, and 90% are processed within 12 hours.

2.4.3 System Architecture

Figure 2.2 gives a schematic overview of the architecture. The pipeline starts by providing a seed set of RSS URLs of reliable publishers. This process happens manually and only sporadically. The RSS crawler continuously monitors the RSS feeds, which in turn contain a list of news article URLs and some associated metadata, such as tags, publication date, thumbnail images etc. Articles that are found in the feeds but not yet present in the database are added to a download queue. Metadata is also stored whenever found in the RSS.

A separate component periodically retrieves the list of new articles and fetches them from the web. The complete HTML is sent to a set of cleaning processes over a message queue. The cleaning process normalizes the HTML into UTF-8 encoding, determines which part of the HTML contains the useful text (see Section 2.4.4), and discards all boilerplate text and all HTML tags. Finally, a language classifier is used to determine the primary language.

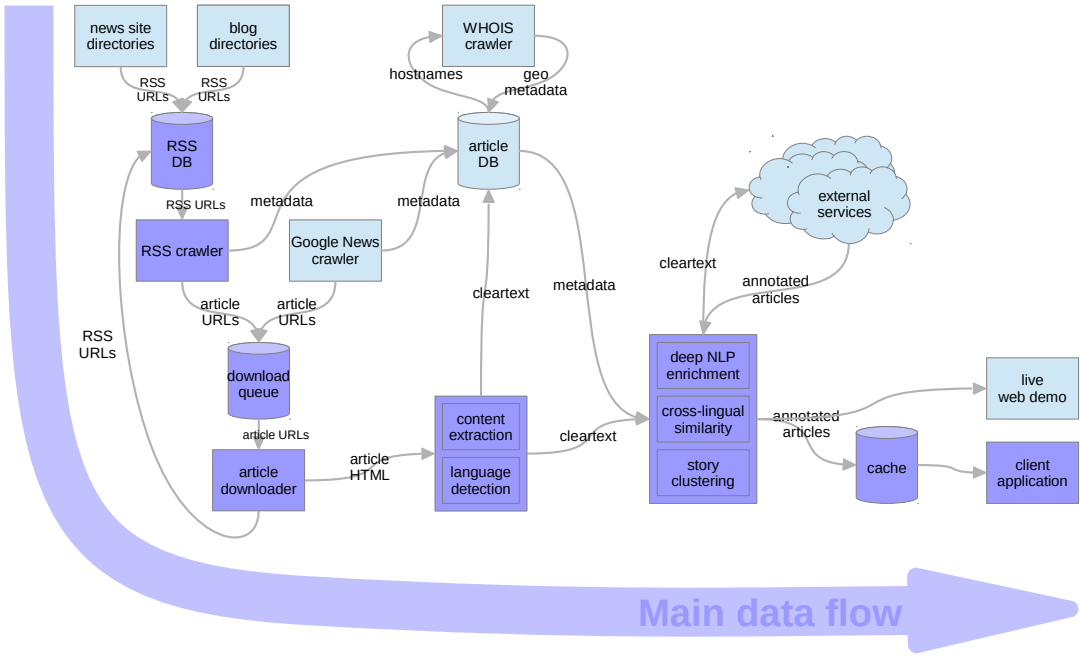


Figure 2.2: Architecture of the NewsFeed system. The darker-shaded elements of the diagram comprise the main part of the real-time pipeline.

The cleaned version of the text is stored back in the database, and sent over a message queue to a sequence of text enrichment services (see Section 2.4.5). These cluster articles into stories and perform various natural language processing (NLP) tasks like named entity resolution (NER), topic categorization and sentiment detection. Each of the services is called from multiple threads; NewsFeed also robustly and gracefully handles spikes in data volume (by using buffers and queues) and volumes that exceed the services' capacities for a longer time (by skipping enrichment if necessary).

The fully annotated articles are then exposed as a Javascript EventSource (an HTTP-based standard for streaming updates to interactive web pages) for the visual demo (see Section 2.4.7) and also made available to end users via an API.

It is notable that the whole system runs on only three physical machines: one for crawling and coordination of services, one for deep NLP, and one that serves as the database host.

2.4.4 Extracting Cleartext from Web Pages

Data preprocessing is an important part of the pipeline, both in terms of the added value provides and in terms of challenges posed by the data volume. The articles themselves are certainly useful, but almost any automated task dealing with them first needs to transform the raw HTML into a form more suitable for further processing. We therefore perform the preprocessing as a part of the data aggregations

process; this is much like the practice followed by professional news aggregation services like Spinn3r[10] or Gnip[11].

Extracting meaningful content from the HTML is the most obviously needed preprocessing step. As this is a pervasive problem, a lot has been published on the topic; see e.g. Pasternack et al. [75], Arias et al. [76], Kohlsch utter et al. [77], and the Indri project [78]. The latter also provides a mechanism for efficient indexing of text, annotations, and metadata, as well as ranking of results based on language models. We reimplemented a state of the art algorithm [75] we deemed the most promising but were disappointed by its performance; the method seems to have been evaluated on a small number of page layouts, onto which it possibly overfit. When confronted with the realistic setting of highly variable content from the internet at large, the performance suffered significantly even with newly trained weights.

We therefore designed our own algorithm which is based entirely on hand-crafted heuristics. As we demonstrate in the next section, it performs significantly better. We next describe the algorithm in broad strokes.

First, we simplify the document structure by parsing the (often standards non-conformant) HTML into a normalized Document Object Model (DOM) structure and removing some of the elements that are clearly not part of the article body:

- Remove `<script>` and `<script>` elements and HTML comments.
- Remove certain HTML5 elements like `<figure>` and `<aside>`.
- Remove hidden elements. For simplicity, we only consider the presence of `visibility:hidden` or `display:none` in inline CSS.
- Remove DOM elements with "suspicious" IDs or class names like `navigation`, `sidebar`, `social` etc.

First, we verify if the document contains an element with the attribute `itemprop='ArticleBody'`. If it does, we remove all tags (not elements!) from the content of that element and return it as the article body. This is consistent with the *schema.org* micro-tagging standard that publishers are encouraged to use precisely for the purpose of simplifying automated extraction of content from web pages.

If there is no explicit *schema.org* markup – as is most often the case – then we resort to Algorithm 2.1. The core idea of the heuristic is to take the first large enough DOM element that contains enough "promising" `<p>` elements. Failing that, take the first `<td>` or `<div>` element which contains enough promising text. The heuristics for the definition of "promising" rely on metrics utilized by other papers (see beginning of this section) as well; most importantly, the amount of markup within a node.

Importantly, none of the heuristics are site-specific and work across tens of thousands of publishers. For the parameters in Algorithm 2.1, we use $C_P = 40$, $C_T = 30$, $C_\Sigma = 350$. The parameters were determined experimentally and are reasonably robust, so no special provisions are made for different languages and alphabets. Note

[10] `www.spinn3r.com`

[11] `www.gnip.com`; acquired by Twitter in April 2014.

that it is possible for the algorithm to return `NULL` if no convincing content is found; this is a feature not commonly found in related work, but is very important in the face of inevitably noisy input. Another advantage of using heuristics is that we were able to manually verify they work well for a handful of the most important publishers and adjust them if needed or even introduce special cases.

Our approach includes a separate heuristic for extracting the article title. This consists of finding a single `<title>` element, a `<meta name="title">` or a `<title>`, whichever succeeds first. The title candidate is further stripped of potential inclusions of the site name, e.g. "`First medal for Tanzania in Sochi | BBC Sports`". In the scope of NewsFeed, title extraction is not very important because the title is almost always given in the RSS feed from which we learned about the article.

Algorithm 2.1 Extracting article body from an HTML article.

Input: Article HTML; constants C_*
Output: Cleartext version of article body

1: $CBP \leftarrow \{$`<p>` elements that contain $\geq C_P$ characters and $\leq C_T$ nested tags per cleartext character.$\}$ ▷ "Content-bearing paragraphs"
2: $CBP \leftarrow CBP \cup \{$paragraphs immediately surrounded by two $p \in CBP\}$
3: $R \leftarrow \emptyset$ ▷ Return value
4: **for all** elements e **do**
5: $\quad P \leftarrow \{p : p \in CBP \wedge \text{child}(p, e)\}$
6: $\quad$ **if** $\text{textLength}(P) > C_\Sigma \wedge \text{textLength}(P) > 2 \cdot \text{textLength}(R)$ **then**
7: $\quad\quad R \leftarrow P$
8: $\quad$ **end if**
9: **end for**
10: **if** $R = \emptyset$ **then**
11: $\quad$ **for all** `<div>`, `<td>` elements e **do**
12: $\quad\quad b \leftarrow |\{$`<a>`, `<img>` elements in $e\}|$
13: $\quad\quad$ **if** $\text{textLength}(P) > C_\Sigma \wedge \text{textLength}(P) > C_T b$ **then**
14: $\quad\quad\quad R \leftarrow \{e\}$
15: $\quad\quad$ **end if**
16: $\quad$ **end for**
17: **end if**
18: **return** R with discarded HTML tags

2.4.4.1 Evaluation

We compared our cleartexting algorithm with two versions of a state of the art algorithm [75] that would, according to the authors, have won the CleanEval 2007 challenge with a statistically significant lead. We refer to the evaluated algorithms with the following acronyms:

- **WWW** — An improved version of the algorithm by Pasternack and Roth [75]. We chose it for of its simplicity and reported state of the art performance. The

algorithm scores each token (a word or a tag) in the document based on how probable it is to comprise the final result. The scoring is done with learned weights over a simple feature set: the string value of the token itself and the two tokens that follow it, plus the name of the current HTML element. The algorithm then extracts the contiguous token subsequence with the maximum sum of scores. For this comparison, we improve the algorithm so that it extracts *two* most promising contiguous chunks of text from the article to account for the fact that the first paragraph is often placed separately from the main article body. We observed an improved performance after this change.

- **WWW++** — A combination of WWW and heuristic pre- and post-processing to account for the most obvious errors of WWW. For instance, preprocessing tries to remove user comments based on HTML element's class names and IDs.
- **DOM** — Our heuristics-based approach described above.

All the heuristics were developed on a set of articles completely separate from the evaluation dataset.

We tested the initial algorithm on a newly developed dataset of 150 news articles. Each of these comes from a different web site, which is a crucial property for deriving a measure of performance relevant to real-world applications. The dataset of 150 articles is divided into 3 sub-datasets of 50 articles each:

- **english** — English articles only.
- **alphabet** — Non-English articles using an alphabet, i.e. one glyph per sound. This includes e.g. Arabic.
- **syllabary** — Non-English articles using a syllabary, i.e. one glyph per syllable. This boils down to Asian languages. They lack word boundaries and have generally shorter articles in terms of glyphs. Also, the structure of Asian pages tends to be slightly different.

In addition, about 5% of input pages in each of the sub-datasets are intentionally chosen so that they do not include meaningful text content. This is different from other data sets but very relevant to our scenario. Examples are paywall pages and pages with a picture or video accompanied by a single-sentence caption or comment.

We evaluated the algorithms in a pairwise setting by comparing per-article performance. For each input document, we compared outputs of two algorithms side by side (the comparison was blind) and marked which of the two outputs, if any, we considered to better capture the body of the page. Guidelines for evaluating performance are given in the descriptions of categories *perfect*, *major overlap*, and *garbage* on the next page. The results are given in table 2.3.

The differences between the algorithms are statistically significant with a 5% confidence interval, with WWW++ performing better than WWW and DOM performing better than WWW++. We did not directly compare WWW and DOM to

	Number of articles where each algorithm performs better[12]			Number of articles where each algorithm performs better		
	WWW	tie	WWW++	WWW++	tie	DOM
english	2	43	**4**	7	34	**8**
alphabet	4	37	**8**	6	36	**7**
syllabary	0	44	**6**	2	12	**32**

Table 2.3: Pairwise performance comparison of webpage body extraction algorithms. The better-performing algorithm is marked in bold.

save time; it was clear from an informal inspection of outputs that the "better-or-equal" relation between algorithms is transitive for most test cases and that DOM would be certain to score significantly higher. DOM is therefore our algorithm of choice in NewsFeed.

We can see that WWW++ and DOM perform comparably on alphabet-based pages (including English). A qualitative comparison of outputs shows that in the cases where DOM performs more favorably, WWW++ tends to include irrelevant snippets interspersing the text (e.g. advertisements) whereas DOM correctly ignores them. In contrast, DOM fails relative to WWW++ mostly on short documents and documents with extreme amounts of markup; DOM can be overly cautious and declare there is no content, whereas WWW++ extracts the correct text with potentially some additional noise. For NewsFeed, the accuracy/recall tradeoff of DOM is preferable.

For DOM, we additionally performed an analysis of errors on all three sub-datasets. As the performance did not vary much across sub-datasets, we present the aggregated results. For each article, we manually graded the algorithm output as one of the following:

- **Perfect [66.3%]** — The output deviates from the golden standard by less than one sentence or not at all: a missing section title or a superfluous link are the biggest errors allowed. This also includes cases where the input contains no meaningful content and the algorithm correctly returns an empty string.

- **Major Overlap[22.1%]** — The output contains a subset or a superset of the golden standard. In vast majority of the cases, this means a single missing paragraph (usually the first one which is often styled and positioned on the page separately) or a single extraneous one (short author bio or an invitation to comment on the article). A typical serious much rarer error is the inclusion of visitors' comments in the output; this achieves small overlap with gold and falls into the next category.

- **Garbage [5.8%]** — The output contains mostly or exclusively text that is not in the golden standard. These are almost always articles with a very short body and a long copyright disclaimer that gets picked up instead.

- **Missed [5.8%]** — Although the article contains meaningful content, the output is an empty string, i.e. the algorithm fails to find any content.

Another way of comparing our method with alternatives is to interpret "Perfect" and "Major Overlap" (where the outcome is most often only a sentence away from the perfect match) as a "Positive" score, both precision and recall for DOM are 94%. This (article-based) metric is roughly comparable with the word- or character-based metrics employed in several other papers on state of the art methods [77]; those also report precision and accuracy of 90–95% depending on the algorithm and evaluation dataset.

In addition, our method has been evaluated informally through continuous use in the last 4 years, an unusual setting for academia. An estimated 100 million articles from tens of thousands of sources have been processed with it and the resulting cleartext used in various projects. During that time, only a few adjustments and improvements to the heuristic rules were needed. For all practical purposes, the quality of data is high enough, with one notable exception. On some of the domains / site layouts, the algorithm erroneously selects a lengthy copyright or similar notice as the article body. Alternatively, it appends the notice to the true body. The solution is to make the algorithm aware, as it is cleaning an article, of previous articles coming from that domain. As the copyright notices do not change or change very infrequently, they would be easy to detect. We also verified this with a quick informal experiment. Implementing this reliably and at scale would require somewhat bigger changes and remains a task for the future.

2.4.5 Deep NLP and Enrichment

Deep analysis and annotation of text are of high significance for this thesis as they (at least partially) transform unstructured text into structured, potentially semantic representations with which we wish to operate.

NewsFeed integrates several plaintext enrichment/annotation services. Here, we give a brief overview of all of them; those that are particularly relevant for this work are discussed in detail later, in the context in which they are used. The annotations represent significant added value for NewsFeed as an enabler technology, and a step beyond what is currently available from several commercial news aggregation services, for example Gnip's social media stream[13] or Bloomberg's Terminal[14] with financial news (among other data).

One of the most important annotations is the **clustering of articles into stories**. Clustering stream-based data with a temporal dimension requires special consideration; lately, a number of approaches have been proposed [79]. In NewsFeed, articles are clustered with a service provided by Flavio Fuart, based on the method proposed by Azzopardi and Staff [80]. "Our" service adds some additional improvements and tweaks to the algorithm to make it better suited to NewsFeed data. In

[13] `gnip.com`
[14] `bloomberg.com/professional`

particular, it deletes news stories when the last update is older than eight hours, it keeps stories hidden until they contain at least five non-duplicate articles from different sources, and it deletes stories that stretch over a span of more than ten days.

Alternatively, the Google News crawler (see Figure 2.2) also stores the clustering information from Google News for a small fraction of articles. For historical reasons, we use those clusters for evaluation purposes.

Akin to clustering, but operating *across* languages, is the **cross-lingual similarity module** developed by Jan Rupnik and Andrej Muhič [81]. For each document, it returns a list of the most similar recent documents in each of the major languages[15]. The method is an extension of Canonical Correlation Analysis (CCA) and works by finding a latent low-dimensional vector space into which it maps all major languages and then performs similarity computations there.

For most of the **semantic processing**, we use Enrycher [82], a pipeline of annotation services in its own right that merges all the annotations into a unified output format. Enrycher supports the following operations:

- **Tokenization** via GATE (see Section 2.3.4).
- **Part Of Speech (POS) tagging** via GATE.
- **Named Entity Recognition** for locations, people and organizations; again via GATE.
- **Named Entity Resolution** — linking entities to DBpedia and several other ontologies. Performed with an algorithm by Štajner [83].
- **Constituency and dependency parsing** via Stanford parser (see Section 2.3.5).
- **Semantic triplet extraction** from parse trees; details are given in Chapter 3.2.
- **Sentiment detection** (positive, negative) with a method by Štajner outlined in Section 5.2.5.
- **Topical classification** into DMOZ[16] with a method by Grobelnik [84].

Most of the Enrycher services are only available for English, and some of them for Slovene as well. In scope of the XLike EU project[17], NewsFeed has been extended with partners' services that provide Enrycher-like functionalities for Spanish, Catalan, German and Chinese.

[15] About 10 languages are currently output, but there is support for 90 more.

[16] Mozilla Open Directory Project (`http://dmoz.org`) provides a large general-purpose hierarchy of topic categories like `Sports` → `Soccer` → `Competitions` → `World_Cup`

[17] urlwww.xlike.org

We also detect the language with the Compact Language Detector library from Google which is reported to have an error rate of 1% or less [85].

The present data download rate of a few articles per second is nothing extreme, especially if we consider scaling to multiple processing nodes; however, it is nontrivial in that adding complex preprocessing steps (e.g. full syntactic parsing of text) or drastically increasing data load (e.g. including a 10% sample of the Twitter feed) would turn preprocessing into a bottleneck and require scaling the architecture.

2.4.6 Data Distribution

Upon completing the preprocessing pipeline, contiguous groups of articles are batched and each batch is stored as a gzipped file on a separate distribution server. Files get created when the corresponding batch is large enough (to avoid huge files) or contains old enough articles. End users poll the distribution server for changes using HTTP. This introduces some additional latency, but is very robust, scalable, simple to maintain and universally accessible. Independent of this server-side, filesystem-based cache, a complete copy of the data is still kept in the traditional structured database (see Section 2). This is the only copy guaranteed to be consistent and contain all the data; from it, the XML files can be regenerated at any time. This is particularly useful in case of XML format changes and/or improvements to the preprocessing pipeline.

2.4.7 Monitoring

A complex, constantly running system like NewsFeed is bound to encounter partial or total outages during its operation. We implemented a central event logging module through which we export performance indicators to CSV files, a local Graphite service for real-time graphing[18], and the Leftronic online graphing service[19]. A sample screenshot of the latter is provided in Figure 2.3. In addition, we monitor service uptime with Pingdom[20], an external service for availability monitoring and alerting.

For informal inspection of the pipeline's output, we have a demo web site that displays articles in real time as they complete all the processing stages. The web site displays the stream of articles with text, a cleartext snippet and a thumbnail image, and shows the locations of publishers and of stories being covered on the world map. See Figure 2.4.

[18] `http://launchpad.net/graphite`
[19] `http://leftronic.com`
[20] `http://pingdom.com`

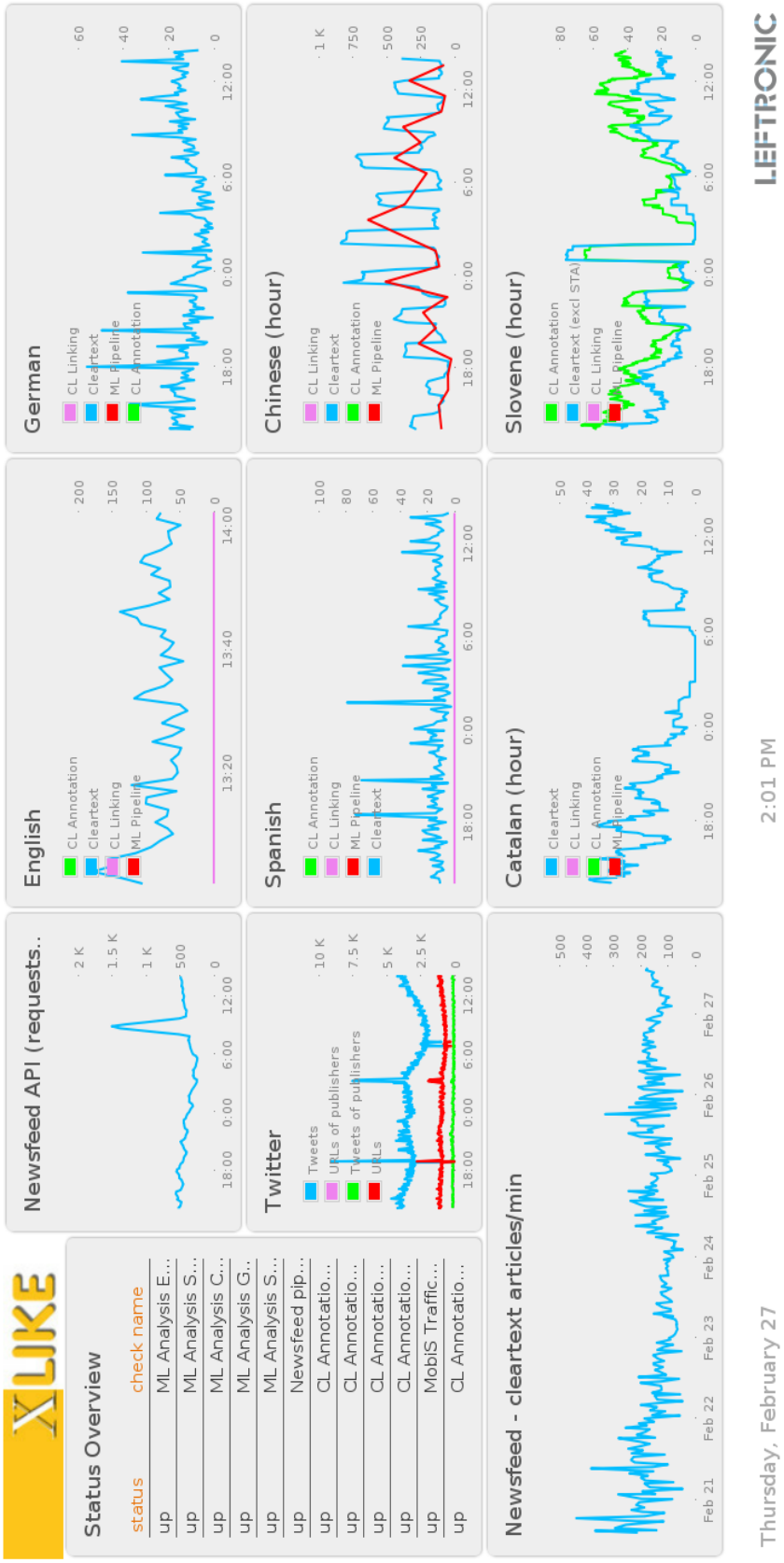


Figure 2.3: A subset of NewsFeed performance indicators monitored via the Leftronic online service. The graphs mostly show the per-minute article volume that passes through various stages of the pipeline, and the error rates. The upper left corner shows the availability status of processing stages/services.

Figure 2.4: A real-time web demonstration and informal monitoring tool of NewsFeed's output, demonstrating some of the annotations. See `http://newsfeed.ijs.si/visual_demo/` for a live demo.

Chapter 3

Semantic Representations of Text

Strictly speaking, the title of this chapter is an oxymoron: for a truly semantic representation of data, it should be largely irrelevant what the original representation of that data was – text, table, image, video or otherwise. However, it is unrealistic to expect from today's methods to be able to produce such true abstractions. Instead, the semantic representations of data often contain telling traces of their original form, and textual data is no exception. This is because we are only able to extract a part of the semantics from raw data, and the type of data dictates what semantics we can obtain most reliably.

With text, we can divide the semantics into two broad categories[1]:

- **Discourse.** This category encompasses all the facts directly expressed by the text itself. There is no single standard way in which to encode them, and often, we only encode parts of these facts to simplify extraction, representation and handling of the data. Representations range from simple lists of entities appearing in the text to complex logic languages like CycL that encode "everything" encoded by the natural language[2]. We opt for the middle ground in terms of complexity and expressiveness, the so-called *semantic triplets* and *frames*. We discuss them in Section 3.1 and describe how to obtain them in Sections 3.2 and 3.3.

- **Metadata.** In this category, we consider all properties that talk *about* the text. This includes *emergent* data, like the topic of the text or its sentiment, as well as "standard" metadata not directly discernible from the text, like the author and the time and place of its creation. We briefly discuss metadata and its semantic encodings in Section 3.5.

[1]There are many more, as philosophers would be happy to point out. Here, we limit ourselves to those we can currently hope to extract with automated processing.

[2]Even the fullest representations are not able to capture most of the narrative semantics and other finer points, e.g. the level of politeness, the affect, sarcasm, joking, word play etc.

3.1 Semantic Modeling of Discourse

Converting natural language text into a semantic form, or, more colloquially, "understanding text" or "machine reading", is a hard task that has not been solved yet at all in its entirety. When developing partial solutions, stepping stones to the ultimate goal, at least two important choices need to be made: what part of the text's semantics to extract, and what formalism to use to represent the results. The formalism typically constrains, to some degree, the types of statements we can express, so the two choices are made hand in hand.

Take, for example, the sentence "Nelson Rolihlahla Mandela, the father of our nation, has died at the ripe age of 95." If we wanted to convey the full information content of that sentence, we could break it down into the following simple statements:

1. We have a nation. ("our nation"; "we" is unspecified or implied by the context.)
2. Our nation has a father. ("the father of our nation")
3. Father's name is Nelson Rolihlahla Mandela.
4. He is dead. ("Mandela [...] has died")
5. He was 95 years old when he died. ("died at [...] 95")
6. Many people die before they turn 95. ("ripe age")

Simply breaking down the sentence into a series of shorter statements does not make this a semantic representation yet; however, no matter what formal logic we choose to encode the original sentence, it is certain that the simplified statements are closer to that formal language in spirit. We omit the encoding in any specific formal language here. In fact, because of the complexity of the task, not many languages exist with enough expressive power to encode our statements. CycL is probably the most prominent of those, but the encoding would take roughly one full page.

The task of fully "understanding" text in this way has been traditionally called *machine reading*, or more specifically *micro-reading*, a term popularized recently by Mitchell et al. [86]. Even from this short example, we can see that it requires knowing a lot about the context and the language itself: it is not self-evident that "Mandela" is a name; or that "95" denotes years and not, say, minutes; or that "ripe age of 95" means Mandela lived a relatively long life; or indeed even that the word "died" denotes the concept of ceasing to live.

Despite micro-reading being a long-standing goal in text mining, there are no automated approaches to it yet, and even semi-automated annotation is prohibitively expensive. In 2004, Vulcan Inc. organized the Project Halo challenge, in which three teams attempted to encode the full text of a biology textbook; the cost came to about $10 000 per page [87, 88]. By 2009, the teams managed to reduce the cost to about $1 000 per page [89, 88], which is a clear step forward but also still far from commercially viable for e.g. routine processing of news articles. It should

also be noted that encoding text from a narrow domain like high school biology is significantly easier than encoding text from the open domain.

In practice, we therefore forgo hopes of extracting everything and focus on only the most important pieces. At the extreme end of this simplification spectrum, there is *Named Entity Recognition and Resolution (NER)*, where we constrain our representation of a text to listing the key named entities appearing in it and disambiguating them against a knowledge base. Closely related is *Word Sense Disambiguation (WSD)*, which aims to achieve essentially the same goal, but focuses on regular dictionary words and is often implicitly understood to skip named entities. Combining these two approaches gives a weakly semantic representation where we "understand" what the individual words mean, but do not "understand" the relations between them.

Fully "understanding" arbitrary relations, i.e. encoding them in a semantic way, is again hard. We therefore propose a compromise and extract only *some* relations — those that are expressed relatively explicitly in the language. In particular, we can focus on **subject—verb—object** (SVO) relations: a grammatical subject and a grammatical object, related by the action implied by the grammatical verb. For example, the sentence "Yesterday, when walking downtown, Sally noticed the mysterious man again." would produce $\boxed{\text{Sally}} \xrightarrow{\text{notice}} \boxed{\text{mysterious man}}$. Identifying grammatical subjects, objects and verbs is not beyond the state of the art; we describe the extraction of such relations in Section 3.2.

Can we do better? There is ample existing work on extracting more key constituents from a sentence than just the subject, verb and the object. In the task of Semantic Role Labeling, a sentence is represented by a set of **frames**. Each frame typically characterizes an action and is described by a set of **frame roles**. For example, the above sentence could be described with the following instance of the *Observing* frame:

Observing	
Target[3]	"noticed"
Observer	"Sally"
Observed	"mysterious man"
Location	"downtown"
Time	"yesterday"

Note that the frames offer a significantly richer representation: the frame name (Observing), the Observer and the Observed provide exactly the information from the subject—verb—object relations. In addition, they are not bound to specific grammatical patterns; for instance, "The sudden glimpse of the mysterious man yesterday, while visiting downtown, made Sally uneasy." describes the act of observing with the *noun* "glimpse", but would still produce the same *Observing* frame (except for *Target*="glimpse").

The "skeleton" of the *Observing* frame — its existence and the list of its possible

[3]The word that triggers, evokes the *Observing* frame.

roles — needs to be predefined so that the information is structured using a fixed vocabulary. Therefore, the frames are advantageous in that we can capture more of the original information, but disadvantageous in that an extensive pre-populated knowledge base of frames and frame roles is required.

It is feasible to try and obtain a semantic representation of text based on either the subject—verb—object model or the frame model. Regardless of which one we focus on, there are two key components to each sentence that we need to semanticize:

- **Constituents.** Single words or short word phrases (mostly nouns, but potentially also verbs, adjectives, and adverbs) need to be aligned to a dictionary-like background knowledge base. For example, if our KB is WordNet, we might map "drink" to `beverage.n.01`; if our KB is Wikipedia, we might map it to `en.wikipedia.org/wiki/Drink`.

- **Structure.** The way in which the constituents relate to each other also needs to be encoded in terms of some formal notation.

These two tasks can be performed at different levels of complexity and expressiveness, depending primarily on the background KB of choice. The more complex our KB, the harder it will be (in general) to map text onto it while fully taking advantage of it features. At the same time, a more complex KB theoretically allows us to lose less information during the semantization of text, as well as making the semantic data more valuable by providing more background information about it (linking it to more concepts, expressing more complex relations about it, etc.).

We consider two approaches to text semantization, with varying levels of complexity:

- The **Simplified Dependency Parse (SDP)** method is a simple and robust method based on dependency parsing. It represents text as a set of lightweight frames with a simple, frame-independent set of roles that go only a small step beyond the subject–verb–object model. Each frame is defined and triggered by a verb. Role fillers are mapped to WordNet, which is the only KB used in this method. We give details in Section 3.2.

- The **Mapped Semantic Role Labels (MSRL)** method is more ambitious, representing text with frames derived from classic Semantic Role Labeling (SRL). The knowledge bases used in this method are FrameNet (which provides labeled training data for SRL and well-defined frames, but contains very little background knowledge) and Cyc (which is rich in background knowledge but provides very little training data for mapping natural language onto Cyc). The method first maps natural language to FrameNet, then uses a concept mapping to represent frames in Cyc. Role fillers (mostly nouns) are mapped to Cyc directly from natural language. We describe the MSRL method in Section 3.3.

A comparison and discussion of the methods are given in Section 3.4. We find the loss of accuracy associated with taking the more complex approach (MSRL) to outweigh the potential advantages.

3.2 Simplified Dependency Parses (SDP)

This section describes the robust method of extracting simplified semantic frames from text, based on simplifying the output of existing dependency parsing methods. As such, the frames rely relatively heavily on the sentence structure to reveal the role of its constituents. The method identifies a fixed set of roles:

- *Verb*, usually simply called the *frame name* in related work and resources. This is what identifies the action performed in the frame. In SDP, frames are always identified by the grammatical verb of a sentence.
- *Subject*, also called *Agent* or *A0* in related work and resources. This is always the grammatical subject of a sentence.
- *Object*, also called *Patient* or *A1* in related work and resources. This is always the grammatical object of a sentence.
- *Instrument*, a physical entity used to performed the action described by the frame. *Patient* or *A1* in related work and resources.
- *Time*, specifying when the action described by the frame happened.
- *Location*, specifying where the action described by the frame happened.

SDP creates frames that are verb-centric in that the method does not map to a dedicated repository of frames like FrameNet: the roles are fixed as described above, and the set of possible frames is defined by the set of WordNet verb synsets. In addition, each frame is necessarily associated with a verb in the original text.

Not all roles apply to all frames. For example, a frame with *verb*="sleep" cannot sensibly have an *object*, and many frames cannot have a sensible *instrument*.

We next describe the technical details of the SDP method.

Extracting frames. Starting with plain text, we first annotate it with some basic semantic and linguistic information. Using the ANNIE tool from the GATE framework [70], we first detect **named entities** and tag them as person, location or organization. We next use Enrycher [83] to perform **coreference and pronoun resolution** ("Mr. Obama", "President Barack Obama" and "he" might all refer to the same entity within an article).

Finally, we use the Stanford parser [72] to obtain **dependency parses** for individual sentences. The accompanying manual [90] describes, among other things, the types of relations extracted by the parser. We simplify the parse trees using the following steps:

- For noun phrases, retain only the head of the phrase. The head is identified by the dependency parser.
- Convert passive to active voice.
- Convert object-like relations (`dobj` (direct object), `acomp` (adjectival complement), `infmod` (infinitivial verbal modifier), `nsubjpass` (passive nominal subject)) to a simple *object* role.
- Convert subject-like relations (`nsubj` (nominal subject), `agent` (passive verb agent), `xsubj` (controlling subject)) to a simple *subject* role.
- Convert the prepositional dependency tree relation (`prep`) to a *time*, *location*, or *instrument* role or ignore it. The mapping is done based on ANNIE annotations: if a dependent was annotated as a time or location expression, we introduce it into the frame as such. If no ANNIE annotations exist, we fall back onto a static list of prepositional modifiers that are known to introduce a dependent of a certain type (e.g. "above", "across", "below" etc. for *location*).

The *subject* and *object* roles can only be filled with nouns. Verbs with no recognizable dependents (roles) are ignored. These rules are intentionally restrictive to increase the reliability of those frames that do get extracted under the rules. As a consequence, noun-based phrasings, sentence fragments and several other constructs are recognized poorly or often not at all. For example, the sentence "My brother was sleeping." produces a frame (*verb*="sleep", *subject*="brother") while "My brother's sleep was short." does not. Similarly, the relatively long sentence "It was Diouf's second goal of exhibition play, both of which were game-winners." produces no frames under the constraints of SDP.

Aligning role fillers to WordNet. As discussed in Section 3.1, we also need to resolve the role fillers against a background knowledge base. With SDP, the knowledge base of choice is WordNet. Note that as a part of the frame extraction process, we already retain only a single word per role filler, except for the verb where we possibly retain a preposition (e.g. "take up" instead of just "take"). This is another compromise between simplicity and expressiveness. Although causing us to lose some information, it greatly simplifies especially the mapping of (now single-word) role fillers to WordNet and their representation.

As a first step, we lemmatize all words using the Morphy morphological processor. We then try to find the corresponding KB concept ("synset" in WordNet terminology) for each lemma. We make use of the POS annotations for the natural text and search for exact lexical matches (ignoring capitalization) among synsets from the appropriate POS group. If more than one synset matches, we choose the most common sense; this is a proven approach and a very strong baseline for unsupervised word sense disambiguation [91]. If no synset matches, we create a new one on the fly, expanding our local copy of WordNet. If the word for which the new

concept was created (e.g. "Obama") was previously tagged by GATE as a person, location or organization, the new synset's hypernym is set accordingly. The new concepts are retained between algorithm runs.

For multi-word verbs (e.g. "take up"), if no synset matches, we first try to map only the head word ("take"). If that fails too, we proceed as described above.

Improving efficiency. A notable downside to using parse trees is that parsing is a costly process in terms of processor time. Since we never make use of the full parse tree, others have tried more efficient approaches based on Part of Speech (POS) tagging or chunking[4]. In particular, Dali et al. [92] experimented with both learned and hand-crafted relation extraction rules on top of POS tags. Unfortunately, they found that the models have a relatively low performance (40% F_1 score on triplet extraction; by comparison, the Stanford parser [72] creates labeled parse trees with a per-edge F_1 of roughly 85%, as a ballpark estimate, we would expect a good triplet (= two relations) extractor to reach about $85\%^2$=70%).

The speed of parsing is currently at about 2 sentences (40 words) per second on a commodity server with 12 cores. For the domain template construction methods presented later in this thesis (see Section 4, this represents the grand majority (over 90%) of total processing time and is the clear performance bottleneck.

3.3 Mapped Semantic Role Labels (MSRL)

Section 3.2 attempted to extract sentence-level frames using only relatively highly reliable, simple transformations of parse trees. Such an approach is inevitably only able to capture semantics expressed with a restricted set of syntactic patterns.

In this section, we describe an approach that tries to cover a wider range of syntactic expressions as well as use better defined frames with a broader range of roles. The main idea is to use known approaches and existing resources for the task of Semantic Role Labeling which is ideally suited to our goal of semantifying text.

To the best of our knowledge, only two SRL knowledge bases come with non-negligible amounts of annotated data: PropBank [93] and FrameNet [63] (see also Section 2.3.2). We opt for FrameNet because PropBank's frames are predominantly verb-based (each dictionary verb defines its own frame) and have a limited set of roles, resembling in many ways the output of the SDP approach. Section 3.3.1 describes how we map text to FrameNet frames, following best practices from previous work.

A limitation of FrameNet (as well as PropBank) is that it is a repository of frames only; it contains (almost) no concepts corresponding to entities and there is nothing inside FrameNet against which to map role fillers. We therefore map them to Cyc (introduced in Section 2.3.1). We chose Cyc over WordNet for two

[4]Chunking is a NLP task of identifying noun phrases and verb phrases but *not* relating them to each other. It produces, in essence, the leaves of a heavily pruned parse tree.

reasons: first, with the MSRL approach we wish to gauge the potential benefits of extracting a richer structure from text in comparison to SDP, and Cyc provides a much more complex ontology than WordNet. Second, Cyc's upper ontology layers contain analogues to the frames themselves, meaning that with Cyc we can encode both the frame structure and its contents within a single formalism.

3.3.1 Semantic Role Labeling

The task. Semantic role labeling (SRL) is a well-established text processing task in which the goal is to mark up text with a predefined set of frames and frame elements, also called roles. A frame is defined [63] as any system of concepts (*roles*) related in such a way that to understand any one concept it is necessary to understand the entire system.

Examples of frames are *Addiction*, *Annoyance*, *Attack*, *Drinking* etc. The latter, for instance, consists of roles *Drinker*, *Fluid*, *Quantity*, *Container* and perhaps others. There are also some roles that can be included in any frame, e.g. *Location*, *Time*, *Frequency*, *Purpose* and *Manner*. Not every occurrence of a frame in natural text needs fill all the roles; for example, the sentence "[$_{DRINKER}$ Paul] took a [$_{TARGET}$ sip] of [$_{FLUID}$ red wine] from [$_{CONTAINER}$ the tall glass] and nodded approvingly." omits the Quantity role as well as all target-nonspecific roles. Note that this and other examples represent the ideal, human-produced labels which can be very hard for algorithms to reproduce because of rich grammar or metaphors ("sip of wine").

The previous sentence also illustrates the standard bracket notation for marking up frames in natural text: everything contained in square brackets is a *role filler*, i.e. a text fragment filling a specific frame role, which in turn is given in subscript in all caps. The special "[$_{TARGET}$ ...]" role is filled by the word that evokes/triggers the frame.

The *target* role of a frame is not necessarily filled by a verb; take for example the following *BiologicalUrge* frame: "[$_{EXPERIENCER}$ He] gave me a [$_{TARGET}$ tired] [$_{EXPRESSOR}$ shrug]."

The three stages of SRL. The process of automatic SRL decomposes naturally into three stages: frame identification ("which frame is evoked by the sentence?"), boundary detection ("which sentence fragments are role fillers?") and role identification ("what roles do the role fillers fill?"). Although these problems can be solved jointly, it is easier and computationally much more efficient to approach them separately. This does not affect performance: it is intuitively clear that syntactic context should suffice for frame identification, but surprisingly, performing boundary detection and role identification jointly does not bring significant gains either [25, 94]. Our method thus performs each of the three stages separately as well.

Stage 1. For the frame identification task, we use a simple recall-oriented approach. First, we make the standard assumption that frames do not extend over more than one sentence. We then consider the lemmatized version of every word w in a sentence s. If, for any frame f, the lemma w occurs in f's list of trigger words,

we consider s to contain f. Some of these decisions are revoked at the later stages if no convincing role fillers are identified for f in s.

Stages 2 and 3. For role boundary detection, we first perform full constituency parsing of sentences using Charniak's parser [95]. We then treat both remaining stages of SRL as classification tasks over the nodes of the parse tree.

Based on recommendations in existing work, we derive the following features for every node:

- Lemma of the target word
- Phrase type (= Penn Treebank tag of node)
- Governing category (= parent node's tag; helps distinguish subjects from objects)
- Path from target to node
- Position relative to target (left/right)
- Passive/active voice of sentence. A sentence is considered passive if its tree contains a path of the form AUX↑VP↓VP↓VPN.
- Lemma of node's lexical headword. The head word is derived using widely adopted rules developed by Collins [96].
- POS tag of node's headword.
- Verb subcategorization, i.e. the ordered list of children of VP immediately containing the node.

It has been shown that the choice of the classifier is not of critical importance; however, support vector machines (SVMs) are one of the most appropriate choices [97, 30]. We use a linear SVM with $C = \frac{1}{avg(|x|^2)}$ implemented in the svmlight[5] toolset. The parameters are the defaults recommended by svmlight.

For **stage 2** (role boundary detection) we use the above features and train a classifier on FrameNet's annotated data to classify parse tree nodes as either `role` or `none`. We then discard all nodes which are classified as `none` with high confidence. The threshold was set so that the on a held-out set, the discarding process was estimated to retain 95% of the true `role` nodes. This significantly speeds up the role identification step and, also very important, greatly reduces class imbalance in the remaining data.

In **stage 3** (role identification), we classify all the nodes remaining after the boundary detection stage into one of multiple classes: all the roles belonging to the frame and `noRole`. There is no clear consensus in the community on the best way to perform multi-class classification in this case, so we follow the recommendation by Hacioglu [98] and use one-vs-all rather than pairwise classifiers or multi-class SVM.

When combining the votes, we operate under two classes of constraints: the soft, local, per-node constraints suggest that each node should be assigned the class voted for with the highest confidence. Global constraints require that a role appear

[5] `http://svmlight.joachims.org`

only once in a frame and that role fillers be strictly disjoint. We therefore employ a constrained greedy algorithm to assign roles. Votes for all nodes and all classes are sorted in descending order of confidence. They are then greedily assigned one by one; if an assignment would violate either of the two aforementioned global constraints, we discard the vote.

Additionally, based on an observed algorithm bias towards selecting nodes further from the root of the tree, we adjust the votes somewhat before sorting. Let us denote by $f(v, r)$ the confidence of vote for role r on node v. If $f(v, r) > f(v, \texttt{noRole})$ and, for some child node v' of v, it holds that $f(v', r) > f(v, r)$, then we set $f(v, r) := f(v', r)$.

Minor issues. To prepare training data, we map FrameNet's annotations (based on word-level boundaries) onto parse tree nodes. In great majority of the cases, a perfect correspondence can be found; if, due to errors in parsing or due to a convoluted sentence structure, a perfect match does not exist, we map the role-filler annotation to the leftmost highest node in the tree which is completely contained in the annotation. Informal inspection shows that in English, this tends to preserve the semantic head of the role filler. Akin to most of the existing work, we build a separate set of classifiers for every frame. This could be improved by taking into account that some roles (e.g. *Place*, *Time*) are shared across frames.

In this work, we limit ourselves to frames that describe actions, e.g. *Drinking* but not *BiologicalState*. There are several reasons for this: action frames are more informative, map to Cyc more cleanly and have better annotation coverage in training data. Action frames were identified by having at least one verb trigger word and not more than 10 times as many non-verb trigger words. Of those, we discard frames with no annotated sentences. By manual inspection, we discarded a further 20 frames deemed too generic or irrelevant (e.g. *Undergoing* with the definition "An Entity is affected by an Event."). We are left with approximately 550 frames. In particular, the following frames were discarded:

Being_active An [Agent] is described as pursuing an [Activity], expending some effort

Being_operational An [Artifact], either a machine or a network of operations, is in a state ready to perform its intended function.

Change_posture A [Protagonist] changes the overall position and posture of the body.

Change_resistance An [Agent] changes a [Patient]'s ability to resist literal or figurative attack

Difficulty An [Experiencer] has an easy or difficult time carrying out an [Activity]

Event An [Event] takes place at a [Place] and [Time].

Eventive_cognizer_affecting An [Event] causes the [Cognizer] to accept some [Content]

Existence An Entity is declared to exist, generally irrespective of its position or even the possibility of its position being specified

Experiencer_obj Some phenomenon (the [Stimulus]) provokes a particular emotion in an [Experiencer].

Familiarity An [Entity] is presented as having been seen or experienced by a (typically generic and backgrounded) [Cognizer] on a certain number of occasions, causing the [Entity] to have a certain degree of recognizability for the [Cognizer].

Have_associated A [Topical_entity] has properties which are affected by the existence and association of an [Entity].

Likelihood This frame is concerned with the likelihood of a [Hypothetical_event] occurring

Locative_relation A [Figure] is located relative to a [Ground] location

Means An [Agent] makes use of a [Means] (either an action or a (system of) entities standing in for the action) in order to achieve a [Purpose].

Mental_stimulus_stimulus_focus A [Stimulus] serves to bring about an emotion of mental stimulation in an [Experiencer].

Obviousness A [Phenomenon] is portrayed with respect to the [Degree] of likelihood that it will be perceived and known, given the (usually implicit) [Evidence], [Perceiver], and the [Circumstances] in which it is considered

Predicament An [Experiencer] is in an undesirable [Situation], whose [Cause] may also be expressed.

Taking_time An [Activity] takes some [Time_length] to complete

Turning_out A [State_of_affairs] turns out to be true in someone's knowledge of the world

Undergoing An [Entity] is affected by an [Event].

3.3.2 Mapping to Cyc

As discussed in the introduction, our end goal is to obtain a semantic representation of input text. The SRL markup obtained using the method from the previous section, though, marks up syntactic constituents of the sentence. We thus still need to map the role fillers to an ontology. In general, this task is no easier than the one we started out with (mapping whole sentences), because role fillers can be whole relative clauses: for example, for frame *Drinking*, we can have the sentence "[$_{DRINKER}$ He] [$_{TARGET}$ drank] [$_{FLUID}$ the strange stink emitting potion she had concocted for him before they left for the journey]". Mapping the *Fluid* role onto a set of ontological concepts is clearly not much easier than the original task. Luckily, it is reasonable to assume that the extra properties about the potion will be identified during analysis of other frames, e.g. *Cooking*: "He drank [$_{FOOD}$ the strange stink emitting potion] [$_{COOK}$ she] had [$_{TARGET}$ concocted] [$_{PURPOSE}$ for him] [$_{TIME}$ before they left for the journey]." and *Apperance*: "He drank the strange [$_{TARGET}$ stink] emitting [$_{PHENOMENON}$ potion] she had concocted for him before they left for the journey."

Our problem therefore reduces to mapping only the headword of each role filler, which is either a noun phrase or a verb phrase. They typically consist of a single word; in other words, we are left with the task of word sense disambiguation (WSD).

In addition, our introduction of Cyc (or any other ontology different from FrameNet) requires us to map the frames and roles as well. The only way to avoid the task would be to reuse existing mappings between Cyc and FrameNet, which however do not exist. This problem is known as ontology alignment. We next describe our approach to both tasks.

3.3.2.1 Mapping Frames and Roles (Ontology Alignment)

Conceptually, it makes sense to align the ontologies before aligning role fillers, for two reasons. First, this is a task that only needs to be done once. Second, it offers support for WSD in that the ontology imposes selectional preferences and constraints on role fillers using its type system. This can aid in the role identification phase of SRL or at least be used immediately after it in a reranking postprocessing step. We do not exploit this in our approach.

(Dis)similarities between the ontologies. Of the numerous concepts found in Cyc, of special interest to us are `#$Event` and `#$BinaryRolePredicate`. Specializations of the first are a natural analogue of FrameNet's frames. Instances of the second are the analogue of FrameNet's roles. They are connected by the `#$rolesForEventType` relation which specifies which roles apply to which events. In short, the structure of that part of Cyc is quite similar to that of FrameNet[6]. The majority of frames has a natural counterpart that is a specialization of the `#$Event` concept in Cyc. We discard the frames that do not; those fall in one of the following categories:

- Frame maps to more than one Cyc concept. For example, the frame *Respond_to_proposal* (with trigger words "reject", "accept", "refuse" etc.) could map to Cyc's `#$Refusing-CommunicationAct`, `#$Accepting-Communication-Act`, `#$Rejecting-CommunicationAct` and some others, but their only common generalization is `#$CommunicationAct`, which is too general. About 5% of frames (i.e. about 50 frames) are like this.

- Concept does not exist in Cyc. For example, Adjusting (trigger words: "adjust", "tweak", "calibrate", ...). This does not necessarily mean the notion cannot be expressed in Cyc, but it would require a non-atomic expression. About 2% of frames fall into this category.

- About 2% of the frames map to relations rather than specializations of `#$Event`. For example, Evoking maps to the relation `(#$evokes ARG1 ARG2)` where `ARG1` is an instance of `#$Individual` and `ARG2` of `#$FeelingAttribute`.

With a moderate amount of additional work, frames from the last two categories could be accommodated as well, meaning that 95% of the frames we consider have a natural counterpart in Cyc. This supports our choice of the two ontologies.

It has to be noted, however, that not all mappings are perfect. In particular, we are sometimes forced to ignore certain subtleties in frame definitions. Consequently, several FrameNet frames might get mapped to the same Cyc concept. An extreme example of this is the `#$Evaluating` concept which is mapped to *Trying_out*, *Labeling*, *Regard*, *Judgment*, and *Assessing*. Another typical example of conflated frames

[6]And it would be very reasonable to perform SRL directly using Cyc as the frame ontology, were it not for a complete lack of training data.

are frame pairs of the form *Cause_to_XYZ* and *XYZ*. We map pairs like this to the same Cyc concept, but with different role mappings.

Computer-assisted mapping of frames. There are about 550 frames to be mapped from FrameNet to Cyc and about 2000 roles. To perform the mapping automatically, we have few reliable features and no training data at our disposal, so a fully automated approach is unrealistic. We opt for a mostly manual setting where an algorithm proposes several possible mappings and a human annotator chooses the best one among them.

When aligning ontologies, there are, broadly speaking, two types of features available: content-based, stemming from the attributes of the nodes themselves (typically, glosses or sample usages), and structural. In our case, aiming at aligning the two ontologies structurally does not make sense as the two have different levels of granularity and coverage. We therefore make use only of the glosses and English denotation strings of entities in both ontologies.

When mapping frames, the trigger words provided with each frame prove to be much more valuable than the frame descriptions. Our method suggests for each frame all the concepts that have at least one of the trigger words of the FrameNet frame listed as their English denotation in Cyc. It also suggests all the common ancestors of these initially collected Cyc concepts in the generalization taxonomy: for example, the frame *Inchoative_change_of_temperature* is associated, among others, with trigger words "chill", "cool" and "heat". In Cyc, "cool" is not associated with any concept (English annotations are lacking), "chill" is associated with `#$Chilling` and "heat" is associated with `#$HeatingProcess`. One of their common ancestors is `#$TemperatureChangingProcess`, which is the right mapping for the frame in question.

The number of suggestions is typically low, so the ranking in which they are presented to the annotator was not essential.

Automatic mapping of roles. Even with the computer-assisted approach described above, the time investment for mapping roles would be on the order of person–weeks. We therefore perform the mapping automatically, based on heuristics only. To increase accuracy, we only map the core roles[7] of each frame. This corresponds to roughly 80% of roles appearing in natural text, as indicated by the FrameNet annotated corpus. In Cyc, we have to consider all roles as mapping targets; we only discard those for which a more specific role (according to role hierarchy) is available as well.

To determine role similarity, we use the glosses and subject/object information. From glosses, a bag of words vector is constructed (with tf-idf weighting, Porter stemming and a stopword list). By subject/object information, we mean that the

[7] *Core role* is a FrameNet concept. Core roles are those that have to either appear in the text explicitly or be implicitly understood from the context. A frame typically has two to four core roles.

two most important roles tend strongly to be the subject and the object, which we try to exploit: For all Cyc roles, it is possible to infer (using role hierarchy) what the subject and the object are, if any. For FrameNet roles, a similar inference is sometimes possible (the hierarchy is much less principled and populated); when hierarchical info is unavailable, we heuristically assume that the first role listed for a frame is the subject with probability 0.7 and object with probability 0.3, and the other way around for the second role listed. For roles that have been identified as subjects or objects, a corresponding tag is added as an extra component to the sparse bag of words vector.

We define role similarity as the cosine between the two length-normalized vectors. To obtain the best global assignment, we create a bipartite graph of roles and weigh every edge connecting two roles r and r' with

$$w(r, r') := \cos(r, r')$$

where $\cos(\cdot, \cdot)$ is the cosine similarity between the feature vectors. We then find the maximum-weight assignment in bipartite graph. The square root was introduced to decrease the "greediness" of the method (propensity to choose the highest-scoring pair regardless of others). Another similarly-performing regularization is logarithmic (treating similarity scores as probabilities; the probability of the global assignment is then the product of pairwise probabilities, i.e. the sum of logarithms). We have also experimented with a few naïve greedy approaches, but found their performance to be worse.

In the above approach, we assume that no two roles from FrameNet map onto a single role in Cyc. This can be problematic. Especially for actions with "symmetric" roles, FrameNet assumes a somewhat confusing notation: for example, the frame *Meeting* contains roles *Party_1*, *Party_2* and *Parties*. Some frame occurrences fill the first two roles and others fill only the third role – depending on the phrasing. In Cyc, all of these correspond to a single role (which may then have two or more distinct fillers).

3.3.2.2 Mapping Role-Fillers (WSD)

For mapping role fillers to Cyc, we use Cyc's built-in `#$termStrings` predicate which connects concepts and English words. Often, a single English word maps onto multiple Cyc concepts. Unlike WordNet, Cyc unfortunately has no "most common sense" information associated with each word. It does, however, have links from its concepts to WordNet. Although created semi-automatically and not of perfect quality or coverage, they allow us to rank all the Cyc concepts suggested by `#$termStrings` using commonness information from their WordNet counterparts. The highest ranking concept is then selected. If there are multiple highest-ranking concepts or if there is no WordNet information available due to absence of links, we give priority to the concepts first returned by the Cyc inference engine.

Task	SDP	MSRL	MSRL baseline
a) Structure semantization ($a_1 \cdot a_2$)	.62 | .59 | .61	.43 | .47 | .45	.67 | .52 | .59
a_1) Frame extraction	.80 | .76 | .78	.56 | .61 | .58	.67 | .52 | .59
a_2) Frame alignment to target KB	.78[8]	.77 (.42)	N/A
b) Constituent sem. (WSD)	.78	.48	.40
Full text semantization (a·b)	.49 | .46 | .47	.16 | .17 | .17	.27 | .21 | .24[9]

Table 3.1: Performance of text semantization methods. Table cells with three numbers give the micro-averaged precision, recall and F_1 at the level of individual roles; cells with single values give the accuracy. (*Micro-averaged* means that each role in each frame contributed equally to the average, regardless of the number of roles in the corresponding frame.) Note that the results are not directly comparable as each algorithm maps to a different knowledge base and under different assumptions. Rather, the numbers give a good idea of how well suited those different assumptions and frameworks are to text semantization.

3.4 Evaluation of Discourse Semantization Methods

We evaluated the performance of the SDP and MSRL method and put them side by side with existing methods that correspond to individual stages of MSRL. The goal is to find which of the two methods is likely more appropriate for deriving a usable semantic representation of text. Remember MSRL is more ambitious in that it tries to recognize more varied syntactic constructs in the input text and in that it maps to a more complex knowledge base. We therefore expect it to perform worse in terms of absolute performance measures, but we hope that the poorer performance measured against higher standards will produce a useful semantic representation, especially when using it in combination with the additional, richer background information available in Cyc.

The results are given in Table 3.1. In summary, SDP is simpler and gives better or comparable results in the fully automatic setting, and is thus used in further experiments in this thesis. For both methods, we evaluated separately the performance for *structure semantization* (identifying the frame and the role fillers) and *constituent semantization* (mapping role fillers to the ontology; essentially word sense disambiguation). Structure semantization is further broken down into two operations: identifying the frame (a verb synset for SDP, a FrameNet frame for MSRL), and expressing them in the target ontology (WordNet for SDP, Cyc for MSRL).

The table also includes the results of two baseline state of the art methods. Because our pipeline is relatively unique, we report separately the performance for each

[8]This is the same as WSD because the frame is identified by the verb synset.

[9]A product of the two lines above for illustrational purposes only. We did not evaluate the WSD stage on the actual SRL output.

of the two stages (structure semantization, constituent semantization) separately. For the first, we refer to the popular SRL tool, Shalmaneser [94]. Shalmaneser maps to FrameNet and stops at that, so we do not report a score for any additional mapping to the target ontology. For WSD to Cyc (as required of MSRL), we report the results by Curtis et al. from Cycorp [99] as the baseline.

SDP. SDP was evaluated against a representative sample of data on which we later (Sections 4 and 5) use the method as the central text preprocessing step. We manually created a golden set of 339 roles. They stem from 50 sentences, each picked at random from a different (also random) online news article. The sentences contain a total 129 frames with 339 nonempty roles including verbs. We achieve an F_1 score of 61% at extracting frames and roles (micro-averaged; i.e. each role in each frame contributed equally to the average).

On the same set of 339 role fillers, we also measured the performance of the "most common sense" WSD heuristic; the accuracy was 78%. This is consistent with the 70–75% result reported in the literature [100, 101] for all-words WSD with the same heuristic (we only disambiguate noun and verb phrase headwords, which is likely somewhat easier).

MSRL. For MSRL, the development of a high-quality golden standard is much harder, so we evaluate against existing FrameNet training data. On a held-out set of 300 sentences, we achieve F_1 of 59%. For the frame alignment stage, the method described in section 3.3.2.1 achieves a disappointingly low accuracy of 42%. Table 3.1 therefore also reports the "maximum" attainable accuracy of 77% that we estimated by mapping 25 randomly selected frames with 83 roles from FrameNet to Cyc completely by hand. It would be possible to map the whole FrameNet to Cyc, a one-time effort, so the 77% are not unrealistic; however, we cannot do better as Cyc lacks relations that would correspond to the remaining 23% of FrameNet roles. We do have to note that mapping accuracy on the subject– and object–like roles is higher, and because real-world sentences use these two roles more than others, the error rate introduced will be somewhat better than what the 42% above suggest.

To estimate the performance of word sense disambiguation in MSRL, we manually inspected a sample of 50 role fillers. The accuracy is 48%, higher than the 40% reported by in related work [99]. As before, the reason for our "improved" performance is very likely our easier task: we only map headwords of role fillers whereas related work evaluates the mapping on all words in a sentence.

A note on WSD evaluation. In computing all the WSD statistics reported here, we ignore the pronouns "he", "she", "her", "him", "his" etc. which are mapped to the generic `#$Person` (Cyc) or `person.n.01` (WordNet) concept with hand-written rules. We also ignore named entities as we can not realistically expect of WordNet or Cyc to know about most of them.

When pipelines get too complex. We can see from Table 3.1 that the MSRL pipeline performs very poorly in terms of both precision and recall. While we partially might chalk up the lower recall to the higher expressivity of Cyc against which MSRL was measured, the combination with low precision is what makes it clear that the initial assumptions were too ambitious. It is of course possible that an approach different to ours would do better at the task, but it seems unlikely that the improvement would be enormous, for two reasons. First, the pipeline is relatively long and complex, combining (sequentially!) two tasks that are in themselves hard: semantic role labeling and word sense disambiguation. Comparison of results with dedicated algorithms suggest there are no obvious huge improvements to be made at either stage of MSRL. Second, there is an inherent mismatch between the Cyc and FrameNet ontologies we cannot do much about, other than changing one or both of the knowledge bases — but in the open domain area, our choice of rich ontologies is fairly limited.

At the same time, we would need precision (and possibly recall) *much* higher than the current 17% in order to meaningfully take advantage of the background knowledge and inference mechanisms available in Cyc. Chapters 4 and 5 therefore both describe methods based on the simpler, SDP-derived frames.

We also cannot skip FrameNet altogether and extract directly to Cyc because Cyc's lexical coverage is still very incomplete. This was also noted by Manning and Ng while attempting to use Cyc in a textual entailment challenge [102]:

> *To be sure, lexical coverage is the deficiency in ResearchCyc which hurts us the most on this task, and it is especially problematic in the absence of functional ResearchCyc NL tools. In most cases we find sparse or suboptimal lexicalizations that render any further search useless. Even on our toy example, the absence of a proper translation for "sells X to Y" keeps us from making the meaningful connection that we would expect from ResearchCyc: that both verbs* ["buy" and "sell"] *express a buying action and can be translated as such given their NP-PP arguments.*

In fact, even with FrameNet, we still see the shortage of training data as a major impediment; most papers and challenges on SRL limit themselves to only the few most-annotated frames as performance drops significantly when averaged over all frames.

Both SDP and MSRL make some unavoidable errors because of their reliance on automatic sentence parsers. In terms of domain independence, full-parse features are problematic because parsers are typically trained on the Penn Treebank (i.e. annotated Wall Street Journal articles) and do not generalize well to other domains, with a domain in change easily causing a 10% drop in performance [97]. SRL, in turn, shows high dependence on parser accuracy, while SDP's high reliance on parsers is even more obvious.

Sample output. As an illustrative example, we are including an excerpt from a newspaper article along with the automatically extracted frames by each of the

methods. For MSRL, we use the lisp-like Cyc notation as this is the target ontology. For SDP, all frames in this example only have the *subject* and *object* roles, so we display each frame as a tiny graph of the form $\boxed{S} \xleftarrow{\text{subject}} \boxed{V} \xrightarrow{\text{object}} \boxed{O}$.

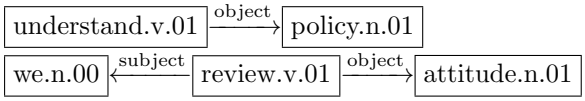
The text: "**(1)** To understand and appreciate the Bush administration's policy regarding Israeli Prime Minister Sharon's disengagement plan, we must briefly reexamine the record. **(2)** For three and a half years now, the administration's attitude toward the Israeli-Palestinian conflict/peace process has been characterized by high rhetoric but little action. **(3)** On the one hand, President Bush is the first US leader to officially endorse the creation of a Palestinian state."

Sentence 1 output:

MSRL:

```
(#$objectImproved #$Comprehending* #$OrganizationPolicy*)
(#$performedBy #$Comprehending* (#$ObjectDenotedByFn "we")*)
(#$evaluationInput #$Evaluating* #$OrganizationPolicy*)
(#$performedBy #$ExercisingAuthoritativeControlOverSomething*
      (#$ObjectDenotedByFn "we")*)
(#$performedBy #$PurposefulAction* (#$ObjectDenotedByFn "Sharon")*)
```

SDP:

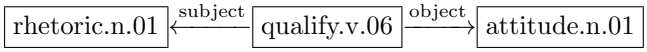


Sentence 2 output:

MSRL:

```
(#$eventOccursAt #$DdescribingSomething* #$Attitude*)
(#$senderOfInfo #$DescribingSomething* #$Action*)
```

SDP:

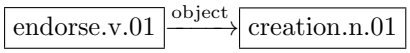


Sentence 3 output:

MSRL:

```
(#$performedBy #$Siding-SelectingSomething* #$Bush*)
(#$doneBy #$ArrivingAtAPlace* #$Bush*)
(#$communicatorOfInfo #$Communicating* #$Bush*)
```

SDP:



For brevity, we denote an instance of a Cyc collection with an asterisk (*). For example, `#$Evaluating` is defined in Cyc as the collection of all evaluating events, so the correct way to denote a single evaluating event (above: `#$Evaluating`*) would be with a variable (e.g. `?E`) and a separate statement (`#$isa ?E #$Evaluating`). The

(#$ObjectDenotedByFn "foo") notation represents a concept Cyc does not know about, but is expressed in English as "foo". Similarly, WordNet synsets ending with .00 represent a concept that does not originally exist in WordNet.

We can see that MSRL extracts a higher number of relations (a relation is represented by one line of Cyc code for MSRL and one arrow for SDP). However, the accuracy of the extracted relations leaves a lot to be desired. Partially, the imperfect match between the FrameNet and Cyc ontologies is to blame. For example, in Sentence 1, we can see that "Sharon's disengagement plan" has been reduced to an uninformative #$PurposefulAction. Other times, SRL errors are to blame; for example, in Sentence 1, "we must" rather needlessly evokes #$ExercisingAuthoritativeControlOverSomething. An example of a WSD error can be seen in Sentence 3, with president Bush being mapped to #$Bush, the garden bush concept.

SDP is more reliable in extracting the correct information, but suffers from its bias towards grammatical subjects and objects. In Sentence 3, president Bush and Palestine are the key entities, crucial for understanding the meaning of the sentence; however, SDP misses both.

Evaluation limitations. The evaluation was performed on newswire text as this is also the domain on which we apply the semantic representation of text in later chapters. The results may be somewhat different on other domains due to several newswire-specific characteristics: vocabulary bias, grammatical well-formedness, and sentences that are longer than usual[10]. However, major differences are unlikely.

A limitation of both methods is that they do not attempt to extract frames that cross sentence boundaries. The evaluation does not consider such frames either, when measuring recall. This is standard practice in related work as well as it significantly reduces complexity while only discarding a reasonably small percentage of frames.

3.5 Semantic Metadata

Besides encoding the (simplified) statements appearing in a the text itself, we can also extract **metadata**, i.e. annotations *about* the text, and present it in a semantic form.

First, there is **emergent information**, extracted by observing larger units of text at a time. Examples include determining the sentiment (positive or negative)

[10] A somewhat extreme example of a single sentence: "To ordinary Malaysians, the more pertinent question about Najib's "Endless Possibilities" campaign is not whether it is a copy of Israeli and/or Mongolian campaign ideas, but whether it would be a clarion and inspirational call to all Malaysians to scale new heights of national endeavor in nation-building and all fields of human accomplishments or it would symbolize the country plumbing new depths of all that is bad, dark, evil, new injustices, exploitation and oppression - the very opposite of the Malaysian Dream for justice, freedom, accountability, transparency, good governance, national unity and harmony for all Malaysians."

of a sentence, of the whole document, or towards an entity; classifying the prevailing topic of the document into predetermined categories; extracting keywords in an unsupervised way; identifying the language of the document; or deriving other, domain-specific scores like spamminess or writing style complexity. These annotations are typically semantic because they are arrived at using purpose-made methods, making their meaning well-defined.

Another source of information about text are the **creation-time annotations** sometimes already distributed along with the text, for example the author, the time of creation, or author-provided keywords. Such metadata is particularly often present in newswire data. The structure is almost always well defined. What may benefit from further semantization are the values provided within that structure. Doing so ranges from easy, for example parsing a date in an unknown format, to slightly more demanding, like resolving a news article's city of origin against a geographical database, to potentially very hard, like disambiguating research paper authors against a worldwide database of researchers.

These types of semantic data are less novel and not the primary focus of this thesis. We touch on them in Chapter 5 where we discuss the integration and aggregation of various types of semantic data to provide better insight into large data collections. The methods for deriving these types of data are referenced in Section 5.2.

Chapter 4

Deriving Domain Templates

In this chapter, we apply the semantic text representation derived in Chapter 3 to the problem of **domain template construction**.

Similar to how we structure *sentences* of a *document* in Chapter 3, here we move to the next coarser level of granularity and consider structuring *documents* of a *document collection*. For the problem of structuring sentences, the structure itself (frames) was manually defined in advance, either by FrameNet or by limiting ourselves to a small, frame-independent set of roles. At the granularity of documents, however, we cannot assume that the frame structure is known; to the best of our knowledge, no appropriate database or schema exists for structuring general texts. Our goal is to construct such document-level frames (here, called "templates") given a collection of topic-related documents.

The output gives us an insight into the recurring types of information common to a large proportion of documents in a collection.

Formal problem statement. We are given a set of documents from a single, relatively restricted domain, for example "news reports of bombing attacks", "weather reports" or "biographies of renowned physicists." The task is to identify, in an unsupervised manner, the most salient properties that can be extracted for most of the given documents; for example, given the "bombing attacks" domain, we wish to detect "attacker", "the destroyed buildings", "victims" etc. as properties that are pervasively present in those articles. We define salient properties as those that would allow a human, if they were given *only* those properties for an *unseen* document, to produce as good an abstract of the unseen document as possible. The properties will be described by their prevailing context and will be assigned a type. For example, the "attacker" property from the previous sentence might be output as $\boxed{\textit{person}} \xrightarrow{\text{detonate}} \boxed{\text{bomb}}$. Here, *person* is the type while $\xrightarrow{\text{detonate}}$ and $\boxed{\text{bomb}}$ provide sufficient context to determine this $\boxed{\textit{person}}$ is the attacker. Automatically assigning the label "attacker" to this property is beyond the scope of this (and existing related) work. We call the collection of these properties for a specific topic a **domain template** or **topic template**.

What constitutes a good domain template? We characterize them as follows:

- A template should be **predictive of expected document content** within a domain. In other words, it should reflect the types of information humans expect to see in documents on that topic. We measure this by comparing the generated templates with human-generated, "golden" ones.
- A template should be **representative** of the domain, i.e. largely independent of the specific training data and not overfitted to single aspects of it. We measure the generalizability of generated patterns by looking at how well a held-out set of on-topic documents fits onto the template that was automatically generated from the remainder of the documents.

The evaluation process and metrics are described in more detail in Section 4.4.

Motivation. A possible application of topic templates stems from the way we defined them – they guide and constrain **Information Extraction** (IE) methods which have a wide variety of applications. Present-day IE algorithms are most often supervised in nature and depend on manual creation of topic templates *and* training documents with labeled slot fillers. Automatic creation of topic templates thus lowers the entry barrier to using IE. Not only does it provide the templates, a high number of labeled slot fillers is almost always a byproduct of automatic template creation.

Another added value of templates is that they expose the **key properties** of a text type. This makes them potentially suitable for guiding summarization or other text shortening tasks by identifying text fragments that should be scored higher.

In combination with information extraction methods, topic templates allow us to create writing "mentors", automated ways of suggesting **missing content** to be included into a document with a known topic. For example, if the user is posting a sales ad for a car (TV, house, ...) – something most people don't do often – the system could remind her of information that is typically included in such ads but the user's ad lacks. Similarly, a journalist covering a story could be reminded of types of information typically covered in related articles but not in hers. On a larger scale, we can imagine a system that analyzes all Wikipedia articles from a given category, derives the template and identifies pages that are missing some of the "standard" properties (e.g. "of all *German Physicist* pages, only Max Planck's lacks info about his schooling").

Another potential use-case scenario involving topic templates is **semi-automatic ontology extension** reminiscent of open IE. Existing relation extraction methods are sometimes used to extend the lowest, fact-based levels of ontologies (e.g. adding `bornIn` relations between persons and places). Templates, on the other hand, provide input for extending the middle level of ontologies: when introducing a new abstract concept C (e.g. "football player") to the ontology, a topic template derived from documents on C can suggest properties and relations (e.g. "played for", "goals scored") to be associated with new instances of C in the ontology.

Input representation. We present two methods for unsupervised construction of domain templates based on semantic representation of input documents. In both methods, we start with the output of the SDP method (Section 3.2) represented as a bag of relational triplets. The transformation of verb frames into triplets can be performed in two ways. For example, the frame

see.v.01	
subject	"Sally.n.00"
object	"man.n.01"
location	"downtown.n.01"

can be equivalently given as the set of triplets $\boxed{\text{see}} \xrightarrow{\text{subj}} \boxed{\text{Sally}}$, $\boxed{\text{see}} \xrightarrow{\text{obj}} \boxed{\text{man}}$, and $\boxed{\text{see}} \xrightarrow{\text{loc}} \boxed{\text{downtown}}$. (Note we also dropped the WordNet suffixes like *.n.01* for legibility.) This is the representation we use in the method of Section 4.3. Alternatively, we may compress the representation even more and discard everything but the absolutely essential *subject* and *object* roles. In that case, we can compress the whole frame into a single relational triplet; for example, the above frame would be given as $\boxed{\text{Sally}} \xrightarrow{\text{see}} \boxed{\text{man}}$. We use this representation in the method of Section 4.2.

The **assumptions** that we make about the input data are as follows:

- A collection of plain-text documents from the domain of interest is available.
- The key information in input documents (and the desired output) can be represented with relational triplets (here, $\boxed{\text{subject}} \xrightarrow{\text{verb}} \boxed{\text{object}}$ or $\boxed{\text{verb}} \xrightarrow{\text{dependency}} \boxed{\text{property}}$). This assumption is likely to be partially violated, which can be alleviated with input data redundancy.

Notation. Let us note again the notation introduced in Section 2.1. We use the following typefaces:

- $\boxed{\text{Node}}$ for concepts extracted directly from documents, e.g. "Obama" and
- $\boxed{\textit{NodeType}}$ for generic, automatically inferred concepts, e.g. "person".

4.1 Overview

Both methods for extracting domain templates presented in this chapter share the preprocessing stage in which triplets are extracted from plain text, as explained above.

In the second, main part of the algorithm, the methods take markedly different approaches. The **Frequent Generalized Subgraph (FGS)** method, presented in Section 4.2, attempts to discover regularities in the *semantic structure* of the documents, i.e. the entities appearing as well as the relations interconnecting them. For example, in documents reporting on murders, we hope to find a complex structure

like

officer —apprehend→ *person* —kill→ *person*
person —receive→ sentence.

The method assumes such complex semantic structures are extremely unlikely to appear outside the context for which they are characteristic (i.e. *murder* stories) and searches for such structures in a manner reminiscent of frequent itemset mining.

The **Characteristic Triplet (CT)** method in Section 4.3 relaxes the assumption on how common these large semantic structures are and instead looks for individual *topic-characteristic triplets* (e.g apprehend —subj→ *officer* and apprehend —obj→ *person* separately), which can be seen as a reduction in the size of sought-after semantic structures. As these small structures appear more commonly even outside the target domain (i.e. in non-*murder* documents), a weakly supervised approach is taken: the algorithm considers both in-domain and out-of-domain documents to learn what triplets are characteristic of the domain.

4.2 Frequent Generalized Subgraph Method

The key idea of the Frequent Generalized Subgraph (FGS) method is as follows: first, construct a **semantic graph** of the document consisting of triplet-derived entities and relations. Then, mine graphs from all topical documents for **frequent subgraphs** whose specializations[1] appear in sufficiently many of those graphs. These generalized frequent subgraphs are what the method suggests as the topic template. The generalized nodes (e.g. *person*) and edges are the template slots and the graph as a whole provides context that makes it possible for humans to interpret the node.

In other words, the method assumes that while an individual triplet (e.g. $person_1$ —kill→ $person_2$) may be frequent across multiple topics and its frequency does not attest to its suitability for a slot pattern, a small subgraph consisting of that triplet and some additional triplets, e.g. $person_2$ ←kill— $person_1$ —detonate→ *explosive*, will only be frequent within a certain topic (here, suicide bomber attacks).

Figure 4.1 illustrates this with sample graphs from the "bombing attacks" domain. The graphs G_1, G_2 and G_3 each represent a semantic graph constructed from an input document. H is the generalized subgraph of all G_i and embodies a (partial) template for the domain. In practice, the graphs G_i are larger, there is more of them and the subgraph H is only required to appear in some of the G_i.

In subsections, we first briefly describe how the semantic graph is constructed, then turn to the technique for mining frequent subgraphs and to its generalization required by our approach.

[1] "Specialization" in the sense of the hypernym taxonomy implied by our background knowledge base. For example, Rodney ←kill— Wiley E. —detonate→ hand grenade is a specialization of *person* ←kill— *person* —detonate→ *explosive*.

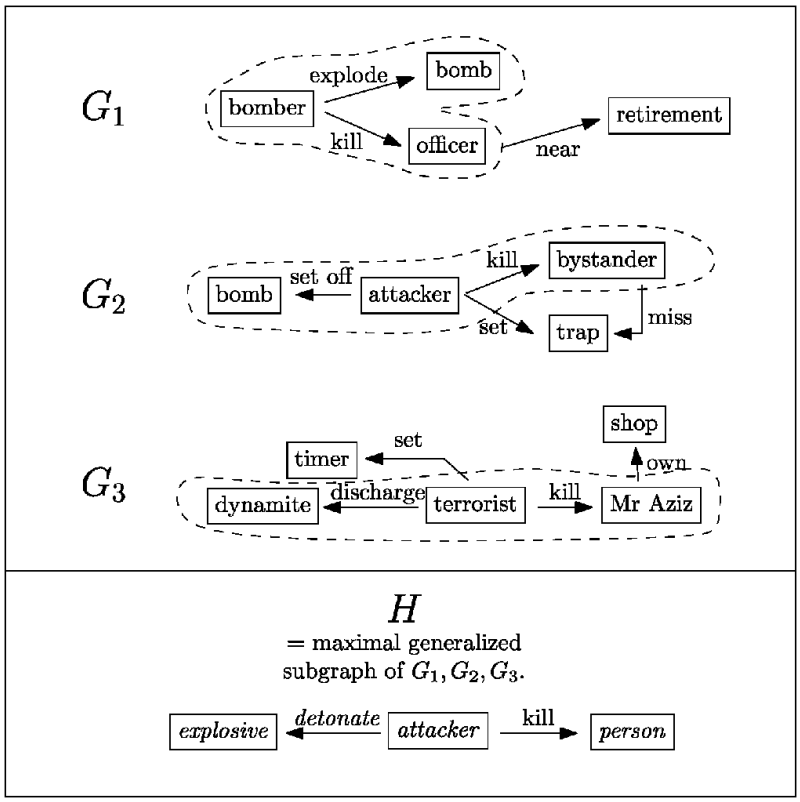


Figure 4.1: Example of a frequent generalized subgraph H as it would be identified by the FGS method for input graphs $G_{1...3}$. Each node in H has a specialization in $G_{1...3}$; e.g., "*attacker*" maps to "bomber" in G_1, "attacker" in G_2 and "terrorist" in G_3. H is the maximal (i.e. "largest") generalized subgraph. Its subgraphs (e.g. *attacker* $\xrightarrow{\text{kill}}$ *person*) are also frequent generalized subgraphs, but not maximal and thus not of particular interest.
This illustrative example is manually constructed for the "bombing attack" domain.

4.2.1 Semantic Graph Construction

We construct the semantic graph from subject $\xrightarrow{\text{verb}}$ object triplets derived as described in the introduction of this Chapter. We consider each triplet to be a **2-node graph**, then treat the collection of all the triplets as a large disconnected graph and finally **merge** (collapse, identify) the nodes with the same labels.

The key simplifying assumption is that input documents tend to be focused in scope: if two entities within a single document share a label, we assume they are the same entity. This is largely true of news articles, our primary corpus of interest. The assumption can sometimes be too strong and introduce some error. As an example, if an article mentions two buildings, one of which burns down and the second of which acted as a shelter for the fire fugitives, our method detects a single "building" and assigns both properties to it. Although having a means

of distinguishing between the two would clearly be preferable, we have found this simplification not to cause significant issues in the newswire domain: entities which do need to be disambiguated are almost always presented with more unique names ("France" instead of "country" etc.). This rationale would have to be revised if one wanted to apply the approach to texts that are broader in focus.

Combating data sparsity. This method mines subgraphs that are frequent across individual article graphs. However, because of the relatively poor recall exhibited by the text semantization method, article graphs tend to be small, each capturing only a part of the information conveyed in the article. Experiments show that article graph almost never share substructures beyond a node or two in size.

We work around this issue by evaluating this method on news stories: each graph is derived not from a single article but from the (textual) concatenation of 20–50 news articles from different sources that are all reporting on the same story. This provides enough redundancy so we can observe subgraph patterns occurring across different *story* graphs. The requirement to have such redundant input data is a current limitation of the method; it could be avoided with significantly higher-recall extraction of triplets from natural text.

4.2.2 Frequent Generalized Subgraph Mining

As described in Section 4.2, the method requires us to find frequent subgraph(s) of input graphs in a generalized manner, taking the hypernym taxonomy into account. This subtask is non-trivial.

Formal problem statement. (Figure 4.1 uses the same notation and can aid in understanding the statement.) Given a set of labeled graphs $S = \{G_1, \ldots, G_n\}$, a transitive antisymmetric relation on graph labels $genl(\cdot, \cdot)$ (with $genl(l', l)$ interpreted as "label l' is a generalization of label l") and a threshold $\theta \in \mathbb{N}$, we wish to construct all graphs H that are *generalized subgraphs* of at least θ graphs from S. A graph H is said to be a generalized subgraph of G iff there is a mapping f of vertices $V(H)$ onto a subset of $V(G)$ such that $genl(v, f(v))$ holds for all $v \in V(H)$, and analogously for edges.

We are only interested in those H that are maximal in size, i.e. there is no graph $H^* \supsetneq H$ such that H generalizes H^* and H^* also satisfies the above criteria. Among those, we only seek H that are as specific as possible.

This is **computationally** an exceptionally **hard problem**. Even finding frequent subgraphs verbatim – without taking possible generalizations (hypernyms) into account – presents a search space of subgraphs that grows exponentially with their size, and isomorphisms make even naive counting non-trivial. Extending the problem with generalizations makes the search space even larger: each node in graphs $\{G_1, \ldots, G_n\}$ can be independently generalized in multiple ways[2], making for yet another exponential growth factor.

[2]For example, possible generalizations of suicide_bomber are *terrorist*, *radical*, *person*

We alleviate the generalization problem as follows: first, transform all input graphs by **completely generalizing** each input node. Then, perform regular frequent subgraph mining on these graphs to obtain candidates for subgraphs H as they are defined in the formal problem statement. The subgraphs obtained this way are typically overly generalized, so we **specialize them back** as much as possible without the support falling below θ.

Regular frequent subgraph mining in itself can be problematic – we had three modern dedicated programs (gSpan [103], Gaston [104] and HybridTreeMiner [105]) crash on our graphs with tens of thousands of nodes and thousands of labels (but work on smaller graphs), so we implemented our own solution based on their ideas. The approach works in a way reminiscent of the classic a priori algorithm in frequent itemset mining: start with the smallest possible frequent graphs, i.e. those on one node, then iteratively add more and more nodes to them, discarding all graphs with an overly low support at each iteration.

4.3 Characteristic Triplet Method

The Characteristic Triplet method is the second approach to constructing topic templates we propose. Its key idea is to find triplets which are **frequent** in documents belonging to the topic, yet **infrequent** in documents not belonging to it. Frequency is again considered in a **generalized** sense: [Obama] contributes to the counts of [*politician*], [*person*] and [*entity*]. As with the FGS method, we are not searching for triplets that appear in the input documents verbatim but rather for their generalizations. For example, for the topic "political visits", we are looking for [*politician*] $\xrightarrow{\text{visit}}$ [*country*], not [Obama] $\xrightarrow{\text{visit}}$ [Germany].

The algorithm is based on the expectation that for any given topic, triplets (both the verbatim and generalized ones) will fit into one of the three categories below. Illustrative examples are given for the "diplomatic visits" domain:

- The **overly specific** triplets (e.g. [...] $\xrightarrow{\cdots}$ [Obama]) and the irrelevant ones (e.g. [...] $\xrightarrow{\cdots}$ [*football player*]) will have a low frequency count.
- The **overly generalized** triplets (e.g. [...] $\xrightarrow{\cdots}$ [*entity*]) will be frequent in on-topic documents but also off-topic ones.
- The triplets that are generalized "**just right**" (e.g. [...] $\xrightarrow{\cdots}$ [*politician*]) will be frequent in on-topic documents but less frequent otherwise; these are the ones we aim to detect.

The remainder of this section describes the algorithm based on this idea. We collect all triplets from input documents and all their generalizations and assign

and [*entity*]

them scores that reflect the above intuition. The highest-scoring triplets form the topic template.

4.3.1 Triplet Lattice

The method assumes, in addition to the on-topic documents, a number of off-topic plain-text documents representative of the **background language**.

We start by representing each document as a set of *verb–dependency–property* triplets as described in the introductory part of this Chapter. In comparison to the Frequent Generalized Subgraph (FGS) method from the previous section, this representation makes the method less susceptible to data sparsity in triplet space but discards some more of the original structure.

We next **construct a lattice** of triplets encountered in the input documents and their generalizations. Let us denote with c' the direct generalization (hypernym) of a concept c. We initialize the lattice with every triplet $\boxed{v} \xrightarrow{d} \boxed{p}$ appearing verbatim in the input documents. Note that the points of the lattice are triplets which themselves are considered atomic. We then recursively extend the lattice by assigning to each triplet $\boxed{v} \xrightarrow{d} \boxed{p}$ as its parents the triplets $\boxed{v'} \xrightarrow{d} \boxed{p}$ and $\boxed{v} \xrightarrow{d} \boxed{p'}$. See **Figure 4.2** for an **illustration**. Because the lattice is constructed using the hypernymy relation, it is a DAG (directed acyclic graph) and implies a partial order relation.

4.3.2 Cutting the Lattice

Each triplet t in the lattice is assigned a *frequency count*, defined as the number of times t or its specializations appear in on-topic documents. Formally, let $t \geq t^*$ denote that t is above t^* in the lattice, and let T_+ and T_0 denote the multiset of triplets in the on-topic documents and in the entire corpus, respectively. In T_+ and T_0, each triplet is counted once per source document. Then we define the frequency count of triplet t in on-topic documents as

$$f_+(t) := |\{t^* : t^* \in T_+, t \geq t^*\}|.$$

Analogously, we define $f_0(t)$ as the frequency count of t in the whole corpus. The value of $f_+(t)$ is also illustrated in Figure 4.2; note how the off-topic documents do not contribute to $f_+(t)$ and how the value is not necessarily the sum of values in t's children, but rather the count of all on-topic descendants (which may be shared among t's children).

Care has to be taken when computing $f_\cdot(t)$ if it is done the natural way, by iterating through input triplets. When encountering a triplet t, it is *not* correct to increase $f_\cdot(t)$ and recursively $f_\cdot(t')$ for all direct lattice ancestors t' of t. This is because there are multiple paths between t and its lattice ancestors two or more levels higher, so they will be counted multiple times.

Additionally, we assign a *score* to each triplet t in the lattice. The score $s(t)$ is tf-idf inspired, taking into account the count of triplet in on-topic documents and

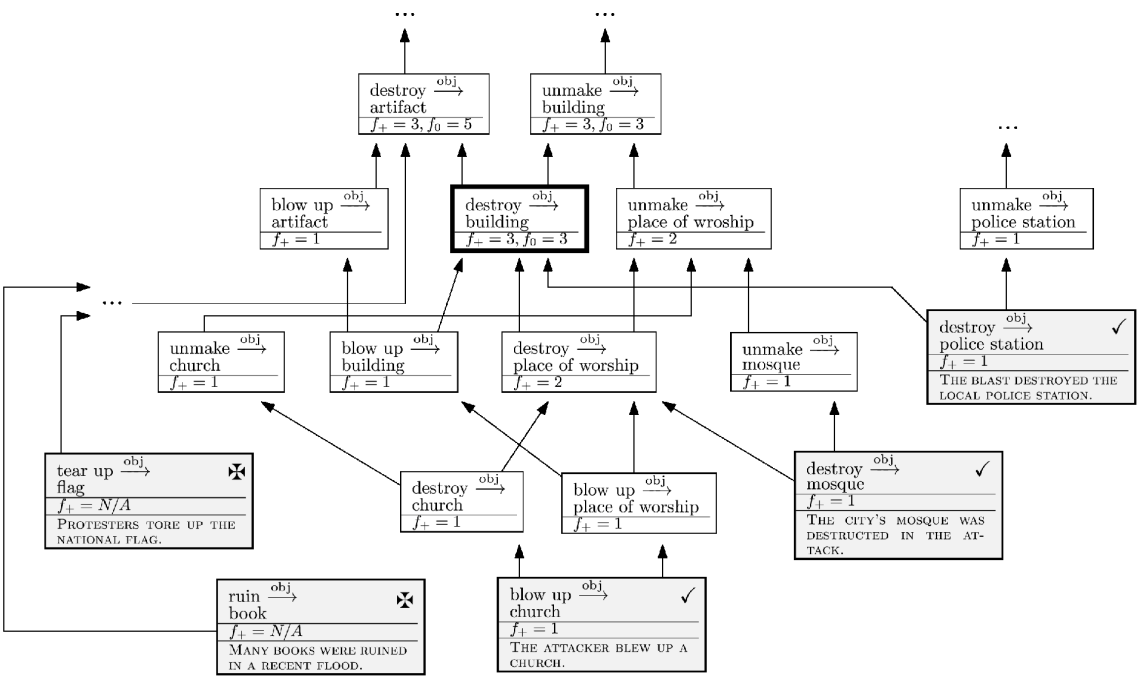


Figure 4.2: An example of a triplet lattice as constructed by the Characteristic Triplet (CT) method. Each box shows a triplet and its frequency f_+ in the on-topic documents. Here, the topic is "bombing attack". Each grey box represents a triplet that appears verbatim in an on-topic (✓) or off-topic (✠) input document. Grey boxes also contain the sentence that gives raise to the triplet. Arrows point from less generalized to more generalized triplets. The thick-bordered box represents the triplet with the highest score that gets selected for the template. The scores are related to the frequency f_+ but not shown here; see Section 4.3 for discussion.

the proportion of the triplet in the entire corpus:

$$s(t) := f_+(t) \cdot \log \frac{|T_0|}{f_0(t)}$$

These scores form the basis for selecting the triplets that will form the topic template. In Figure 4.2, the triplet $\boxed{\text{destroy}} \xrightarrow{\text{obj}} \boxed{\textit{building}}$ and its two parent triplets have the highest $f_+(\cdot)$. However, $\boxed{\text{destroy}} \xrightarrow{\text{obj}} \boxed{\textit{artifact}}$ has a lower score than the other two since it also appears in the two non-topical documents ($f_+ = 3$, $f_0 = 5$).

For $s(t)$, we also briefly experimented with a log-linear, Bayesian-like variant of the formula ($s(t) := f_+(t)/(f_0(t) - f_+(t))$) but discarded it quickly – even after tweaking smoothing factors, the results were so much worse that quantitative evaluation was not necessary

4.3.3 Triplet Respecialization

Before we select the triplets with highest scores, we **correct** for an **undesired artifact** of the scoring function.

First, observe that for any triplet t with $s(t) \neq 0$, every generalization t' of t such that no other specializations t^* of t' has $s(t^*) \neq 0$, we have $s(t') = s(t)$. For example, in Figure 4.2, since there are no off-topic documents that contain (a specialization of) $\boxed{\textit{unmake}} \xrightarrow{\text{obj}} \boxed{\text{building}}$, this triplet has the **same score** as $\boxed{\textit{destroy}} \xrightarrow{\text{obj}} \boxed{\text{building}}$ even though the more specialized version is clearly preferable.

To illustrate another closely related problem using Figure 4.2, assume *unmake* is the direct hypernym of *destroy* and *disassemble* in WordNet. Then, if a single $\boxed{\textit{disassemble}} \xrightarrow{\text{obj}} \boxed{\text{building}}$ were (possibly erroneously) detected in the on-topic documents, $\boxed{\textit{unmake}} \xrightarrow{\text{obj}} \boxed{\text{building}}$ would have a higher score than $\boxed{\textit{destroy}} \xrightarrow{\text{obj}} \boxed{\text{building}}$ even though $\boxed{\textit{unmake}}$ "earned" most of its score through $\boxed{\textit{destroy}}$. More generally, climbing the lattice always causes the score to **monotonically increase** as long as we don't encounter triplets that have specializations occurring in non-topical documents.

We correct for this effect by **discarding** all triplets t which have one or more children t^* such that $s(t^*) > 0.80s(t)$. Here, 0.80 is a parameter that we fixed by manually tuning and observing performance on a held-out set of documents for topics `bomb` and `airplane` (described in Section 4.4.1). It is fairly robust; values in the range from 0.75 to 0.90 all gave comparable results. In Figure 4.2, the triplet $\boxed{\text{unmake}} \xrightarrow{\text{obj}} \boxed{\textit{building}}$ is discarded in favor of $\boxed{\text{destroy}} \xrightarrow{\text{obj}} \boxed{\textit{building}}$ since it has the same score.

Finally, we take the 1000 top-scoring triplets and retain those that represent topic slots, i.e. have more than one specialization in the input documents.

An alternative to TF-IDF based scores We also considered a feature-selection motivated variation of the method described here. When selecting the triplets into the template, we first constructed a Naive Bayes classifier classifying documents as either in-domain or not. We then replaced the TF-IDF scores with weights from the Naive Bayes classifier; i.e., triplets chosen into the template were the most discriminative ones. However, even without a quantitative evaluation, it was clear that the algorithm performs less well using these scores.

4.3.4 Frequent Generalized Subgraph (FGS) vs Characteristic Triplet (CT) Method

Note that like the FGS method described in the previous section, the CT method operates in the space of triplets. However, it makes several notable improvements:

- CT does not treat each topic in isolation but rather in relation to the background corpus distribution.

- By operating on structurally less complex units (triplets instead of subgraphs), CT does not require clusters of tightly related documents as input (see "Combating data sparsity" in Section 4.2.1).
- Due to not having to perform frequent subgraph mining which is superlinear in complexity, the CT method scales considerably better.
- FGS expects a high level of regularity in the data to detect patterns, an expectation that often goes unfulfilled. CT is more flexible (and can therefore detect a higher number of patterns, as the evaluation later on also shows).

4.4 Experimental Setup

Evaluation of domain templates is not straightforward[3], to the point that several related works only evaluate qualitatively (i.e. show a selected part of the output) or evaluate other aspects of their methods. There was so far no direct comparison of methods.

We evaluate on news articles from five domains, comparing three methods: our FGS and CT and a state of the art baseline. Section 4.4.1 describes the data and Section 4.4.2 proposes a detailed methodology for evaluating this research problem.

4.4.1 Datasets

We evaluated the algorithms on five domains/topics, each captured by a set of news articles. The datasets are identified by single-word names:

- `airplane` - Reports of airplane or helicopter crashes.
- `bomb` - Reports of terrorist attacks (often by suicide bombers).
- `sentence` - Reports of sentencings passed in a court of law.
- `earthquake` - Reports of past earthquakes.
- `visit` - Reports of diplomatic visits by politicians.

We chose the topics based on what is best represented in the media and based on the choices made by [37], the work we compare with. They evaluate on four domains: airplane crashes, earthquakes, presidential elections and terrorist attacks. However, for the presidential elections domain they discover it is ill-defined – when trying to define the golden standard collection of domain slots, the inter-annotator agreement was only 0.32.

For each of the topics, we collected a number of news articles from the web using a combination of manually designed keyword queries, then exploiting story-level

[3]A related article [41] notes, "While [template creation] is a difficult problem, its evaluation is arguably more difficult due to the dearth of suitable resources."

clusters provided by Google News[4] to quickly obtain multiple articles reporting on the same story (and therefore the same topic). The sizes of collections are given in Table 4.1. The articles were published mostly in March and April 2009. In addition, we collected a random set of news articles from the same time period by crawling the top articles from Google News; those articles represent the background distribution and are with relatively high probability not topical for any of our topics.

Topic	**# of docs**	**# of stories**
airplane	294	40
bomb	937	12
earthquake	311	5
visit	489	9
sentence	350	8
(nontopical)	3638	100
total	*6019*	*174*

Table 4.1: Size of the corpus

4.4.2 Evaluation Methodology

As stated in the introduction, we aim to extract templates that are a) representative and predictive of new documents' content within a topic and b) not overfitted to training data. The first property, in particular, is hard to evaluate, and there is no established methodology. We are therefore devoting an entire section to proposing one.

To maximize the reproducibility of results, we need to create a golden standard, i.e. the "ideal" template for every domain we wish to evaluate on. There are two problems associated with creating a golden standard:

Golden standards are noisy. Like the better-known problem of summarization, our problem is inherently weakly defined; the notion of the "best" template differs from human to human. In our case, the problem is even more pronounced because laypeople do not easily understand what a template/schema is, so getting a consensus is harder.

Determining similarity to the golden standard. Because of the expressivity of natural language, it is possible to obtain an output that is syntactically largely different from the golden standard, but semantically closely related. This is again a problem faced when evaluating summarization algorithms.

[4] `http://news.google.com`

4.4.2.1 Creating the Golden Standard

We combat the first problem listed above by disguising our task: we ask evaluators to have a look at some domain documents and then **pose 10 questions** that they believe would best help them summarize a **new, unseen document** from the domain if they got answers to them. This idea is largely due to Filatova et al. [37].

We used the TaskRabbit[5] platform to recruit evaluators. The workers were not required to be domain experts, i.e. they had common-sense understanding of the domains only. They were native English speakers and were not in any way affiliated with the research. To provide reproducibility, the exact phrasing (which proved to be very important) of the instructions given to workers is available; see Appendix A. We used three workers for each task.

Finally, we revised and aggregated the questions ourselves. About a quarter of questions was discarded because they did not follow instructions. They tended to fall into two categories: 1) questions obviously referring to a single article instead of the topic in general and 2) meta-questions, e.g. "Who is reporting?", "Where was the article published?" etc. Within the remaining questions, we identified synonymous ones and retained the top 10 questions based on the number of times they were asked by our evaluators. Ties were broken by an unaffiliated friendly colleague in the hallway. These remaining *golden questions* form the golden standard. Table 4.2 lists the most popular questions for the "bombing attack" domain.

Sample golden questions
Who was killed?
Who was injured?
Which organization is suspected / admitted responsibility?
Where did the event happen?
Who was the bomb intended for?

Table 4.2: Sample golden questions for the "bombing attack" domain.

While somewhat cumbersome to evaluate with, the golden standard in the form of natural-language questions has another advantage: it does not impose a representation or format on the algorithm output. This potentially allows a greater number of algorithms to be compared against each other, especially with the domain template construction problem where the community has not yet converged on a single template representation.

4.4.2.2 Comparing Against the Golden Standard

Although the language in which we express our templates (i.e. taxonomy-aligned subject-verb-object triplets) is more constrained than English, it still cannot com-

[5] `http://taskrabbit.com`; it differs from typical crowdsourcing platforms in that the tasks are larger and the involvement with workers more personal.

pletely avoid the phenomenon of having multiple expressions (triplets) representing essentially the same property of the domain template. The golden questions therefore cannot be uniquely mapped to triplets and cannot be compared against the algorithms' output directly.

We therefore evaluate manually, using the CrowdFlower[6] crowdsourcing platform. We present the workers with a form that allows them to mark, for each output triplet, all the golden questions for which the triplet entails the answer. They can also mark that the triplet answers no questions. In CrowdFlower terms, one such triplet-questions pair is called a *unit*.

We go to some length to ensure the output from CrowdFlower is of high quality. First, we use their built-in mechanism of "gold units" (unrelated to our "golden standard"): we provide the expected worker responses to five clear-cut units, and workers that do not get them right are excluded from further evaluation. Each unit is answered by five workers. We then further filter the responses in post-processing: we ignore all responses from users that, for any unit, marked more than two questions or marked a question and simultaneously the "this triplet answers no question" option. Additionally, we filter out workers that have a CrowdFlower-internal trustworthiness score below 0.88.

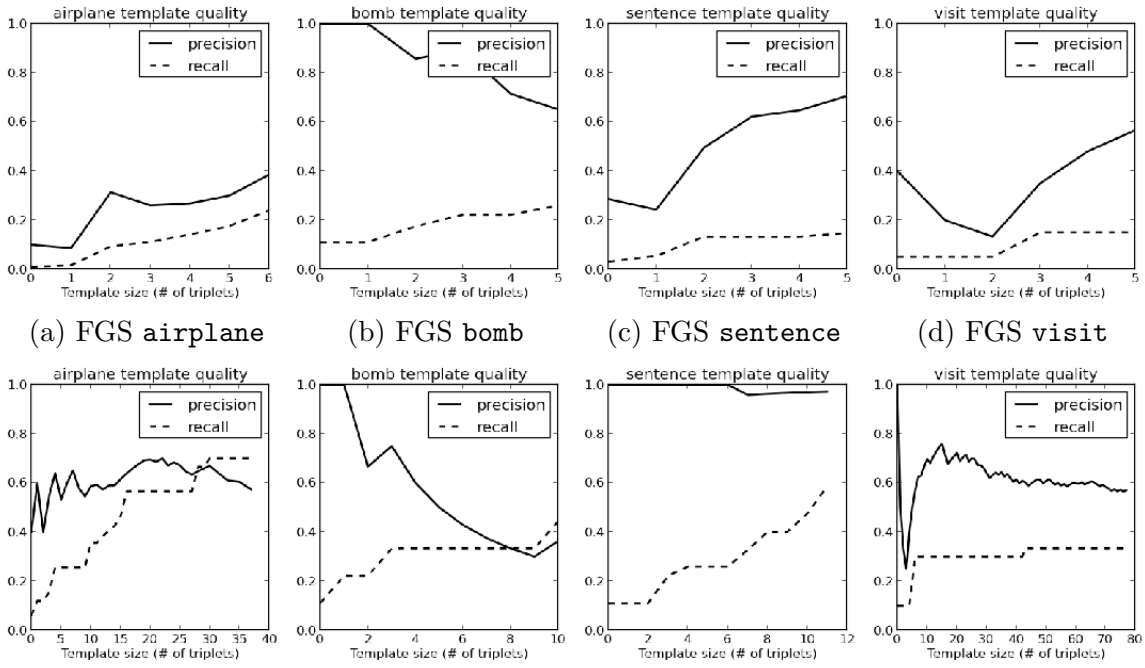
Finally, precision is computed as the percentage of output triplets that answer some golden question. Recall is computed as the percentage of golden questions answered by some output triplet.

4.4.2.3 Gauging Generalizability

As mentioned in the introduction and at the beginning of Section 4.4.2, we also wish to verify that the templates are not overfitted to the training corpus; this is of particular concern with our approach that qualifies template slots with detailed type information. A slot might look reasonable on the outset, e.g. $\boxed{\text{earthquake}} \xrightarrow{\text{hit}} \boxed{\textit{capital}}$ captures the location of an earthquake, but in reality earthquakes do not only hit capital cities and $\boxed{\textit{city}}$ is preferred to $\boxed{\textit{capital}}$.

As this property is not of central importance, we measure it automatically by proxy. For each topic, we take at most 80% of topical documents and use them to construct the topic template. For the remaining held-out set of documents, we verify how many of their triplets can be aligned to (i.e., are specializations of) the template triplets. We are careful to make the training-vs-test cut so that no story is split between the two sets, ensuring that matches observed in the held-out set are due to *topic*-specific, not story-specific pattern triplets. This metric does not generalize to other datasets, but as we only aim to compare our own methods, this simple approach suffices.

[6] `http://crowdflower.com/`; a reseller for Mechanical Turk and other, smaller crowdsourcing platforms



(a) FGS `airplane` (b) FGS `bomb` (c) FGS `sentence` (d) FGS `visit`

(e) CT `airplane` (f) CT `bomb` (g) CT `sentence` (h) CT `visit`

Figure 4.3: Precision and recall of template triplets as measured by the golden standard.

4.5 Results and Discussion

This section gives results of the evaluation described in Section 4.4, with emphasis on template quality.

4.5.1 Template Quality

This subsection describes results pertaining to the evaluation described in Section 4.4.2.2, *Comparing against the golden standard*. We compare ourselves with FVM [37], a state of the art method. While at least two methods [41, 40] were suggested in the literature later than FVM, it is impossible to know which one is the best as no direct comparisons have been made between any template construction methods so far. In addition, FVM is representative of a large group of related methods. Finally, it is the most detailed in its description of evaluation, making it possible to compare against at all. The method is summarized in Section 2.2.2 on related work; in brief, it characterizes domain templates as frequent subtrees of input sentences' parse trees.

Our evaluation is set up so that it only extends the measurements performed in the FVM paper. The metric they report is recall (i.e. percentage of answered golden questions) at 20 "patterns", which are comparable to our triplets (see Table

4.4 for examples of both). When preparing golden questions, the FVM authors do not merge individual worker's questions into a single golden set and instead measure performance against each worker's "golden" questions. The differences in measured performance across workers are however low, in the 5% range, so we use the average for the purpose of our comparison. The results are given in Table 4.3. FVM does not report precision and we agree that recall is the truly relevant metric. The generated templates are primarily intended for humans, and discarding e.g. 3 out of 4 suggested templates (as would happen with a very low precision of 25%) is a task our brain can still do easily and quickly.

Domain	FVM	FGS	CT
airplane	0.53	0.24	**0.57**
bomb	**0.52**	0.26	0.44
earthquake	0.38	0.50	**0.54**
visit	—	0.15	**0.30**
sentence	—	0.15	**0.59**

Table 4.3: Recall@20, i.e. the percentage of golden questions answered by top-20 template triplets. Comparison with state of the art (FVM). Results for FVM are taken from the original paper [37].

It is clear from the table that FGS generates relatively poor templates relative to the other two algorithms. However, CT and FVM are roughly comparable, with our method performing better than FVM in two out of three domains. Both methods are consistently able to cover between one third and one half of golden questions with the automatically generated templates.

FVM authors did not evaluate on the `visit` and `sentence` domains. For the `earthquake` domain, the FGS method failed to discover any frequent subgraphs and thus produce a template. This is due to a somewhat unfortunate choice of input data which only clusters into five stories combined with the fact that FGS operates on stories, not individual documents; discovering "frequent" subgraphs in five input graphs, large as they may be, is extremely noisy.

For our own methods, FGS and CT, we also provide precision and recall curves in Figure 4.3. The figure further confirms that CT is preferred over FGS. The irregular shapes of the precision curves show there is room for improvement in triplet ranking; whenever a high-quality topic triplet is ranked lower than a low-quality one, this causes an increase in the average precision and thus an upwards slope, while the precision curve of an ideally ranked set of template triplets would be monotonically decreasing. This discrepancy is particularly noticeable for the FGS method where a triplet "score" for the purposes of this plot is simply its frequency in input graphs, making for a poor ranking. The jagged lines are also the reason we chose an unorthodox but (in this case) more legible format for the precision-recall graphs. However, the overall precision is good, showing that our templates can facilitate manual domain template construction.

Sample outputs. In Table 4.4, we show a **sample of patterns** produced by the three algorithms for the `bomb` domain. The *italic* text denotes template slots.

Note the highly detailed, automatically extracted slot types[7] in the output of our methods, which exploit background knowledge, compared to the output of FVM which operates on raw text and only abstracts away named entities (presumably with *number*, *date*, *person*, *location* and *organization*). Using a general-purpose taxonomy like WordNet also allows us to identify slot fillers that are not named entities (hotel, mosque, policeman, ...), unlike the great majority of related work.

Limitations of the semantic approach. In Table 4.4, we have intentionally included triplets that illustrate the limitations which any semantics-based (here, WordNet-based) approach likely has to face. First, the parsing of text into concepts and relations during preprocessing introduces errors that propagate through the pipeline. For example, "kill $\xrightarrow{\text{object}}$ *city/metropolis*" from CT output is technically wrong – city is the location of the killing, not its object. Second, the hypernym/hyponym distinctions in WordNet are sometimes very subtle, making variation in content across documents appear larger than it is. This causes, for example, the CT method to detect a slot *attack* with sample slot fillers *bombing*, *attack* and *raid*, between which people likely do not care to distinguish. Third, while it certainly helps that WordNet collapses synonyms, sometimes the choice of the representative lemma for the synonym group (synset) is unusual or misleading. For example, the verb *collar/nail* in one of the CT triplets corresponds to the synset (WordNet concept) that also means "to arrest". Ideally, our algorithm should track the fact that this is the synset lemma that appeared in the text most often and use this lemma for display purposes.

Reducing redundancy in the output set of triplets. Triplets as returned by existing methods are still not purely semantic: a fact can still be expressed with multiple triplets which are, as far as the ontology is concerned, unrelated (ex: $\boxed{\text{be\_after}} \xrightarrow{\text{obj}} \boxed{\textit{person}}$ and $\boxed{\text{target}} \xrightarrow{\text{obj}} \boxed{\textit{personnel}}$). We tried to make the results easier to interpret by clustering the pattern triplets post hoc. Two pattern triplets are considered more similar if their slots are more often filled with the same filler in the same story. Multiple similarity measures deriving from this intuition were tried, but none yielded satisfactory results, most likely due to data sparsity and the underconstrained nature of the problem. For example, $\boxed{\text{enter}} \xrightarrow{\text{obj}} \boxed{\textit{building}}$ and $\boxed{\text{destroy}} \xrightarrow{\text{obj}} \boxed{\textit{building}}$ were clustered by these methods because both triplets appear almost exclusively in articles related to bombing attacks, where they obviously strongly correlate. Given a much higher number of random non-bombing documents, the number of non-correlated occurrences of $\boxed{\text{enter}} \xrightarrow{\text{obj}} \boxed{\textit{building}}$ and

[7] Sometimes, statistics reveal more than we might expect – in determining that the location of a bombing attack is usually of type *Asian_country*, the CT method unknowingly makes a sad but true political commentary.

[destroy] $\xrightarrow{\text{obj}}$ [*building*] would likely increase, possibly making the proposed approach effective. However, Yates et al. [12] report only 35% recall in identifying synonymous relations despite this being the primary goal of their paper; this proves that the problem is hard.

4.5.2 Triplet Generalizability

This subsection gives the results of evaluation from Section 4.4.2.3, comparing the FGS in CT outputs in terms of how well they generalize to unseen data. The previous section shows that CT produces templates that human evaluators score as being more meaningful for their respective topics. However, this does not necessarily reveal much about how well the patterns represent the topics on the syntactic level. Do CT triplets also apply better to a held-out set of documents? Table 4.5 lists the AUC metric for a simple topical classifier: a document's score for domain d is defined as the sum of scores of template triplets that can be found in d. The classifier classifies into the domain with the highest score. Please note that we are not suggesting CT or FGS should be actually used for classification; their performance at this task is measured only to see which of the two produces a template that generalizes better to unseen data.

CT strongly outperforms FGS in this scenario as well. Analysis shows that this is mostly a direct consequence of the low number of patterns that FGS is able to suggest; its template is thus too restricted and relatively unlikely to fit unseen documents.

Although not relevant to relative comparison of CT's and FGS's performance, the variation in performance across domains is notable. The differences are expected; some domains are simply more dissimilar to the other domains and inherently easier to distinguish.

4.5.3 Data Representation Error Analysis

On the same set of 129 verbs and 210 modifiers, we also measured the performance of the "most common sense" word sense disambiguation heuristic. The measured accuracy was 76%. This is consistent with the 70–75% result reported in the literature [100, 101] for all-words word sense disambiguation with the same heuristic. (We only disambiguate noun and verb phrase headwords, which is likely somewhat easier.)

Note that even an incorrectly disambiguated word might still produce desired results. For verbs, the hypernym hierarchy is flat and the final patterns often contain verbs as they appeared in the text, without further generalizing them. When the pattern is finally presented to the user, it is semantically incorrect (as it is linked to the wrong WordNet concept) but looks correct, which might suffice depending on the use case. For nouns, the different senses of a word are sometimes very related and share the same hypernym: for example, `car.n.01` (an automobile) and `car.n.02` (a

railway car) are both specializations of `wheeled_vehicle`. When a triplet involving the word *car* gets generalized during the template creation process, it does not matter any more whether it was initially disambiguated to the correct sense.

There are also cases where disambiguation goes critically wrong. For example, in the `bomb` domain, a relatively high-scoring pattern was $\boxed{\text{vehicle}} \xleftarrow{\text{subj}} \boxed{\text{kill}}$, largely the consequence of the word *bomber* being consistently incorrectly disambiguated as a bombing aircraft.

Frequent Verb Modifier (FVM)
killed (*number*) (NNS people)
(*person*) killed
(NN suicide) killed

Freq. Generalized Subgraph (FGS)
bomber – kill – *person/individual* **Ex:** worshipper, policeman, civilian, person
bomb – kill – *integer/whole_number* **Ex:** 10, one, two
person/individual – claim – duty/responsibility **Ex:** leader, commandant
bomber – strike – *station* **Ex:** police_station, terminal
person/individual – explode/detonate – explosive **Ex:** man, soldier, militant

Characteristic Triplet (CT)
kill $\xrightarrow{\text{object}}$ *defender/guardian* **Ex:** guard, constable, policeman
kill $\xrightarrow{\text{object}}$ *integer/whole_number* **Ex:** 10, twelve, 15
target/aim $\xrightarrow{\text{object}}$ *force/personnel* **Ex:** police, military_personell
damage $\xrightarrow{\text{object}}$ *vehicle* **Ex:** car, truck, airplane
destroy/destruct $\xrightarrow{\text{object}}$ *building/edifice* **Ex:** hotel, building, mosque
kill $\xrightarrow{\text{location}}$ *Asian_country* **Ex:** Afghanistan, Pakistan, Iraq
kill $\xrightarrow{\text{object}}$ *city/metropolis* **Ex:** Beyrut, Kandahar, Bari
collar/nail (= arrest) $\xrightarrow{\text{time}}$ *weekday* **Ex:** Monday, Tuesday
attack/onslaught $\xleftarrow{\text{subject}}$ come/come_up **Ex:** bombing, attack, foray/raid

Table 4.4: Sample output from all three methods for the `bomb` domain. Template slots are shown in *italics*, **Ex** shows automatically extracted example values for the slot. All labels are taken directly from WordNet.

Domain	FGS	CT
airplane	0.69	**0.83**
bomb	0.67	**0.71**
earthquake	0.50	**0.78**
sentence	0.73	**0.91**
visit	0.52	**0.82**

Table 4.5: Domain classification AUC in one-vs-all scenario.

Chapter 5

Exposing Opinion Diversity

In Chapter 4 on domain templates, we saw a way of extracting a structured description of what a set of related documents has *in common*. In this Chapter, we consider the complementary problem: given a set of related documents, can we easily expose the ways in which they *differ* from each other?

Documents can differ in many ways, of course. Here, we consider the differences in opinions held by the authors. Opinions are a subjective category, difficult to interpret automatically and with no clear format in which they should be presented. We therefore do not aim to produce a structured output presenting the differences between documents. Instead, we design a user-facing application that allows for easier discovery of documents with contrasting opinions.

We focus in particular on newswire documents, where multiple reports of a single event typically present multiple slightly different opinions, viewpoints, and even facts. Exposing the differences in those, enabling a well-rounded view of a subject matter, has a clear value in practice. Online news in particular is a very pertinent use case. The internet has been strongly gaining prominence as a news medium; in 2012, it overthrew TV in the US as the most popular source of news for people under 30 [106]. In addition, internet has significantly changed the way in which many people find and consume news. Multiple publishers are now reachable more easily than ever before. Social bookmarking sites present us with news deemed interesting by our peers. News aggregation sites give us an instant overview of the topics of the day.

Although this plethora of sources theoretically provides a richness of information that even fifteen years ago was unthinkable, practice can prove it much harder to find multiple and truly varied views on a subject matter. Consider the following example scenarios:

- Alice is browsing the internet when she encounters an article saying that Coca Cola announced a new shape for its bottle, a first in many years. Since Alice owns some Coca Cola stock, she is curious to know more, especially about the likely business implications. It turns out that the general public is primarily interested in the history of Coca Cola bottle design and after searching online,

Alice finds mostly articles on that topic as those are the most popular. The comments on Reddit are similarly narrow in focus: the ones about the design are popular, upvoted and displayed prominently.

- Bob is interested in politics and would like to know more about the developing civil unrest in Elbonia. He is savvier than Alice and uses Google News to efficiently obtain a large number of reports on the issue. In fact, there are literally hundreds of reports and Bob is overwhelmed – he has no easy way of finding and contrasting the leftist and the rightist opinions, the local or international points of view, the articles in support of the rebels and the pro-state ones.

There is nothing special about Reddit or Google News in this context; the above anecdotes could just as easily take place on any major news sharing or aggregation website. They serve to illustrate broader, general issues that are being faced daily by users browsing news on the internet:

1. **Single point of view.** A single article almost always means a single author, a single perspective and only partial coverage of an event. The users' desire to overcome this limitation is evidenced by the success of news aggregation sites like Google News, Bing News, NewsVine and many others.

   However, even these sites represent each event with only one or maybe a handful of articles. There is a clear incentive to promote the most popular articles, thus making them even more popular and consequently exposed; a classic "rich get richer" scheme. These sites optimize for discoverability of events, not diversity of coverage.

   A similar effect happens on social link sharing websites (Facebook, Reddit, Pinterest, Fark, ...). The promote—upvote self-fulfilling cycle gives rise to the so-called *hive mind*, pushing fringe opinions and content further into obscurity. Current news sites do not provide an easy way to surface the diversity in the data.

   When readers come across an article on a novel topic, they often don't have the necessary contextual knowledge that would allow them to put that piece into perspective and judge the novel information. Such questions can be answered by providing access to the topic background, the involved people, organizations, the places that the events are occurring at, and where that article fits into the overall opinion spectrum.

2. **Information overload.** While existing news aggregators are reasonably good at collecting large amounts of articles reporting on a single news story or issue, users are mostly left to their own devices when it comes to navigating those articles. Typically, we can filter or sort by relevance and time. However, articles on a single issue differ in much more: their provenance, trustworthiness, fact

coverage, topical focus, point of view and more. Current news sites provide no way of navigating according to these criteria.

In short, the diversity in news reporting is underrepresented on the internet. In addition, individual news sources are reducing the amount of editorials and commentary [107], while simultaneously, people of each coming generation spend less time reading the news even as they age, according to a Pew[1] study [106]. In other words, diverse views are becoming scarcer *and* people are willing to invest less and less time into finding them. This is clearly an undesirable situation that we should fight against.

In this chapter, we propose a software system, *DiversiNews*, that presents news through a novel user interface that helps readers expose contrasting perspectives. The central screen of the application lets the user explore a single news story. It presents an overview of the contributing articles from across the world: what subtopics they emphasize, where in the world they were written and what their sentiment towards the story is. The individual articles are also presented, along with an automatic summary. The user can reorder the articles based on any combination of the modalities mentioned above (subtopic, geography of origin, sentiment) to surface a specific point of view. The summary changes in near real time to reflect the new focus of interest.

The system operates on top of semantically represented news: users can navigate the documents according to semantic metadata (which acts as a proxy for opinion), and the results are displayed as a summary built on top of semantic text representation from Chapter 3.

A demo version of the interface is a available online at `http://aidemo.ijs.si/diversinews`.

5.1 System Overview

Traditionally, publishers and news aggregation services create a particular, static, view on a news story. Our guiding principle was that no single view on the data and no single aggregation fits all users and purposes.

An important consideration when designing the user interface was to allow users to navigate and explore different modalities of a story. The challenge here is to show the "big picture", thus reducing information overload, but still allow the drill-down to the "raw" news articles. The latter is very important to strengthen the trustworthiness of the system: at every step, users should be allowed to verify the original content that contributed to the aggregated view created by the system.

[1]Pew Research Center is a nonpartisan, nonprofit organization that conducts public opinion polling, demographic research, media content analysis and other empirical social science research. It is one of the more prominent US organizations of its type.

5.1.1 Starting Screen

The application is composed of two screens. The first one, illustrated in Figure 5.1, enables the user to select a collection of news articles of interest. There are two ways of doings so: first, the user can search for all articles containing a keyword, using the search bar at the top of the page. Second, the user can select one of the pre-computed clusters of articles that comprise a single news story.

This part of the interface is provisional in nature. News discovery is not a focus of the proposed DiversiNews system; our intent is rather to demonstrate how we can support the *understanding* of news stories once they are discovered. To create an end-to-end news reading application, the discovery process would need to be improved. Flavio Fuart and Jan Berčič have in fact done so, creating a full-fledged iPad application, called iDiversiNews[2], based heavily on the web prototype presented here. Readers are encouraged to try it out.

5.1.2 Story Exploration

The most prominent part of the application is the Story Exploration screen (Figure 5.2) where different views on a single story can be explored. The screen is conceptually divided into two halves. The *right* half offers controls for expressing interest in articles in terms of publisher location, subtopic or sentiment. Based on the expressed interest, the the *left* half is regenerated on the fly: it contains an automatically generated summary (on top; details in Section 5.2.7) and the most relevant articles (bottom). Each article is given with its title and the first paragraph in plaintext form and links to the original website that published it.

The summary section also allows for switching to the TopicSum summarization algorithm [108], reimplemented by our colleagues [109] in the scope of the RENDER project and used in Section 5.3.1 on evaluation as a baseline.

The user controls work as follows, described in order from top to bottom as they appear in Figure 5.2:

- The **Subtopic** control is the least conventional in appearance. It shows a "map" of subtopics, represented by keywords. The x and y coordinates of keywords do not carry an inherent meaning; the two dimensions are latent, chosen so that similar subtopics and keywords get displayed more closely together. Again, the user are presented with a target icon that the user can move to any part of the topic space to focus on that subtopic. See Section 5.2.3 for technical details.

- The **Publisher Location** control shows the individual articles on a world map, along with a target icon. By moving the marker to any location on the map, the user instructs the system to focus on articles published in that part

[2] `http://ailab.ijs.si/tools/idiversinews/`; the apple store page is at `http://is.gd/idiversinews`

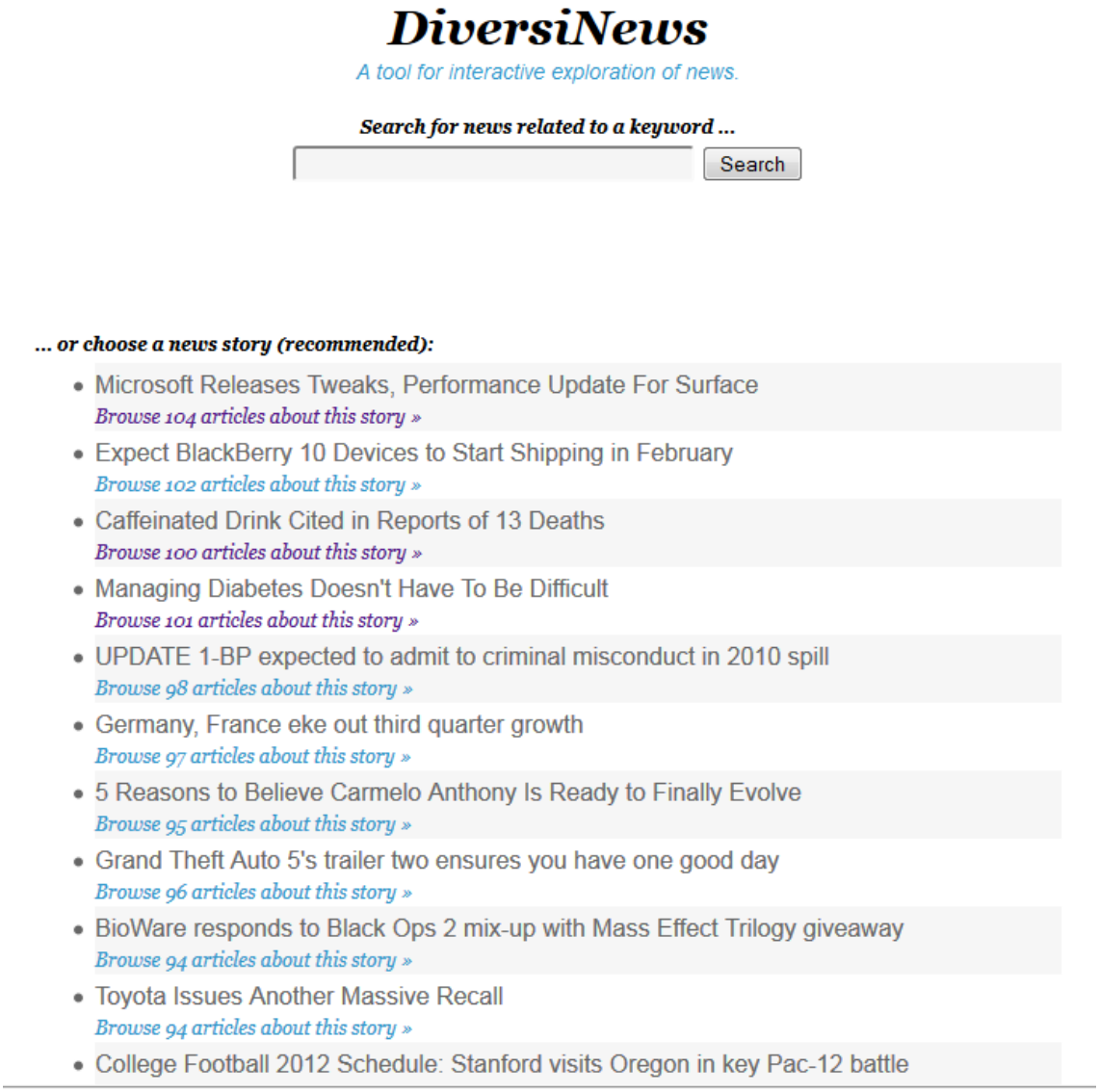


Figure 5.1: Starting screen of DiversiNews, showing two ways of choosing the articles for analysis: searching by keyword, or choosing a pre-clustered story.

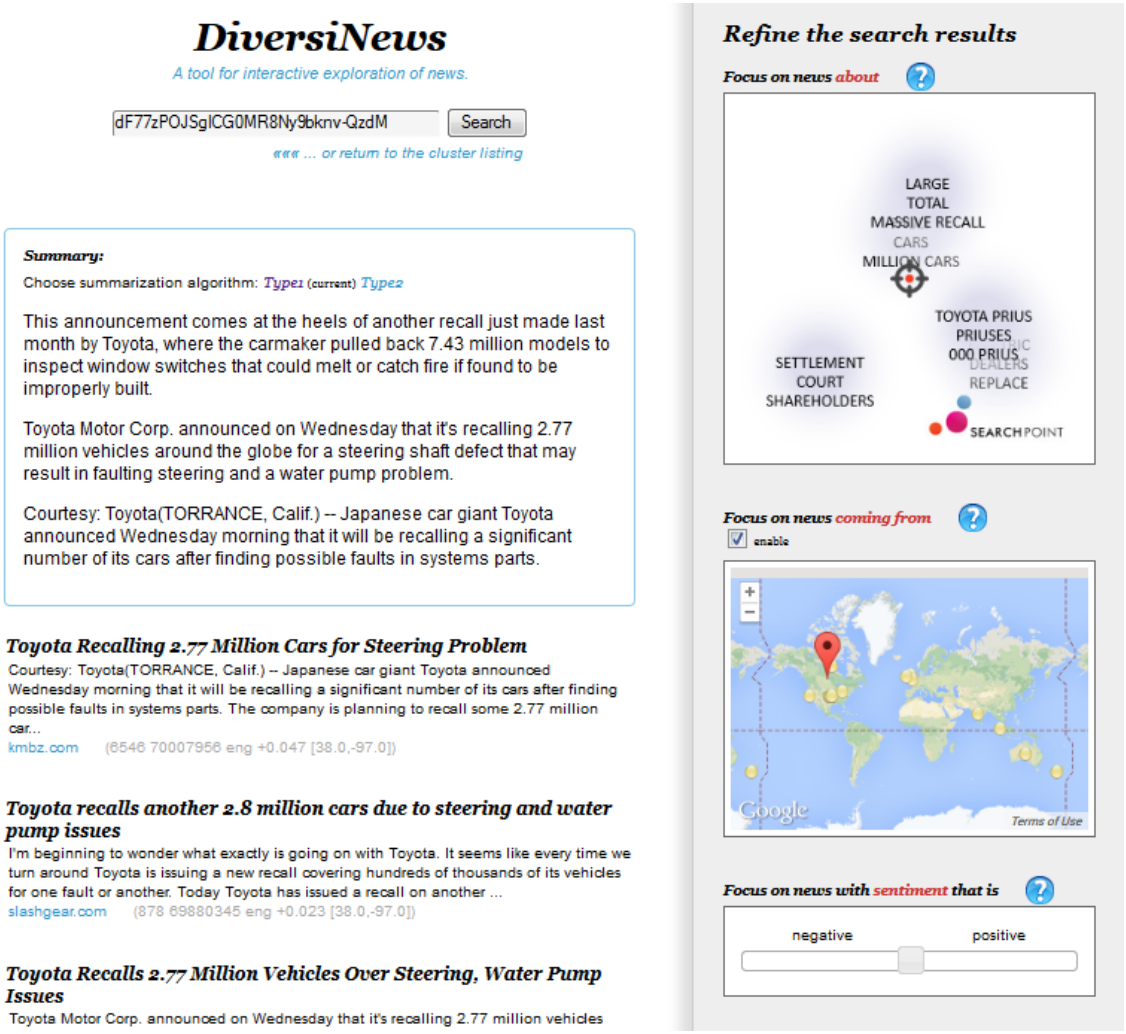


Figure 5.2: The main DiversiNews interface for exploring a story. The right hand side contains the user controls, the left hand side shows the summary and top articles, both corresponding to the user preferences.

of the world. Pale yellow dots represent publisher locations of the articles presented on the left. In addition, moving the mouse pointer over any individual article causes the associated dot on the map to get highlighted in blue. Section 5.2.4 has details on how we obtain the publisher locations.

- The **Sentiment** control is a simple slider that allows the user to focus on articles with a positive or negative spin. Section 5.2.5 describes the sentiment detection algorithm.

Both maps, the geographical and topical, serve the double function of giving the user an *overview of the active areas* (in either the geographical or topical space) but also allowing the user to *focus* on an area.

Design considerations. In choosing the dimensions users can navigate, we worked under the restriction of fitting everything on one screen, a necessity for making the discovery process fast and truly interactive. Equally important, the complexity of interaction had to remain manageable. With these considerations in mind, we settled on the three attributes described above – they are relevant to the task at hand, readily understandable and can be reasonably fit onto a single screen.

The single-screen requirement however still poses interesting challenges to presenting all the information. The challenges are exacerbated by the stark mismatch of the 2D nature of the screen and the high-dimensional nature of text data. The sentiment is the least problematic in this regard; its one-dimensional value can be directly presented on the screen. Publishers' locations like *Paris* or *US* can be relatively easily merged with background knowledge bases to produce coordinates and intuitively displayed on the map; to accommodate the small space, heatmap-like summarization is required. Lastly, the inherently high-dimensional space of topics is the most problematic and required the most tailored solution, a combination of dimensionality reduction (2D projection) and data filtering (reduction to keywords) to produce the topic map. Similarly cumbersome is the representation of the output, i.e. the relevant articles. Here, we opted for data filtering alone, in the form of an adaptive summary and top article titles.

5.2 Data Processing Pipeline

In this section, we give an overview of the implementation and deployment of the system. Section 5.1 forward-references most of the subsections that follow here, giving a clue as to how the individual components contribute to the system's functionality.

5.2.1 Overall System Architecture

The architecture, presented in Figure 5.3, is a standard client–server oriented one, with a very lightweight client that runs in a web browser and retains almost no state. Session tracking and caching of intermediate results is implemented on the server.

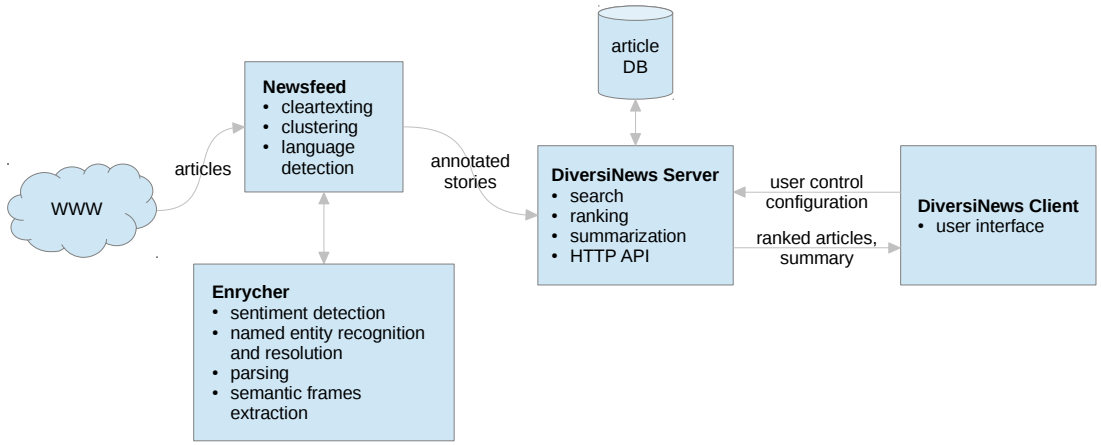


Figure 5.3: High-level system architecture.

The server part of the architecture is in turn composed of multiple parts. The structure is a natural extension of the NewsFeed pipeline presented in Section 2.4.3. Articles are first collected from the internet, stripped of everything but cleartext, clustered into stories and sent to Enrycher, the natural language processing service framework. We integrated the SDP method from Section 3.2 into Enrycher (Section 2.4.5) in order to obtain semantic frames/triplets for each article. Enrycher also assigns a sentiment score to articles.

The fully annotated articles are then passed to the DiversiNews-specific part of the server. It keeps articles in a dedicated in-memory data store, indexed for fast retrieval by story ID or content keywords. The data store also keeps cached results of these searches, which enables faster re-ranking of the articles.

The server communicates with the client via HTTP. There is essentially only one type of request/response messages. For every user action, the client sends to the server the article query (consisting of a story ID or searched-for keywords) and the configuration of user controls (topical map, geo map, sentiment). The server retrieves the articles based on the former, and ranks them and generates a focused summary based on the latter.

The server and client support an additional, nonparametric request/response exchange that generates the start screen of the application. Here, the server simply retrieves the top applications based on their recency and number of articles.

5.2.2 Data Aggregation

The NewsFeed system (Section 2.4) is used to collect the articles and their metadata. Of the available metadata, publisher geography information is particularly important. Another crucial feature is language detection. About 50% of the content is in English; we currently discard non-English articles in DiversiNews.

We considered including blog content as it is likely it would contribute more

varied opinions. However, we discovered that including blogs in DiversiNews makes for a poorer overall user experience because of their low average quality of content.

5.2.3 Subtopic Detection

To generate the topical map (topmost user control in Figure 5.2), we need to identify the subtopics of a story and present them in 2D.

To detect subtopics, we partition the articles comprising the story into at most 5 clusters using agglomerative k-means clustering based on bag-of-words vectors and the cosine similarity function. From each cluster, we then extract three terms with the highest TF-IDF weight in its centroid and use them as the description of the subtopic. It is reasonable to assume, as we do in this approach, that each relevant subtopic will have several articles focus on it prominently: if a subtopic is covered in equal depth in all articles, it will not form a cluster, but it is also not of particular interest for analyses of news diversity.

To map the articles (and with them, subtopics) onto a plane we use multidimensional scaling (MDS) [110]. The goal of this nonlinear dimensionality reduction technique is to position the articles in 2D so that their Euclidean distance in the plane is as close as possible to the cosine distance in the original bag-of-words space.

When the user interacts with the subtopic control, the relevance of individual articles to the user's choice is interpreted within this two-dimensional "compressed" topic space. Although this introduces some error, it greatly simplifies all operations and increases responsiveness of the controls.

With the above approach, we projected the inherently high-dimensional documents to two dimensions while trying to preserve inter-document distances. The resulting two dimensions are commonly referred to as "latent semantic dimensions", though this term stretches the definition of what we consider semantic data in this thesis.

5.2.4 Geo-tagging

The map widget requires the location information of individual articles. We associate each article with a news publisher, and locate those using a combination of data sources in the following order of precedence:

1. We crawled, parsed and integrated a number of public, hand-curated news publisher listings that provide for each publisher, among other information, the city and country of origin.
2. If available, we use the country code top-level domain (ccTLD) to determine the country. For example, `guardian.co.uk` is mapped to the United Kingdom.
3. As a last resort, we query the WHOIS databases and heuristically parse out the address of the domain's owner. Care has to be exercised as the address has no fixed format and is besides often that of a privacy-protecting proxy

registrant. In this and the previous method, we do not attempt to extract the city.

5.2.5 Sentiment Detection

Sentiment analysis is a natural language processing task which aims, applied to our scenario, to predict the polarity (positive, negative or neutral) of articles and opinions expressed therein.

We use a supervised model combined with background knowledge in the form of sentiment lexicons. The approach, developed by Stajner [111], uses a multi-layer classifier that first performs several independent predictions on the basis of individual feature sets (words, lexicon features, orthographic features), followed by an aggregation classifier that produces the final result. The classifier was trained on an annotated news corpus [112], combined with SentiWordNet [113], and achieves an F_1 score of 0.78.

The classifier outputs a numeric value of its prediction confidence to allow for better visualization and comparison of individual news articles.

5.2.6 Article Ranking

When user manipulates the controls in the Story Exploration screen, the articles are reranked to reflect the change. The ranking is computed based on the *goodness of fit* of each article to the position of user controls. The goodness of fit score is a linear combination of the three distances between the target value of a control and an article's value. The world map uses the logarithm of the Haversine distance formula for the great-circle distance[3], the topical map uses the Euclidean distance in the 2D plane, and the sentiment slider uses the squared difference between the target sentiment and the article sentiment.

The goodness of fit is also used as an importance weight $\text{score}_0(\cdot)$ for articles on input into the summarizer; refer to the formula in Section 5.2.7.1.

5.2.7 Summarization

In response to user's manipulation of controls, DiversiNews shows not only an appropriately ranked list of articles, but also a focused summary of their content. While automatic multi-document summarization is a reasonably well researched topic, there are some context-specific constraints under which the summarizer in DiversiNews must operate:

- The algorithm needs to be able to provide focused summaries, giving priority to some factoids, entities, sentences or documents over others. This way, we can generate summaries that reflect user's current interest.

[3] `https://en.wikipedia.org/wiki/Haversine_formula`

- Automatic summarization is not an easy problem and solutions may take a while to complete. In our case, the user interface needs to be responsive, the summaries need to be generated quickly. On the other hand, there is ample time for preprocessing, which the summarization algorithm may exploit to generate partial results in advance.

Based on the above constraints and our interest in the utility of semantic representations of text, we developed a multidocument summarization algorithm called **FrameSum** [8]. FrameSum performs extractive summarization, meaning that it selects a subset of sentences and presents them as the summary in an unchanged form.

The algorithm is built around a key assumption: in the multi-document summarization setting, a strong signal for the importance of a piece of information is that piece being repeatedly reported by multiple sources. In practice, for newswire, we observe this to be a sound assumption with 10–20 input documents (depending on the task) or more.

At the same time, we have to consider the standard tradeoff of diversity vs. relevance: an algorithm that only considers relevancy will end up constructing the summary out of sentences that convey almost exactly the same, albeit very relevant, information. We combat this by requiring that no two sentences in the final summary be too similar to each other.

In the following subsections, we present the algorithm that formalizes these two considerations.

5.2.7.1 Preprocessing and Initial Content Scoring

We represent all input data in the semantic space, using the SDP algorithm presented in Section 3.2.

This representation loses a lot of detail, but allows for similarity comparisons between sentences and even sentence fragments that goes beyond simple string matching. In fact, the basic unit of information in FrameSum is a frame, not a whole sentence.

In this first stage of the algorithm, each frame is scored separately. The score of a frame T is based primarily on its position in the document, a well-established heuristic/feature [114]:

$$\text{score}(T) = \frac{\alpha \cdot \text{score}_0(T) + \beta^{\text{pos}(p(T))}}{\sum_{p'} \text{sim}(p(T), p')}$$

where:

- $\text{score}_0(T)$ is a prior belief about the importance of a frame or its originating sentence. This allows FrameSum to be used in a "guided summary" framework where a certain aspect of the summary is emphasized – for frames or sentences relating to that aspect, we can boost $\text{score}_0(T)$. If FrameSum is used on its

own or if no emphasis, $\text{score}_0(\cdot) \equiv 1$ can be used for a uniform prior and hence a "standard" multi-document summary.

- $p(T)$ is the sentence containing T.
- $\text{pos}(p(T))$ is is the zero-based index of the sentence within the document.
- $\text{sim}(\cdot, \cdot)$ is the similarity function between sentences, defined as the Jaccard similarity coefficient[4] for the sets of character 4-grams of the two sentences and ranges from 0 (no similarity) to 1 (identity).
- $\alpha = 0.4$ and $\beta = 0.8$ are constants determined empirically by observing behavior on several ten collections of documents.

In the nominator, the exponentially decaying factor quantifies the intuition that especially in news reporting, the important facts tend to be given early on. α is a smoothing factor, corresponding to the firmness of our belief in the $\text{score}_0(\cdot)$ prior.

The denominator is a normalization, particularly affecting sentences that not only have similar content but are almost completely identical. This tends to happen frequently with journalistic texts as the content is often partially copied from a press release or a news syndication network's article.

As pointed out previously in Section 5.2.6, in DiversiNews the score prior $\text{score}_0(\cdot)$ is based on article ranking for a specific user control configuration. Due to a project-imposed schedule, we only evaluated a simple step-function weighting scheme: frames from the top k ranked articles ($k = 20$) are weighted with 1, the remaining frames with 0.

5.2.7.2 Frame Graph

In the second stage, the frames are connected into a directed weighted graph with weights representing the *information flow* between frames. Intuitively, a large flow from T to T' means that T conveys a lot of T'. Formally, the information flow w between two frames T and T' is defined as follows:

```
 1: w ← 0                                                    ▷ "Initialize flow"
 2: for all attribute a ∈ {subject, verb, object, instrument, location, time} do
 3:     X ← value of a in T
 4:     X′ ← value of a in T′
 5:     if X = X′ then
 6:         w+ = 1
 7:     else if X is a hypernym of X′ then
 8:         w+ = 0.7                                         ▷ X strongly infers X′
 9:     else if X is a hyponym of X′ then
10:         w+ = 0.25                                       ▷ X weakly infers X′
11:     end if
12: end for
13: return w
```

[4]The Jaccard similarity coefficient of two sets A and B is defined as $\frac{|A \cap B|}{|A \cup B|}$.

5.2.7.3 Greedy Optimization

In the final stage, the most relevant frames are selected greedily. The sentences containing them are promoted into the summary. To determine the relevance of a frame, we first define its information content. This is initialized to

$$\mathrm{IC}(T) = \mathrm{score}(T) + \sum_{T'} w_{T \to T'} \mathrm{score}(T')$$

where $\mathrm{IC}(\cdot)$ is the information content, $\mathrm{score}(\cdot)$ was defined above and $w_{T \to T'}$ is the information flow as defined in the previous paragraph. The following two steps are then repeated greedily until the desired length of the summary is reached:

1. Promote the shortest sentence containing the frame T with the highest $\mathrm{IC}(T)$ into summary. Ignore sentences that were originally part of quoted speech or that begin with a linking word ("however", "also", ...). If no suitable sentences are found, skip T.

2. For every frame T contained in this sentence, decrease $\mathrm{IC}(T')$ for all remaining input frames T' by a factor of $1 - w_{T \to T'}$.

The fact that $\mathrm{IC}(T)$ is based on the score (which in turn is based on frequency) ensures that the summary contains relevant information. The second step of the greedy iteration above ensures that the information contained in the summary is not redundant. Intuitively, the performed decrease in $\mathrm{IC}(T')$ can be understood as "if T is told, then $w_{T \to T'}$ of T' is already told as well, so its information content decreases by that much."

The order of the output sentences is determined based on the sentences' positions in their respective original files. In case of ties, higher-scoring sentences are placed first.

5.3 Evaluation

Most of the components have been evaluated individually before, with results presented in their respective reference papers. Here, we present the evaluation of the crucial, user-facing components: the summarizer as one of the main feedback mechanisms to the user, and the user interface as a whole.

The evaluation effort was led by Daniele Pighin at Google Switzerland within the scope of the RENDER research project [115].

5.3.1 Summarization

With the summarizer, we tried to capture their overall quality of summaries as well as their responsiveness to changes in user focus. For this evaluation we randomly

selected 20 news stories and, for each of the two summarizers, we generated 4 different 3-sentence summaries based on different states of the sentiment and subtopic widgets:

- *Neutral:* controls are left in the default position.
- *Topic:* the topics control is moved to overlap a randomly selected target subtopic. The sentiment widget is in the default position.
- *Positive:* the sentiment control is moved to the far "positive" side. The subtopic control is in the default position.
- *Negative:* the sentiment control is moved to the far "negative" side. The subtopic control is in the default position.

Two expert annotators (journalists) have then annotated the summaries generated for each news collection according to the following dimensions:

- **Fluency:** to what extent is the summary understandable and grammatically adequate? Each summary was rated on a 3-point Likert scale:

  1. *Inadequate:* by and large disconnected, the summary is difficult to understand, inadequate for user consumption.
  2. *Adequate:* there are local disfluencies, the summary is understandable even though it is not perfect. It is adequate for user consumption.
  3. *Human grade:* indistinguishable from a human-generated summary or nearly so.

- **Informativeness:** does the summary contain relevant information with respect to the selected news collection? How useful is the information in the summary? Each summary was, again, rated on a 3-point Likert scale, with values:

  1. *Inadequate:* the summary is uninformative, nothing relevant can be learned from the summary.
  2. *Adequate:* the summary conveys useful information, but it's not exhaustive.
  3. *Human grade:* the summary is informative and captures relevant aspect of at least one of the main topics in the news cluster.

- **Topic sensitivity:** among the 8 summaries generated for each news collection, the raters have been asked to select up to 2 of them which are more clearly centered on a specific topic.
- **Sentiment sensitivity:** among the 8 summaries generated for each news collection, the raters have been asked to select up to 2 of them which clearly exhibit a positive sentiment, and up to 2 which exhibit a clearly negative sentiment.

	Fluency			Informativeness		
Summarizer	1	2	3	1	2	3
TopicSum	34.95	39.81	25.24	30.10	40.78	29.13
FrameSum	34.95	41.75	24.27	33.01	39.81	28.16

Table 5.1: Distribution of raters' decisions concerning the fluency and informativeness of the two summarizers. The columns 1, 2, and 3 correspond to judgments of *Inadequate*, *Adequate* and *Human Grade*, respectively.

- **Geography sensitivity** was not evaluated as we considered it too hard for evaluators to grade a summary as e.g. "very Italian" or "only moderately middle-Eastern".

We the approach against a state of the art topic-modeling summarizer, TopicSum [108], based on a probabilistic graphical model. Like our summarizer, TopicSum is sentence-extractive in nature. It works by first estimating the word distribution of the whole input document collection, then finding a set of sentences that estimate that language model as closely as possible under the Kullback-Leibler divergence.

We calculated the inter-annotator agreement by means of Intra-Class Correlation (ICC) [116] over 40 summaries. The evaluators reached a high agreement on Fluency (0.6) and Relatedness (0.71), while we observed a relatively low agreement on Polarity (0.44) and Informativeness (0.19). We believe the reason is that the first two dimensions can be associated with objective criteria, while the latter are inherently more subjective. In particular, the annotation for informativeness is complicated by the fact that real-world news "stories" are imperfect clusters of articles, not infrequently aggregating several different events; that makes the decision on what is truly relevant even more subjective.

Results. The results are listed in Tables 5.1 and 5.2. The two approaches are very similar in performance when it comes to simple, unbiased summarization (Table 5.1). It is a testament to the amount of information encoded in WordNet and semantic relations that the relatively sparse semantic representation – averaging 15 frames or 45 WordNet concepts per article – can achieve comparable performance to a state of the art method that uses the full text.

The one notable exception where the FrameSum method fails is the summarizers' ability to adapt to topic changes (tight-hand column of Table 5.2); here, TopicSum significantly outperforms our method. Analysis shows this to be due to the overly sparse data representation used in FrameSum, due partially to the nature of semantic frames and partially to the modest recall of our frame extraction method. The key concepts extracted during the text semantization process are enough to successfully capture the gist of a story. However, they can discard or overlook the concepts associated with the fringe (sub)topics, making it much harder for FrameSum to

	Sentiment widget			Topic widget		
Summarizer	P	R	F_1	P	R	F_1
TopicSum	0.75	0.59	0.66	0.98	0.8	0.88
FrameSum	0.86	0.53	0.66	0.75	0.23	0.35

Table 5.2: Comparison of the two summarizers in terms of their sensitivity to the UI controls. Precision (P), Recall (R) and F_1 describe the *evaluator*'s ability to identify, in a set of 8 summaries, the one that the algorithm produced to be e.g. negative in sentiment.

capture them in the summaries. In those cases, the users had to defer to reading the headlines of the top articles, ranked in accordance with the Subtopic control. While this does not endanger subtopic discovery (which is performed via the subtopic map, see Section 5.2.3), it is a limitation of the FrameSum summarizer that is unfortunate in the particular use case of DiversiNews.

There is another notable limitation to FrameSum hat does not get easily revealed in the news summarization scenario: it performs poorly when the set of input documents is small. We observe the quality of the output to noticeably and sharply degrade at sizes of 10 to 25 documents, depending on the dataset. This is due to the large feature space of triplets on which the algorithm operates – it takes more data to overcome the sparsity compared to the word-space of TopicSum.

5.3.2 User Experience

To evaluate the reaction of users to the news browsing paradigm proposed in this thesis, we designed an experiment divided in two main parts: the **user interface (UI) evaluation**, where we measured if the UI controls are intuitive and well-suited to the task; and, most importantly, the **perceived usefulness evaluation**, where the goal was to see if users find the interface useful.

The user experience evaluation involved 16 subjects. Each subject has gone through the two stages of the evaluation in the same order and in the same amount of time. Of the 16 subjects, 14 were casual readers of web news portals and 2 were professionals, news operators working for a press office. All the subjects declared to use the internet for several hours every day. Two thirds of the subjects mostly access the news through online news portals such as Google News, 12.5% declared to access news mostly through social networks and 6.25% from newspapers. Surprisingly, none of the subjects considers either radio or television as their main source of news.

User Interface. To assess the intuitiveness of the UI, test subjects were exposed to static images of the interface and asked to build an expectation concerning the function of the UI components without interacting with them. After that, the subjects independently used the interface for a set amount of time. Between the

two activities and at the end of the second, they were asked questions about the quality of the interaction, the responsiveness and the ergonomics of the interface. This session provided useful evidence concerning the intuitiveness and accessibility of the interface.

All the views of the interface have been found to be highly self-explanatory to the large majority of the subjects. They easily identified the relations among the dynamic parts of the interface (e.g., that the provided summary synthesizes the news in the ranked list, and that acting on the controls would reorder the news and update the summary). The vast majority of the users confirmed that the interactive panels behaved as expected.

A major unexpected finding however was that during the static inspection, about half of the subjects built the expectation that acting on any of the UI controls would have an effect also on the others. For example, they imagined that changing the position of the sentiment slider would also affect the content of the topics panel to show the topics having more positive connotation. We have since altered the labeling of the widgets to further stress that the panels are independent.

Usefulness. In this part of the evaluation, the subjects answered questions about the utility of the individual components and their potential impact on their news-browsing habits. They answered specific questions about the potential of the different components to highlight and emphasize diversity of opinion in news.

The evaluation included over 50 questions in total and we only present an outline here. In general, the raters found summaries to be an effective device to capture and represent relevant information and diversity of opinion and confirmed that the controls succeed in modeling different dimensions and provide a more balanced paradigm for online news consumption. They did also suggest a number of small UI improvements. Most notably, users wished to be able to see the political outlook of news publishers and wished for a cleaner overview of the subtopics than the topics panel offers. Importantly, however, the concerns present the minority of feedback.

Figure 5.4 graphically shows the distribution of the answers to the key questions that we asked about utility. Similarly positive is the feedback to the questions not shown in the figure: over 80% of the subjects found that summaries are at least adequate in quality; just below 80% believe that summaries are an effective way of letting relevant information emerge and stress diversity in news; over 80% find all the interactive panels implement a functionality that is considered desirable in a news browser and is instrumental in easing the discovery of diversity; etc.

The users have especially appreciated the geographic source widget, while the sentiment widget is found to be less effective than the others in letting diversity emerge. We speculate that this is because the language of news is most often objective and lacks sentiment.

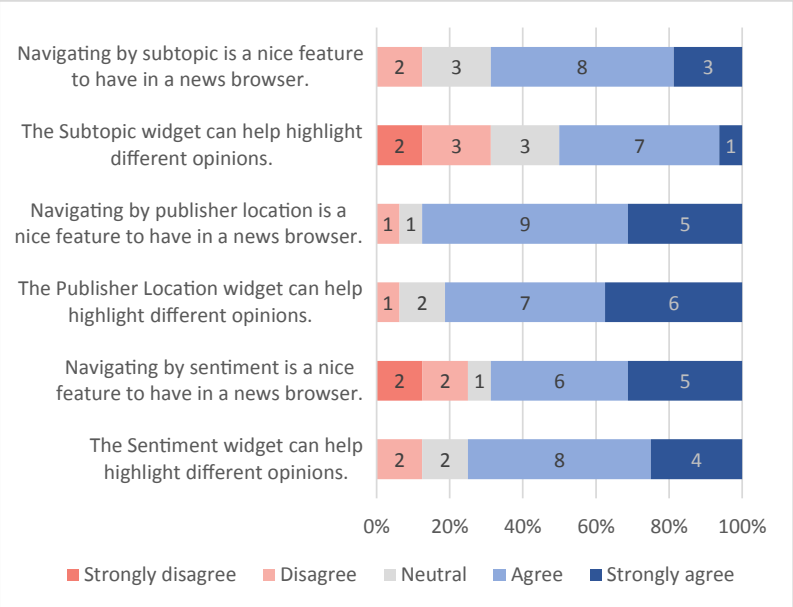


Figure 5.4: Distribution of the answers to some of the perceived utility evaluation questions.

Chapter 6

Conclusion

In this final chapter, we summarize the implications of using semantic data representatitons and methods in the tasks of template construction and opinion mining. Some of the findings are quite general and likely apply to large areas of text mining, while some of the conclusions are specific to our selected tasks.

In Chapter 3, we compared methods for arriving at structured representations of text of varying complexities, and seen that in choosing the representation, there is a precarious balance to keep between it being rich and very theoretically informative on one side, and being unrealistically hard to automatically extract on the other side. We arrived at this observation by creating two text semantization approaches, each integrating existing technologies into a single method, but differing in complexity. SDP, the simpler approach, integrating a dependency parser and WordNet, was demonstrated to have a more appropriate level of representation complexity than MSRL, the approach that combined a semantic role labeler and Cyc.

As demonstrated by the evaluation, automated structuring of text still has a long way to go. Because of the large gap between the textual and purely semantic representation, it is almost inevitable for approaches to employ long pipelines. While it is possible to achieve reasonable accuracy at each individual step, the pipeline length means a large number of errors accumulates. This would remain a problematic factor even if we improved the individual stages significantly. As a potential venue of research, learning the tasks jointly might remedy the problem of long pipelines. A single "deep" system would likely have a slow learning rate; very recently, the deep learning community has partially overcome this by learning the stages jointly but adding them to the learning model gradually.

In Chapter 4, we applied the semantic frame data representation to the task of unsupervised domain template construction for the first time, designing and implementing two novel methods. Both search the space of relational triplets to construct those that are not overly generic yet have strong support in the in-domain documents.

We evaluated the approach on five domains and achieved results that are at least comparable with current state of the art in terms of quality while also providing finer-grained type information about the template slots.

In Chapter 5, we developed DiversiNews, a web application that allows users to gain a more complete insight into different opinions on a news event. It does so by aggregating reports on the story from across the world, then allowing the user to navigate them with regards to topical focus, geography of origin and sentiment to let the opinions emerge. We further presented a focused multidocument summarization method based on the semantic frame representation.

A user study testifies to the utility the solution as a whole. Further, comparative evaluation of the summarization algorithm with the state of the art baseline shows that combined with sufficient background knowledge (here in the form of the WordNet taxonomy), the semantic representation can achieve competitive results with very few input features. Because the inputs are pruned so heavily, the semantic approach can fail more easily by pruning away information that is unimportant under the algorithm's assumptions but important for the task at hand. This mismatch in assumptions is what caused the summarizer to perform poorly in focusing the summaries on a narrow subtopic of the input documents.

In summary, the transformation of text to ontology-aligned frames or triplets for use in template construction and opinion mining brings important **benefits**:

- *Feature selection*: Only the key fragments of sentences are retained, following heuristics based on the usual structure of natural language sentences. This significantly reduces the size of inputs and potentially simplifies subsequent optimization tasks, like the selection of domain template-worthy fragments or gauging the similarity of two sentences in our case.

- *Data normalization*: Aiming for a canonical representation of information simplifies comparison of independent pieces of data (text) and reduces sparsity in the data. With a semantic representation, methods are much less exposed to issues of conjugation, tenses, synonyms, etc.

- *Access to background knowledge*: Having data that is aligned to a knowledge base allows taking advantage of existing, possibly time-consuming work done by others. In our case, we exploit WordNet's hypernym taxonomy.

At the same time, we also have to note several **limitations**:

- *Compounding errors.* The semantization pipeline consists of multiple *successive* stages, and as a consequence their errors compound. In addition, to make the processing tractable, all stages produce hard decisions (as opposed to giving weighted top alternatives), making error recovery in later stages nearly impossible.

- *Brittleness.* This is an alternative take on the previous bullet point. Each stage of the pipeline is designed under some assumptions. If any of those sets of assumptions is violated, the performance of the whole pipeline suffers. A notable strong assumption is that of language well-formedness — the methods proposed here would do very poorly on microblog (e.g. Twitter) data.

- *Computational cost.* Parsing in particular is a combinatorially complex operation; speeds are typically in the range of a few sentences per second. Although it parallelizes trivially, there is a limit to the size of datasets we can conveniently process.

- *Low recall.* Full "machine reading" is still far from a reality. Therefore, text semantization methods must make do with only recognizing *parts* of the language, causing a lot of information to be discarded. Hopefully, the discarded data is less important and the net effect can be positive (see the bullet points with benefits above), especially with large redundancy in input data. However, sometimes the effects are also clearly negative. For example, our methods in Chapter 3 will not produce any output for the sentence "President's visit to China was productive." because "visit" is not a verb; other methods might miss something else. Note we are not claiming this is an unsolvable problem, but rather that there are a lot of cases like these to consider.

- *Required language resources.* Another limitation of semantic approaches is that they require language-specific methods and especially language resources, which are complex, time-consuming, and costly to produce. Non-English languages are slowly catching up in terms of available resources, but at the same time, English resources continue to grow and evolve, so there will always be a certain discrepancy in how far semantic methods can get for different languages.

- *Complexity of implementation.* Though it does not directly influence the results, this can be an important factor. Having to integrate multiple software components (e.g. part of speech tagger, parser, named entity recognizer, word sense disambiguator), their corresponding knowledge bases, and any application-specific logic requires long development times. The situation is luckily improving constantly as deep NLP techniques are becoming more commonplace and user-friendly libraries and tools are emerging.

Most of these limitations stem from the longer-than-usual pipelines. As we saw, many of the errors can be relatively successfully offset by data redundancy and the value of background knowledge. In the future, as the preprocessing stages grow in reliability and performance, and as the amount of exploitable background knowledge grows in quantity and interlinkedness, semantic methods may well still see a performance boost that sets them above simpler models traditionally based on word unigrams or n-grams.

Although the accuracy of text semantization methods is still very far from human-grade, we however demonstrated that with a suitable level of data redundancy, the frame/triplet representation of text can be used in text mining methods to achieve results comparable to those of more traditional, token-based approaches. At the same time, they may provide additional benefits like relevant information

from a knowledge base (slot types in Chapter 4) or a very lightweight input data representation (summarization in Section 5.2.7).

In total, introducing semantic representations of text makes the processing pipeline much longer, which brings very real disadvantages associated with longer processing times, considerably larger implementation effort, harder reproducibility of results and harder upkeep of all the pipeline components. At the present moment, we therefore suggest that the integration of background knowledge via forcing a semantic representation might bring more disadvantages than advantages. At the same time, it is clear that background knowledge does have a lot of value and is therefore certainly advisable to use in situations where data is represented in a more structured form (e.g. tables) that is more amenable to integration with background knowledge sources. After all, our methods far from cover all possible text semantization approaches nor all problem areas in which to use semantic representations of text.

6.1 Contributions to Science

The thesis makes the following new contributions to science:

- *Text semantization methods.* We propose two new methods (SDP and MSRL) for semanticizing text and evaluate their performance quantitatively, in terms of accuracy, and qualitatively, in terms of SDP's role in more complex natural language processing tasks.

- *Domain template construction methods.* We are the first to integrate background knowledge into the task of unsupervised construction of domain templates and solve the task in two ways, both using a data representation that is significantly different from the norm. The CT method achieves performance at least on par with the state of the art and additionally produces, unlike the work so far, fine-grained type constraints for template slots.

- *Domain template construction principled evaluation and data.* Evaluation for the task of domain template construction is complicated, and there was so far no well-documented evaluation methodology or sizable public datasets and golden standards for comparing methods. We provide both.

- *Exposing opinions in news.* We present a full-stack, integrated system for news collection, processing, aggregation, manipulation, and opinion discovery. By inferring multiple modalities of news (geography, topics, sentiment) and presenting them in a unified interface, we enable users to explore opinions in news in a manner significantly different from existing tools, with much easier and more explicit access to the diverse views on a topic.

- *Content understanding through adaptive summarization.* Also novel is the use of near real-time adaptive summary as an interface element: users get

immediate feedback on their selection of perspective, without having to read several articles. The multi-document summarization process is broken down into a computation-intensive preprocessing stage (text semantization) and a fast focused summarization stage for arbitrary weights on input sentences.

In addition, we presented NewsFeed, a system for real-time web news crawling, extraction, metadata annotation, enrichment and aggregation. While building such a system is mostly an engineering challenge, it is an important enabler technology for many experiments, existing and future, that wish to evaluate data mining methods on streaming textual data, or demonstrate the value of research in a real-world scenario.

6.2 Future Work

Text semantization. We believe it may be worth further exploring text semantization techniques based on simpler structural features, e.g. POS tags or chunker output. This would potentially limit some of the issues with the approaches presented here. Most notably, it would speed up the computation, making it possible to apply the method on large text corpora. For an approach almost necessarily plagued by low recall, the ability to process large amount of data can help mitigate that problem. Replacing the parser with simpler methods would also simplify the pipeline somewhat, making it more robust as parsers tend to generalize relatively poorly across different domains [117, 47]. The downside, of course, is that the semantization method would have to infer frames or an equivalent representation based on less informative input features.

Related to full semantic role labeling, it might be worthwhile looking into bootstrapping from existing labeleres to produce more training data. We see the lack of data as a major impediment; most papers and challenges on SRL limit themselves to only the few best-annotated frames.

Domain template construction. There is an aspect of performance for domain template construction methods that the standard measures of precision and recall do not fully capture: redundancy of patterns. The CT method presented here suggests several patterns that describe essentially the same role. It would be worth looking into clustering the patterns to obtain richer and more robust descriptions of more clearly disjoint roles. The clustering could possibly be based on role filler collocations. The danger is of course in the errors introduced by adding yet another layer to the processing pipeline. Our preliminary experiments in this direction failed to produce good results; they were however not extensive enough to warrant conclusions.

Alternatively, it would also be interesting to relax the restriction that every document reflects a single event type or domain. An initial set of roles could be defined based on a semi-supervised approach like our CT. These roles might then be

used to bootstrap annotation of individual documents or small collections without knowing their domain in advance, and a document could be assigned roles tailored to its content specifically. For example, a document on civil unrest may mix and match the roles from the "bomb attack" domain and a hypothetical "coup" domain. This approach is close in spirit to the open information extraction efforts like TextRunner [12] and NELL [20].

Exposing diversity in news. News sources can differ significantly in trustworthiness and quality. While the whole point of DiversiNews is to show many opinions and reports, not just those of the likes of CNN and Reuters, the truly low-quality sources riddled with spam or extremely short blurbs do have to be removed. This is especially true of blogs. It would be beneficial to research methods for estimating the quality of news sites and blogs in particular. Blogs, although on average of lower quality, have high potential for uncovering new viewpoints on a topic. Carpenter [118] showed that "citizen journalism" produces more diverse content than the mainstream media.

Diversity of opinions is not bound to only geography, topical focus and sentiment — those are simply the proxies we chose in DiversiNews. Other dimensions could be included in the framework and more principled research done into which ones are best suited to achieving our goal. For example, time almost certainly plays a role in forming opinions, as does the author's political affiliation. In addition, the geography aspect of diversity could be much better exploited with semi-automated ways of bridging the language barrier and not having to constrain ourselves to news in English only.

To make DiversiNews useful to real-world users, the current interface would have to run on top the real-time NewsFeed stream. This provides challenges in learning to present users with the most interesting, opinionated stories. For example, a dry technical report on a minor earthquake does not differ much between the dozens of agencies that will publish it. On the other hand, news with a political twist, reports on sports events, articles on upcoming technologies etc. include a higher degree of authors' personal opinions and are more interesting to explore. This aspect has to be balanced, of course, with the objective interestingness of the story.

Some of these research venues are being actively pursued by colleagues in the Artificial Intelligence Laboratory at Jozef Stefan Institute in the scope of related projects. See for example the iDiversiNews application (an extension of DiversiNews as presented here, due to Jan Berčič and Flavio Fuart) in the Apple App Store (`http://is.gd/idiversinews`), or the Event Registry, due mostly to Gregor Leban, at `http://eventregistry.org`.

6.2.1 Unexpected Problems and Limitations

As with every venture into the unknown, we encountered some unexpected rough spots during the work on this thesis, and in a few areas fell short of what we thought

was reasonably achievable at the outset. The details we found deceitful may be valuable to others attempting similar work in the area. They may be "obvious" or known in retrospect, but without knowing to look for them, they make for easy oversights.

When it came to semantic frame extraction, we were the most surprised by the disconnect between structured resources (ontologies, semantic role systems) and text. Most systems claim to have some associated syntax information and/or annotated data. They do, but in practice, the data seems incomplete or too low in volume to train statistical annotation models. This is easily explained by the cost of manual labor, but only became apparent to us once we actually attempted to build models. In Section 3.4, these observations bring us to conclusion that the frame extraction method that works best for us is SDP, which is disappointingly simple. While simplicity is good in and of itself, it also limits the "depth" and complexity of information we are able to extract. Originally, we were hoping for a more principled method and for outputs with a stronger semantic meaning.

However, when trying to *use* the generated frames for domain template construction and automatic text summarization, it proved very fortuitous that our frames strike a compromise between full semantics and simplicity. The most unexpected problem at this stage was the rigidity and sparsity of WordNet. By sparsity, we mean that while it does a reasonably good job of encoding hierarchical relationships, in particular hypernymy, the concepts exhibit all sorts of other relations as well. Verbs are particularly problematic. For example, "tear apart" and "kill" are not related at all in WordNet. Also, while the discrete solution space is smaller than the space of free strings, it is computationally even more expensive to explore than we expected; it's easy to underestimate the combinatorical explosion. We thought we would be better able to compensate for those deficiencies with corpus statistics.

In DiversiNews, we met the most challenges when trying to represent subtopics of a news story. With an abundance of existing work in clustering, fuzzy clustering, latent semantic dimension detection etc., it seemed the task should be relatively easy. However, we fell short of our initial expectations. Clustering itself, as well as the presentation of clusters to the user, proved harder than expected.

6.2.2 Applicability to non-English Languages

An important advantage of semantic representations is that they are based on *concepts* rather than words, and thus language-independent. Once we represent the text in a semantic form, all downstream methods (e.g. domain template construction, article re-ranking in DiversiNews, and the FrameSum summarizer from this thesis) will work without a single change, regardless of the language of the original text. All background knowledge (e.g. that encoded in Cyc or WordNet) is language agnostic and reusable too. However, the methods for converting text to a semantic form depend, to varying extents, on resources or heuristics that are language-specific. Since we only presented results for English, it is natural to ask ourselves if comparable

resources exist for other languages (with a focus on Slovenian), and if they do not, how costly and time-consuming it would be to introduce them.

The resources fall into two main groups: static resources (dictionaries, verb usage patterns, labeled training data, etc.) and tools (tokenizers, POS taggers, parsers, etc.). Let's look at them in order of complexity, from low to high.

Tokenization Tokenization is easy for languages that delimit their words with spaces, but non-trivial for those that do not (e.g. Japanese, Chinese, Korean). However, being such a rudimentary task necessary for almost any further processing, it is well solved for the major languages.

POS tagging Part of speech tagging is one of the most basic natural language processing tasks, and has therefore been made available for a number of languages. Even Slovenian, for example, got its first POS annotator in 1997 [119]. The required features are relatively easy to construct and there is little in terms of dependencies; the main problem is acquiring enough training data.

It is however worth noting that just as languages differ in vocabulary, they differ in grammar, too, so different sets of POS tags apply to different languages. A normalization layer would thus be required for the downstream systems to function unchanged. Luckily, we only use coarse grammatical roles in our work: nouns, verbs, and pronouns, and those are almost certain to exist in all major languages. For several languages, the Universal Dependencies project[1] provides mappings from language-specific tags to coarse(r) language-independent tags we could use with our approach.

POS tagging alone is enough to build a rudimentary semantic frames from text [92], making the approaches discussed in this thesis theoretically viable even for languages that lack more advanced NLP tooling. In practice, however, the decreased precision and recall are likely to critically impact the quality of end output.

Parsing There are many variants to the task of parsing: shallow parsing or chunking, constituency parsing, dependency parsing, and more. The SDP approach to text semantization discussed in Section 3.2 operates on dependency parses, but those in turn are usually derived (using handcrafted rules, see e.g. [73]) from constituency parses. While dependency parsers are somewhat less common, constituency parsers have been developed for a number of languages. The same relationship holds between English and non-English languages as it did for POS tagging: English has bigger datasets, higher accuracy, and more readily available tools, but non-English is doable too, and has been done. The framework is generic and "English" parsers can be reused, even for languages like Japanese; the problematic part is getting the data. For example, the Slovenian dependency treebank [120] has over 300 000 words, which is enough to get to about 60% accuracy on labeled dependencies [121].

[1] `http://universaldependencies.github.io/docs`

Like with part of speech tagging, different languages give rise to slightly different sets of relations between sentence constituents, so the labels employed by parsers differ from language to language. What's more, not even parsers within a single language may not define relations in the same way and not use the same set of labels (for English, compare e.g. MiniPar [122] and Stanford Parser [90]). Parser-specific normalization would therefore possibly be needed before subsequent steps – be that feature generation for our MSRL approach (Section 3.3), or the rule-based conversion of trees into frames in the SDP approach (Section 3.2).

Coreference resolution can be seen as a subtask of (semantic) parsing. Here, too, the biggest problem is getting enough annotated data. For example, Hendrickx et al. [123] report annotating a corpus of over 300 000 words to create a reasonably performing coreference resolution system for Dutch. This is comparable to what is needed for POS tagging [119], but the use case is more limited and the expense therefore harder to justify. I am not aware of a coreference resolution system for Slovenian. Coreference resolution is "optional" for text semantization in that the pipelines will still work without it, but recall will suffer significantly as a lot of facts in natural langauge are expressed using pronouns.

Semantic role labeling With SRL, the required amount of training data is gargantuan, and as we saw in Section 3.3.1, problematic even for English. The time and money requirements make it unrealistic to build a comprehensive set of verbs and roles with sufficient training data in the near future. There is research in doing SRL for non-English languages; notably, the CoNLL-2009 challenge provided datasets for Catalan, Chinese, Czech, German, Japanese, and Spanish [124]. However, the goal is not to create a comprehensive SRL solution, but rather to see on a limited set of roles how well the systems can handle new languages (with new grammatical structures, poorer tooling etc.). The results vary by language; in general, F1 scores tend to be about 5% lower than for English [125].

In summary, the semantization technology is capable of consuming non-English languages, but depends on non-trivial, costly amounts of training data. Therefore, while work has been done on many languages other than English, the training data falls short of its English counterpart, and so does performance of the resulting systems.

Bibliography

[1] M. Trampuš and B. Novak, "Internals of an aggregated web news feed," in *Proceedings of the fifteenth international Information Science conference IS SiKDD 2012*, pp. 431–434, 2012.

[2] M. Trampuš and D. Mladenić, "High-Coverage Extraction of Semantic Assertions from Text," in *Proceedings of SiKDD 2011 at the Information Society multiconference*, 2011.

[3] M. Trampuš and D. Mladenić, "Constructing Event Templates from Written News," in *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology-Volume 03*, pp. 507–510, IEEE Computer Society, 2009.

[4] M. Trampuš and D. Mladenić, "Approximate Subgraph Matching for Detection of Topic Variations," in *Proceedings of the 1st International Workshop on Knowledge Diversity on the Web (DiversiWeb 2011) at 20th International WWW Conference, Hyderabad, India*, pp. 25–28, 2011.

[5] M. Trampuš and D. Mladenić, "Constructing Domain Templates from Text: Exploiting Concept Hierarchy in Background Knowledge," *Information Technology and Control*, vol. 43, no. 4, 2014.

[6] M. Trampuš, F. Fuart, J. Berčič, D. Rusu, L. Stopar, and T. Štajner, "(i)DiversiNews – a stream-based, on-line service for diversified news," in *Proceedings of SiKDD 2013*, 2013.

[7] M. Trampuš, F. Fuart, D. Pighin, T. Stajner, D. Rusu, and L. Stopar, "DiversiNews: Surfacing Diversity in Online News," *AI Magazine*, vol. to appear, 2015.

[8] D. Rusu, M. Trampus, and A. Thalhammer, "Diversity-Aware Summarization - RENDER Project Deliverable D3.2.1," tech. rep., RENDER project, 2013.

[9] S. Sarawagi, "Information extraction," *Foundations and trends in databases*, vol. 1, no. 3, pp. 261–377, 2008.

[10] J. Mayfield, J. Artiles, and H. Trang Dang, "Text Analysis Conference (TAC) 2012 Proceedings."

[11] M. Banko and O. Etzioni, "The tradeoffs between open and traditional relation extraction," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics ACL '08*, pp. 28–36, Citeseer, 2008.

[12] A. Yates and O. Etzioni, "Unsupervised methods for determining object and relation synonyms on the web," *Journal of Artificial Intelligence Research*, vol. 34, pp. 255–296, 2009.

[13] A. Fader, S. Soderland, and O. Etzioni, "Identifying relations for open information extraction," in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pp. 1535–1545, Association for Computational Linguistics, July 2011.

[14] F. Suchanek, M. Sozio, and G. Weikum, "SOFIE: a self-organizing framework for information extraction," in *Proceedings of the 18th international conference on World wide web*, pp. 631–640, ACM, 2009.

[15] N. Nakashole, M. Theobald, and G. Weikum, "Scalable knowledge harvesting with high precision and high recall," in *Proceedings of the fourth ACM international conference on Web search and data mining - WSDM '11*, (New York, New York, USA), p. 227, Feb. 2011.

[16] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago," in *Proceedings of the 16th international conference on World Wide Web - WWW '07*, (New York, New York, USA), p. 697, ACM Press, May 2007.

[17] Mausam, M. Schmitz, R. Bart, S. Soderland, and O. Etzioni, "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*, pp. 523–534, Association for Computational Linguistics, July 2012.

[18] B. Van Durme and L. Schubert, "Open knowledge extraction through compositional language processing," in *Proceedings of the 2008 Conference on Semantics in Text Processing*, pp. 239–254, Association for Computational Linguistics, Sept. 2008.

[19] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. Hruschka Jr, and T. Mitchell, "Toward an architecture for never-ending language learning," in *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI 2010)*, pp. 1306–1313, 2010.

[20] T. M. Mitchell, "NELL - Never Ending Language Learning," 2013.

[21] F. Suchanek and G. Weikum, "Knowledge harvesting in the big-data era," in *Proceedings of SIGMOD'13*, 2013.

[22] T. Mikolov, I. Sutskever, and K. Chen, "Distributed representations of words and phrases and their compositionality," *Advances in Neural Information Processing Systems (NIPS 2013)*, vol. 26, 2013.

[23] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A Convolutional Neural Network for Modelling Sentences," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Apr. 2014.

[24] E. Grefenstette, P. Blunsom, N. de Freitas, and K. M. Hermann, "A Deep Architecture for Semantic Parsing," in *Proceedings of the ACL Workshop on Semantic Parsing 2014*, Apr. 2014.

[25] D. Gildea and D. Jurafsky, "Automatic labeling of semantic roles," *Computational linguistics*, 2002.

[26] V. Punyakanok, D. Roth, W. Yih, and D. Zimak, "Semantic role labeling via integer linear programming inference," in *Proceedings of the 20th international conference on Computational Linguistics*, pp. 1346–es, Association for Computational Linguistics, 2004.

[27] S. Yih and K. Toutanova, "Automatic semantic role labeling," in *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Tutorial Abstracts on XX*, pp. 309–310, Association for Computational Linguistics, 2006.

[28] K. Hermann and D. Das, "Semantic frame identification with distributed word representations," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pp. 1448–1458, 2014.

[29] K. Litkowski, "Senseval-3 task: Automatic labeling of semantic roles," in *Senseval-3: Third International Workshop on the*, pp. 2–5, 2004.

[30] X. Carreras and L. Màrquez, "Introduction to the CoNLL-2005 shared task: Semantic role labeling," in *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pp. 152–164, Association for Computational Linguistics, June 2005.

[31] J. Ruppenhofer, C. Sporleder, R. Morante, C. Baker, and M. Palmer, "Semeval-2010 task 10: Linking events and their participants in discourse," in *Proceedings of the 5th International Workshop on Semantic Evaluation*, pp. 45–50, Association for Computational Linguistics, 2010.

[32] S. Lim, C. Lee, and D. Ra, "Dependency-based semantic role labeling using sequence labeling with a structural SVM," *Pattern Recognition Letters*, 2013.

[33] D. Croce, G. Castellucci, and E. Bastianelli, "Structured learning for semantic role labeling," *Intelligenza Artificiale*, vol. 6, no. 2, pp. 163–170, 2012.

[34] K. Woodsend and M. Lapata, "Text Rewriting Improves Semantic Role Labeling," *Journal of Artificial Intelligence Research*, vol. 51, pp. 133–164, 2014.

[35] D. Das, M. Kumar, and A. Rudnicky, "Automatic Extraction of Briefing Templates," in *Proceedings of the International Joint Conference on Natural Language Processing IJCNLP '06*, pp. 265–272, 2008.

[36] Y. Shinyama and S. Sekine, "Preemptive information extraction using unrestricted relation discovery," in *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics NAACL/HLT '06*, (Morristown, NJ, USA), pp. 304–311, Association for Computational Linguistics, June 2006.

[37] E. Filatova, V. Hatzivassiloglou, and K. McKeown, "Automatic creation of domain templates," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics COLING/ACL '06*, (Morristown, NJ, USA), pp. 207–214, Association for Computational Linguistics, 2006.

[38] N. Chambers and D. Jurafsky, "Unsupervised learning of narrative schemas and their participants," in *Proceedings of ACL-IJCNLP '09*, (Morristown, NJ, USA), p. 602, 2009.

[39] N. Chambers and D. Jurafsky, "Template-Based Information Extraction without the Templates," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics ACL '11*, pp. 976–986, 2011.

[40] N. Chambers, "Event Schema Induction with a Probabilistic Entity-Driven Model," *Proceedings of the Conference on Empirical Methods on Natural Language Processing EMNLP '13*, pp. 1797–1807, 2013.

[41] L. Qiu, M. Kan, and T. Chua, "Modeling Context in Scenario Template Creation," in *Proceedings of the Third International Joint Conference on Natural Language Processing IJCNLP '08*, pp. 157–164, 2008.

[42] H. A. Santoso, S.-C. Haw, and Z. Abdul-Mehdi, "Ontology extraction from relational database: Concept hierarchy as background knowledge," *Knowledge-Based Systems*, vol. 24, pp. 457–464, Apr. 2011.

[43] X. Kang, D. Li, and S. Wang, "Research on domain ontology in different granulations based on concept lattice," *Knowledge-Based Systems*, vol. 27, pp. 152–161, 2012.

[44] M. Michelson and C. Knoblock, "Constructing reference sets from unstructured, ungrammatical text," *Journal of Artificial Intelligence Research*, vol. 38, no. 1, pp. 189–221, 2010.

[45] K. Radinsky and S. Davidovich, "Learning to predict from textual data," *Journal of Artificial Intelligence Research*, vol. 45, no. 1, pp. 641–684, 2012.

[46] R. Grishman and B. Sundheim, "Message Understanding Conference-6: A Brief History.," in *Proceedings of the International Conference on Computational Linguistics COLING '96*, pp. 466–471, 1996.

[47] D. Croce, C. Giannone, P. Annesi, and R. Basili, "Towards Open-Domain Semantic Role Labeling," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 237–246, Association for Computational Linguistics, July 2010.

[48] J. An, D. Quercia, and J. Crowcroft, "Fragmented social media: a look into selective exposure to political news," in *Proceedings of the 22nd international conference on the World Wide Web WWW2013.*, pp. 51–54, 2013.

[49] S. Maier, "Accuracy matters: A cross-market assessment of newspaper error and credibility," *Journalism & Mass Communication Quarterly*, 2005.

[50] P. Voakes and J. Kapfer, "Diversity in the news: A conceptual and methodological framework," *Journalism & Mass Communication Quarterly*, 1996.

[51] S. Munson and P. Resnick, "Presenting diverse political opinions: how and how much," *Proceedings of the SIGCHI conference on Computer Human Interaction*, 2010.

[52] S. Munson, *Exposure to Political Diversity Online*. PhD thesis, University of Michigan, 2012.

[53] S. Park, S. Lee, and J. Song, "Aspect-level news browsing: Understanding news events from multiple viewpoints," in *Proceedings of IUI'10*, 2010.

[54] S. Park, S. Kang, S. Chung, and J. Song, "A Computational Framework for Media Bias Mitigation," *ACM Transactions on Interactive Intelligent Systems*, vol. 2, pp. 1–32, June 2012.

[55] S. Park, M. Ko, J. Kim, H. Choi, and J. Song, "NewsCube2.0: An Exploratory Design of a Social News Website for Media Bias Mitigation," in *Proceedings of the 2nd International Workshop on Social Recommender Systems*, 2011.

[56] R. Steinberger, B. Pouliquen, and E. V. D. Goot, "An introduction to the europe media monitor family of applications," in *Information Access in a Multilingual World - proceeding of SIGIR 2009*, 2009.

[57] A. Rortais, J. Belyaeva, M. Gemo, E. V. D. Goot, and J. P. Linge, "MedISys: An early-warning system for the detection of (re-) emerging food-and feedborne hazards," *Food Research International*, vol. 43, no. 5, pp. 1553–1556, 2010.

[58] R. Ennals, B. Trushkowsky, and J. Agosta, "Highlighting disputed claims on the web," *Proceedings of the 2010 ACM conference on World Wide Web*, 2010.

[59] J. Zhang, Y. Kawai, and T. Kumamoto, "Extracting Similar and Opposite News Websites Based on Sentiment Analysis," in *Proc. of 2012 International Conference on Industrial and Intelligent Information (ICIII 2012)*, 2012.

[60] K. Leetaru, S. Wang, and G. Cao, "Mapping the global Twitter heartbeat: The geography of Twitter," *First Monday*, 2013.

[61] D. Lenat, "CYC: A large-scale investment in knowledge infrastructure," *Communications of the ACM*, vol. 38, no. 11, pp. 33–38, 1995.

[62] D. Gunning, V. Chaudhri, P. Clark, and K. Barker, "Project Halo Update - Progress Toward Digital Aristotle," *AI Magazine*, 2010.

[63] C. F. Baker, C. J. Fillmore, and J. B. Lowe, "The Berkeley FrameNet Project," in *Proceedings of the 36th annual meeting on Association for Computational Linguistics* -, vol. 1, (Morristown, NJ, USA), p. 86, Association for Computational Linguistics, Aug. 1998.

[64] J. Ruppenhofer, M. Ellsworth, M. Petruck, C. Johnson, and J. Scheffczyk, "FrameNet II: Extended Theory and Practice," 2006.

[65] G. Miller, "WordNet: a lexical database for English," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[66] C. Fellbaum, *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press, 1999.

[67] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: A nucleus for a web of open data," in *The semantic web*, pp. 722–735, Springer, 2007.

[68] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data - SIGMOD '08*, (New York, New York, USA), p. 1247, ACM Press, June 2008.

[69] J. Boyd-Graber and C. Fellbaum, "Adding dense, weighted connections to WordNet," in *Proceedings of the Third International WordNet Conference*, pp. 29–36, 2006.

[70] H. Cunningham, D. Maynard, and K. Bontcheva, *Text Processing with GATE*. University of Sheffield Department of Computer Science, 2011.

[71] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan, "GATE: an Architecture for Development of Robust HLT Applications," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02*, (Morristown, NJ, USA), p. 168, July 2002.

[72] D. Klein and C. Manning, "Accurate unlexicalized parsing," in *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pp. 423–430, Association for Computational Linguistics, 2003.

[73] M.-C. De Marneffe, B. MacCartney, and C. D. Manning, "Generating typed dependency parses from phrase structure parses," in *Proceedings of LREC 2006*, 2006.

[74] D. Cer, M. D. Marneffe, D. Jurafsky, and C. Manning, "Parsing to Stanford Dependencies: Trade-offs between Speed and Accuracy.," *LREC*, 2010.

[75] J. Pasternack and D. Roth, "Extracting article text from the web with maximum subsequence segmentation," *Proceedings of the 18th WWW conference*, 2009.

[76] J. Arias, K. Deschacht, and M. Moens, "Language independent content extraction from web pages," *Proceedings of the 9th Dutch-Belgian information retrieval workshop*, 2009.

[77] C. Kohlschütter, P. Fankhauser, and W. Nejdl, "Boilerplate detection using shallow text features," *Proceedings of WSDM 2010*, 2010.

[78] T. Strohman, D. Metzler, H. Turtle, and W. Croft, "Indri: A language model-based search engine for complex queries," *Proceedings of the International Conference on Intelligent Analysis*, vol. 2, no. 6, pp. 2—-6, 2005.

[79] J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. P. L. F. de Carvalho, and J. a. Gama, "Data Stream Clustering: A Survey," *ACM Comput. Surv.*, vol. 46, pp. 13:1—-13:31, July 2013.

[80] J. Azzopardi and C. Staff, "Incremental Clustering of News Reports," *Algorithms*, vol. 5, no. 3, pp. 364–378, 2012.

[81] A. Muhic, J. Rupnik, and P. Skraba, "Cross-lingual document similarity," in *Proceedings of the 34th International Conference on Information Technology Interfaces (ITI2012)*, (Cavtat, Dubrovnik), pp. 387–392, IEEE, 2012.

[82] T. Stajner, D. Rusu, L. Dali, B. Fortuna, D. Mladenić, and M. Grobelnik, "A service oriented framework for natural language text enrichment," *Informatica (Ljubljana)*, vol. 34, pp. 307–313, Oct. 2010.

[83] T. Štajner, D. Rusu, L. Dali, and B. Fortuna, "Enrycher: service oriented text enrichment," in *Proceedings of SiKDD*, 2009.

[84] M. Grobelnik and D. Mladenić, "Simple classification into large topic ontology of web documents," *Journal of Computing and Information Technology*, vol. 13, no. 4, pp. 279–285, 2004.

[85] M. McCandless, "Accuracy and performance of Google's Compact Language Detector (CLD)," 2011.

[86] T. M. Mitchell, J. Betteridge, A. Carlson, E. Hruschka, and R. Wang, "Populating the Semantic Web by Macro-Reading Internet Text," in *Proceedings of ISWC 2009* (A. Bernstein, D. R. Karger, T. Heath, L. Feigenbaum, D. Maynard, E. Motta, and K. Thirunarayan, eds.), vol. 5823 of *Lecture Notes in Computer Science*, Springer, 2009.

[87] M. V. I. Greaves, "An Introduction to Project Halo," 2010.

[88] P. Haley, "Background for our Semantic Technology 2013 presentation (part 1)," 2013.

[89] D. V. I. Gunning, "HaloBook and Progress Towards Digital Aristotle," 2011.

[90] M.-C. de Marneffe and C. D. Manning, "Stanford typed dependencies manual," tech. rep., Stanford, CA, 2013.

[91] D. McCarthy, R. Koeling, J. Weeds, and J. Carroll, "Finding predominant word senses in untagged text," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics ACL '04*, pp. 280–287, 2004.

[92] L. Dali and B. Fortuna, "Triplet Extraction from Sentences using SVM," in *Proceedings of SiKDD 2008*, 2008.

[93] M. Palmer, D. Gildea, and P. Kingsbury, "The proposition bank: An annotated corpus of semantic roles," *Computational Linguistics*, vol. 31, no. 1, pp. 71–106, 2005.

[94] K. Erk and S. Pado, "Shalmaneser–a toolchain for shallow semantic parsing," in *Proceedings of LREC*, vol. 6, Citeseer, 2006.

[95] E. Charniak and M. Johnson, "Coarse-to-fine n-best parsing and MaxEnt discriminative reranking," in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pp. 173–180, Association for Computational Linguistics, 2005.

[96] M. Collins, *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania, 1999.

[97] K. Toutanova and S. W.-t. Yih, "Automatic Semantic Role Labeling - Tutorial," tech. rep., Microsoft Research, 2007.

[98] K. Hacioglu and W. Ward, "Target word detection and semantic role chunking using support vector machines," *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology companion volume of the Proceedings of HLT-NAACL 2003 - short papers*, pp. 25–27, 2003.

[99] J. Curtis, J. Cabral, and D. Baxter, "On the Application of the Cyc Ontology to Word Sense Disambiguation.," in *19th International FLAIRS Conference*, (Melbourne Beach, FL), 2006.

[100] R. Navigli, "Word sense disambiguation: A survey," *ACM Computing Surveys (CSUR)*, vol. 41, no. 2, pp. 10:1–10:69, 2009.

[101] A. Kilgarriff, "How dominant is the commonest sense of a word?," *Text, Speech and Dialogue*, vol. LNCS 3206, pp. 103–111, 2004.

[102] C. D. Manning and A. Y. Ng, "Exploring the Utility of ResearchCyc for Reasoning from Natural Language," tech. rep., Leland Junior Stanford University, 2006.

[103] X. Yan and J. Han, "gspan: Graph-based substructure pattern mining," in *Proceedings of ICDM 2003*, 2003.

[104] S. Nijssen and J. Kok, "A quickstart in frequent structure mining can make a difference," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 647–652, ACM, 2004.

[105] Y. Chi, Y. Yang, and R. Muntz, "HybridTreeMiner: An efficient algorithm for mining frequent rooted trees and free trees using canonical forms," in *Proceedings of 16th International Conference on Scientific and Statistical Database Management*, pp. 11–20, 2004.

[106] A. P. R. Kohut, "Pew surveys of audience habits suggest perilous future for news," 2013.

[107] J. P. R. Enda, "In print, newspapers cut opinion," 2013.

[108] A. Haghighi and L. Vanderwende, "Exploring Content Models for Multi-Document Summarization," in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, (Boulder, Colorado), Association for Computational Linguistics, 2009.

[109] E. Alfonseca and M. Trampus, "Diversified News Service - RENDER project deliverable D5.2.2," tech. rep., RENDER consortium, 2012.

[110] B. Fortuna, M. Grobelnik, and D. Mladenic, "Visualization of text document corpus," *INFORMATICA-LJUBLJANA-*, vol. 29, no. 4, p. 497, 2005.

[111] T. Štajner, I. Novalija, and D. Mladenić, "Informal Multilingual Multi-domain Sentiment Analysis," *Informatica*, vol. 37, pp. 373–380, 2013.

[112] A. Balahur, R. Steinberger, M. A. Kabadjov, V. Zavarella, E. Van Der Goot, M. Halkia, B. Pouliquen, and J. Belyaeva, "Sentiment Analysis in the News.," in *LREC*, 2010.

[113] S. Baccianella, A. Esuli, and F. Sebastiani, "SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining.," *LREC*, 2010.

[114] D. Marcu, "Improving summarization through rhetorical parsing tuning," in *Proceedings of The 6th Workshop on Very Large Corpora*, pp. 206–215, 1998.

[115] D. Pighin and E. Alfonseca, "Evaluation of the Diversified News Service - RENDER project deliverable D5.2.4," Tech. Rep. September 2013, RENDER consortium, 2013.

[116] R. A. Fisher, *Statistical Methods For Research Workers*. Cosmo study guides, Cosmo Publications, 1925.

[117] F. Huang and A. Yates, "Open-domain semantic role labeling by modeling word spans," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 968–978, Association for Computational Linguistics, July 2010.

[118] S. Carpenter, "A study of content diversity in online citizen journalism and online newspaper articles," *New Media & Society*, 2010.

[119] P. Jakopin and A. Bizjak, "O strojno podprtem oblikoslovnem označevanju slovenskega besedila," *Slavistična revija*, vol. 45, no. 3-4, pp. 513–531, 1997.

[120] S. Džeroski, T. Erjavec, and N. Ledinek, "Towards a Slovene dependency treebank," in *Proceedings of The Fifth Slovenian and First International Language Technologies Conference*, 2006.

[121] A. Chanev, "Studying the Learning Curves of a Statistical Dependency Parser for Four Languages," in *Proceedings of The Fifth Slovenian and First International Language Technologies Conference*, 2006.

[122] D. Lin, "Dependency-based evaluation of MINIPAR," *Treebanks*, 2003.

[123] I. Hendrickx, G. Bouma, and F. Coppens, "A Coreference Corpus and Resolution System for Dutch.," *LREC*, 2008.

[124] J. Hajič and M. Ciaramita, "The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages," in *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009)*, 2009.

[125] A. Björkelund, L. Hafdell, and P. Nugues, "Multilingual semantic role labeling," in *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009)*, 2009.

[126] S. Nijssen and J. Kok, "Frequent Subgraph Miners: Runtimes Don't Say Everything," *Proceedings of the Int'l Workshop on Mining ...*, 2006.

[127] T. Stajner, B. Thomee, A.-M. Popescu, and A. Jaimes, "Automatic selection of social media responses to news," in *ACM WSDM 2013*, 2013.

Appendix A

Datasets

A.1 Domain Templates

All the data used for evaluation of domain templating code is available online at `http://mitjat.com/research/topic_templates_data.zip`. The package includes the following:

- Clear-text and metadata (title, url, topic/domain) of all input documents used in our algorithms (see Section 4.4.1). Comprised of 2000 nontopical and 4000 topical documents across 5 topics.
- A sample TaskRabbit instructions page (see Section 4.4.2.1) that can be used for reproducing the golden standard or expanding to new topics.
- The output of TaskRabbit workers and the actual golden questions used in our evaluation.
- A sample CrowdFlower task and instructions (see Section 4.4.2.2) that can be used for reevaluation of our results or evaluating new algorithms.
- A `README.txt` describing the contents in more detail.

A.2 NewsFeed Data

The NewsFeed stream is available on a case by case basis, strictly for research purposes. Due to copyright concerns, we cannot grant access to the general public. The NewsFeed continues to be maintained by the Artificial Intelligence Laboratory at Jozef Stefan Institute. If interested in the data, visit the NewsFeed homepage at `http://newsfeed.ijs.si` to find up-to-date contact information.

Dodatek B

Razširjen povzetek v slovenščini

B.1 Uvod

Pisana beseda je v zgodovini človeštva igrala pomembno vlogo: je trajen, razmeroma enolično razumljiv sistem prenosa idej in znanj, ki nam omogoča vsakovrstne napredke. Z iznajdbo tiskalnega stroja, še toliko bolj pa z razmahom interneta, je količina napisanega besedila postala neobvladljiva za posameznega človeka[1]. Težavo skušamo danes militi med drugim s pomočjo računalnikov, natančneje z metodami s področja tekstovnega rudarjenja. Te metode znajo oceniti podobnost dveh besedil, izluščiti iz besedila ključne podatke, sestaviti povzetek besedila, oceniti njegov sentiment ali temo, prevajati med jeziki, iskati dokumente po ključnih besedah, in še marsikaj.

Zgodovinsko večina teh metod temelji na *sintaktičnih značilkah* – grobo poenostavljeno, uporabljajo statistike o tem, ali se določene besede pojavijo v besedilu, kolikokrat si sledijo določeni pari besed, kolikokrat se v besedilu pojavijo različne končnice besed ipd. V zadnjih letih pa so bile veliko pozornosti deležne tudi *semantične metode*. Namesto z besedami operirajo z logičnimi *koncepti*; ti so predstavljeni z enoznačnimi oznakami, ki jih lahko uporabimo znova in znova, na različnih problemih. Na primer, "pes", "psu", "kuža" in "Hund" so na sintaktičnem nivoju popolnoma ločene besede, v semantični predstavitvi pa vse dobijo isto oznako, saj predstavljajo isti koncept. Prednost tega je, da lahko različni problemi in njihove rešitve med seboj delijo znanje, uporabijo pa lahko tudi t.i. predznanje (background knowledge), ki je bilo zbrano z izrecnim namenom pomagati raznovrstnim semantičnim metodam. Na primer, če delamo avtomatski povzetek besedila, ki vsebuje stavek "Pes Fik je tehtal 70 kilogramov," želimo, da računalnik ta stavek vključi v povzetek, ker opisuje nekaj nenavadnega – a dejstvo, da sedemdesetkilogramski psi niso nekaj vsakdanjega, je računalniku neznano, razen če ima dostop do predznanja v obliki baze podatkov o psih. Da lahko takšno bazo uporabi, pa mora biti sposoben asociirati besedo "pes" iz besedila z ustreznimi podatki v bazi; to je mnogo lažje, če tako baza kot besedilo psa označita z enotno, semantično oznako.

[1] `http://what-if.xkcd.com/76/`

Čeprav je bilo na področju semantičnih metod narejenega že veliko, ostaja precej smeri še vedno neraziskanih. Izkaže se, da je *semantizacija* besedil, t.j. pretvorba iz sintaktične v semantično obliko, vse prej kot enostavna, in veliko raziskav poteka prav na tem področju. Kot opišemo v razdelku 2.2 (Related Work), se raziskovalci v glavnem omejijo na izluščanje posameznih konceptov ali relacij, npr. vseh oseb ali vseh parov podjetje—direktor. Manjše število drznejših projektov (npr. NELL [20]) poskuša izluščiti čim večje število entitet in parov, s ciljem, da bi zgradili bazo univerzalnega predznanja, na primer "stol je tip pohištva" ali "pes je žival". Takšnemu predznanju pravimo tudi "zdrava pamet" (common sense).

Projekti, ki gradijo takšno predznanje, se osredotočijo na natančno izločanje dejstev iz ogromnih količin teksta, dejstva pa nato združijo, da izločijo šumne podatke. Ker pregledajo velike količine teksta, si lahko privoščijo, da je priklic na nivoju dokumentov *nizek*: iz marsikakšnega dokumenta npr. ne izluščijo sploh nobenega dejstva. V pričujoči disertaciji se za razliko od teh projektov osredotočimo na metode, ki pri semantizaciji besedil dosežejo *večji priklic* za ceno manjše natančnosti ali manj strogo strukturiranih izhodnih podatkov. Ker je gostota tako pridobljenih semantičnih oznak mnogo večja, upamo, da bodo tvorile tako informativno predstavitev posameznih dokumentov ali celo stavkov, da se bomo z njihovo pomočjo lahko lotili manj makroskopskih nalog kot je grajenje "zdrave pameti".

Natančnost naših metod preverimo najprej intrinzično (pomerimo natančnost in priklic), bolj obetavno od metod pa še ekstrinzično: rezultat semantizacije besedila s to metodo uporabimo kot osnovo za rešitev dveh nalog v tekstovnem rudarjenju; eno, ki išče, kaj je množici dokumentov *skupnega*, in eno, ki išče, v čem se sorodni dokumenti *razlikujejo*.

Disertacija je razdeljena na sledeča poglavja:

- **1. poglavje** predstavi raziskovalno področje in poda motivacijo in uvod.
- **2. poglavje** opiše obstoječe delo in rezultate na področju ter orodja in podatkovne zbirke, ki se jih poslužimo v našem delu. Opiše tudi sistem za trajno zajemanje novic z interneta, ki smo ga razvili in ki priskrbi podatke za večino analiz v preostalih poglavjih. Ključne dele tega poglavja v slovenščini povzamemo sproti, ko vpeljemo posamezne koncepte v razdelku B.2.
- **3. poglavje** predstavi in primerja dve metodi za semantizacijo besedila. V slovenščini to poglavje povzamemo v razdelku B.2.
- **4. poglavje** vpelje in ovrednosti dve metodi za grajenje domenskih predlog, temelječi na semantiziranem tekstu. V slovenščini to poglavje povzamemo v razdelku B.3.
- **5. poglavje** predstavi sistem za odkrivanje različnih stališč in pogledov na medijske dogodke iz spletnih novic, prav tako temelječ na semantičnih podatkih. V slovenščini to poglavje povzamemo v razdelku B.4.

- **6. poglavje** na podlagi prejšnjih poglavij povzame prednosti in slabosti uporabe semantičnih reprezentacij.

B.2 Semantizacija besedil

Zapis semantike (t.j. pomena) danega besedila v formalno obliko je težka naloga že za ljudi, še toliko bolj pa za avtomatizirane sisteme. Trenutni tehnologiji do tja manjka še precej, zato se osredotočimo le na semantizacijo bistvenih delov besedila. Model, ki ga povzamemo v tej disertaciji, temelji na *semantičnih okvirjih*. Semantični okvir je, grobo rečeno, akcija, opisana z nekaj vnaprej predpisanimi lastnostmi. Stavek navadno vsebuje enega ali več okvirjev. Na primer: stavek "Med včerajšnjim sprehodom po mestu, je Mojca v odsevu izložbenega okna opazila šolmoštra." porodi naslednje okvirje:

Zaznavanje	
Zaznavalec	"Mojca"
Zaznavani	"učitelj"
Lokacija	"mesto"
Čas	"včeraj"

Hoja	
Hodeči	"Mojca"
Lokacija	"mesto"
Čas	"včeraj"

Opazimo lahko, da s pretvorbo besedila v okvirje izgubimo nekaj informacije, hkrati pa preostali informaciji podamo mnogo strožjo strukturo, kot jo je imela prej. Tudi t.i. *polnilci vrzeli* v okvirju (mesto, učitelj, ...) niso več besede, pobrane neposredno iz besedila, temveč koncepti iz vnaprej definirane *ontologije*. Naloga gradnje takšnih okvirov se imenuje *označevanje semantičnih vlog* (Semantic Role Labeling, SRL. Več o SRL povemo v razdelku 3.3.1.

V disertaciji se problema gradnje okvirjev lotimo na dva načina. Oglejmo si ju.

Preslikane oznake semantičnih vlog (Mapped Sem. Role Labels, MSRL). Referenčna baza semantičnih okvirjev je FrameNet. Njegovi glavni pomanjkljivosti sta, da primanjkuje učnih podatkov in da

Najprej smo zgradili sistem za SRL na osnovi FrameNeta [64], referenčne baze semantičnih okvirjev (in pri tem krepko sledili obstoječim raziskavam), nato pa smo se lotili polavtomatske gradnje povezav med strukturami FrameNeta in relacijami Cyca [61]. FrameNetova pomanjkljivost je namreč ta, da uporablja oznake, izolirane od vseh ostalih ontologij in baz predznanja (npr. Cyc, WordNet[66] ...). Ko označimo besedilo s FrameNetom, te oznake same po sebi torej še niso zelo uporabne. Cyc pa je splošnonamenska ontologija z bogatim naborom predznanja, a šibko navezavo na angleški jezik, kar je ravno komplementarno FrameNetu. Končni rezultat ekstrakcije s tako zgrajenim sistemom so logične trditve v Cycovem formatu; glej sliko B.1.

Detajli so v razdelku 3.3.

Poenostavljene odvisnostne razčlembe (Simplified Dependency Parses, SDP). Čeprav FrameNetovi okvirji zajamejo samo bistvene atribute dogodkov, so še vedno precej ekspresivni. Pri pristopu SDP smo se jih odločili poenostaviti do te mere, da imajo vsi enak nabor atributov: akter (npr. Zaznavalec v našem okviru *Zaznavanje*), tarča (npr. Zaznani), čas, kraj in orodje. Ta poenostavitev odpravi potrebo po individualnih učnih podatkih za vsako akcijo in nam omogoči mnogo učinkovitejšo ekstrakcijo okvirjev, pa tudi njihovo predstavitev s "poljubno" ontologijo; izbrali smo WordNet [66]. Wordnet je, grobo rečeno, hierarhija konceptov, urejenih po relaciji nadpomenka-podpomenka.

Za ekstrakcijo se poslužimo odvisnostnega razčlenjevalnika s Stanforda [73]. Njegov izhod poenostavimo s sistemom ročno sestavljenih pravil, da dobimo čim več od zgoraj naštetih atributov. Vrednosti atributov nato povežemo z WordNetom s preprosto "first sense" hevristiko za disambiguacijo večpomenskih besed. Izhod je je torej statična struktura, napolnjena s koncepti iz WordNeta. Primer najdemo na sliki B.1. Za kompaktnejši prikaz okvirje rišemo grafovsko: koren drevesa je ime/akcija okvirja, listi pa so atributi.

Detajli so v razdelku 3.2.

Rezultati Če merimo uspeh s številom pravilno zapolnjenih posameznih vrzeli v semantičnem okvirju, doseže SDP mero $F_1 = 0.47$, MSRL pa $F_1 = 0.17$. Nepričakovano slabo obnašanje MSRL je večinoma posledica (pre)dolgega cevovoda (v katerem vsaka faza doprinese nekaj napake) ter pomanjkljivih leksikalnih informacij v Cycu, zaradi katerih je z razumnim vložkom časa in denarja težko doseči dobro poravnavo s FrameNetom ali naravnim jezikom.

Končni rezultat tega poglavja so torej okvirji, pridobljeni po metodi SDP. Vrednosti atributov ter imena okvirjev so koncepti iz WordNeta, tako da lahko za vsak okvir trivialno izpeljemo tudi njegove splošnejše variante tako, da nadomestimo enega ali več atributov z njegovimi nadpomenkami.

Rezultate natančneje poredstavimo v tabeli 3.1 in analiziramo v razdelku 3.4.

B.3 Grajenje domenskih predlog

V 4. poglavju se na osnovi pravkar prikazanih orodij za semantizacijo lotimo problema *gradnje domenskih predlog* (domain template construction). Cilj je zgraditi sistem, ki na vhod prejme množico dokumentov iz iste domene (npr. novice o bombnih napadih), nato pa avtomatsko generira atribute, ki jih lahko pripišemo večini dokumentov iz vhodne množice (npr. število smrtnih žrtev; število ranjenih; kraj napada; organizacija, ki je privzela odgovornost). Za te atribute želimo najti tudi tipološki opis (npr. "kraj napada" je vedno tipa "geografska lokacija").

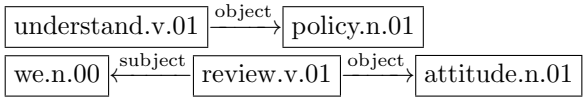
Predstavimo dva nova algoritma za reševanje tega problema; oba sta osnovana na semantični reprezentaciji vhodnih besedil.

"To understand and appreciate the Bush administration's policy regarding
Israeli Prime Minister Sharon's disengagement plan, we must briefly reexamine the record."

MSRL:

```
(#$objectImproved #$Comprehending* #$OrganizationPolicy*)
(#$performedBy #$Comprehending* (#$ObjectDenotedByFn "we")*)
(#$evaluationInput #$Evaluating* #$OrganizationPolicy*)
(#$performedBy #$ExercisingAuthoritativeControlOverSomething*
      (#$ObjectDenotedByFn "we")*)
(#$performedBy #$PurposefulAction* (#$ObjectDenotedByFn "Sharon")*)
```

SDP:



Slika B.1: Dejanski primer izpisa metod SDP in MSRL na vzorčnem stavku.

Metoda FGS (pogosti posplošeni podgrafi – Frequent Generalized Subgraphs) Vhodno besedilo pretvorimo v semantične okvirje z metodo SDP, kot smo opisali v razdelku B.2. Okvirje nato še dodatno poenostavimo: zavržemo vse atribute razen akterja, tarče in akcije oz. imena okvirja. Te predstavimo kot trojice oblike akter $\xrightarrow{\text{akcija}}$ tarča. Na primer, stavek "Včeraj je Marko jedel golaž" postane Marko $\xrightarrow{\text{jesti}}$ golaž.

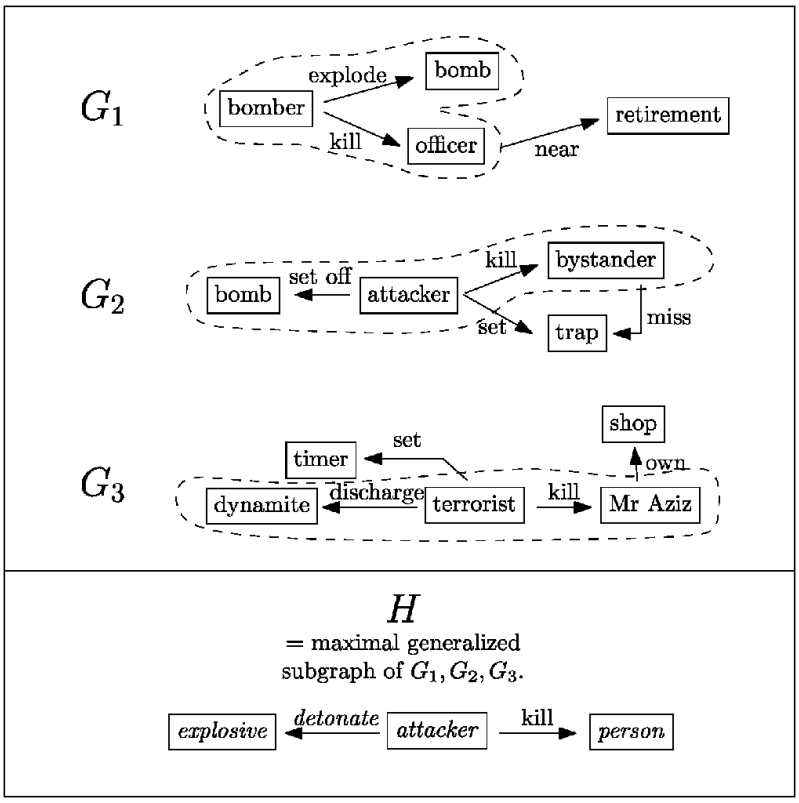
Takšne trojice nato povežemo v graf – akterji in tarče so vozlišča, akcije pa (označene) povezave. Za vsak vhodni dokument konstruiramo po en graf. Ključna intuicija, na kateri temelji metoda FGS, je zdaj tale: v večini dokumentov bo podana večina ključnih atributov zgodbe (število žrtev, kraj napada itd.), in med seboj bodo povezani s semantično podobnimi relacijami. V grafovski terminologiji to pomeni, da pričakujemo, da se v večini grafov ponovijo vozlišča enakega tipa[2], povezana z enako označenimi povezavami. Še bolj formalno: iščemo tak (majhen) graf, katerega oznake so nadpomenke, njegove *specializacije* (t.j. kopije tega grafa, kjer so oznake vozlišč morda nadomeščene s podpomenkami) pa se pojavijo v vhodnih grafih. Ta mali graf predstavlja "bistvo" oz. iskano predlogo vhodne domene. Idejo ilustriramo na sliki B.2

Implementacija takšne hevristike ni preprosta, ker je kombinatorični prostor vseh podgrafov vhodnih dokumentov ter njihovih posplošitev velik. Problem rešimo v grobem tako, da oznake vhodnih grafov najprej posplošimo (t.j., vsako vozlišče predstavimo z njegovim kar se da splošnim tipom). Nato v posplošenih grafih poiščemo ponavljajoče se podgrafe; tudi to trivialno, vendar izvedljivo; priredili smo algoritem Nijssena in sodelavcev [126]. Kot zadnji korak najdene podgrafe spet specializiramo,

[2]A verjetno ne s čisto enakimi oznakami. Na primer, pri vhodnih dokumentih o bombnih napadih bo en graf vseboval vozlišče z oznako Kabul, en Karači, en pa New York. To so različna oznake, vendar imajo skupno *nadpomenko* oziroma tip, v tem primeru "mesto".

kolikor se da, da pri tem njihove oznak še vedno ostanejo nadpomenke oznak vhodnih grafov.

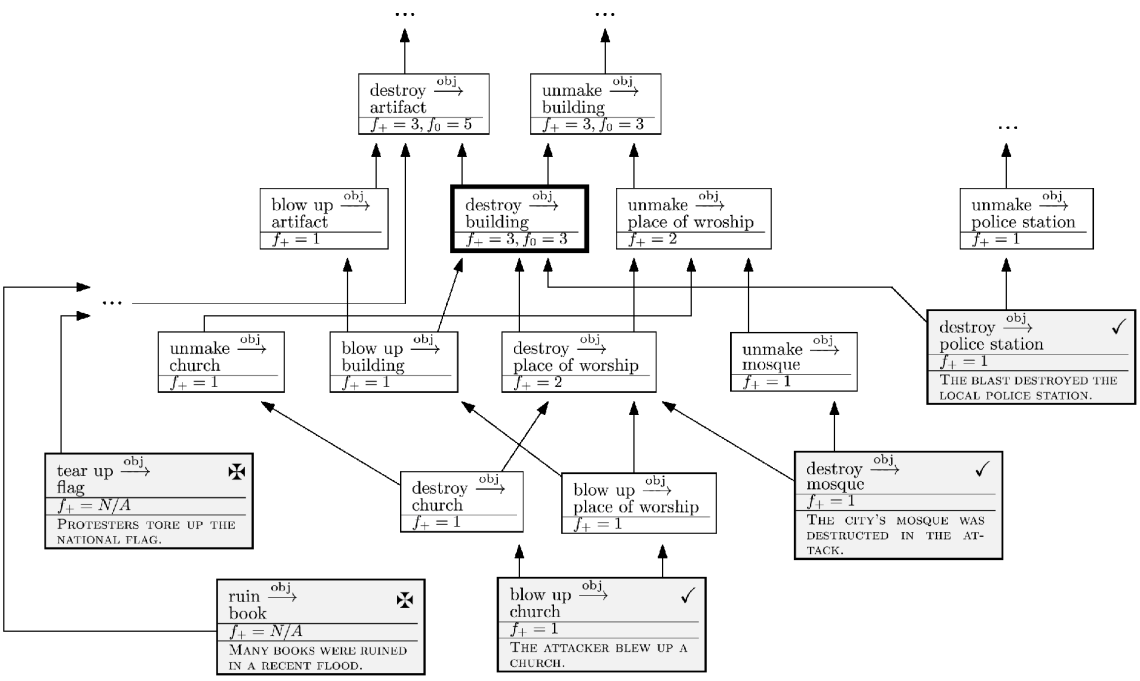
Algoritem je podrobneje opisan v razdelku 4.2.



Slika B.2: Primer delovanja metode FGS na umetno ustvarjenem, zelo majhnem primeru. Grafi $G_{1...3}$ predstavljajo vhodne dokumente, graf H pa iskano predlogo. Vidimo, da vsako vozlišče v H ustreza po enemu vozlišču v $G_{1...3}$; npr., "*attacker*" ustreza vozliščem "bomber" v G_1, "attacker" v G_2 in "terrorist" v G_3. Ta vozlišča so med seboj v vseh vhodnih grafih povezana na enak način. Predlogo H na koncu interpretiramo takole: v vsaki vhodni zgodbi se pojavi entiteta tipa "*attacker*", ki detonira ("*detonate*") eksploziv ("*explosive*") in ubije ("*kill*") entiteto tipa oseba ("person"). Entitete na sliki so vzete iz WordNeta in zato predstavljene z originalnimi angleškimi oznakami.

Metoda CT (karakteristične trojice – Characteristic Triplets) Tudi ta metoda semantične okvire najprej predstavi s trojicami, a nekoliko drugače: akcija okvirja in vsak od njegovih atributov porodita po eno trojico. Uporabimo isi primer kot prej: stavek "Včeraj je Marko jedel golaž" postane [jesti] $\xrightarrow{\text{akter}}$ [Marko], [jesti] $\xrightarrow{\text{tarča}}$ [golaž], [jesti] $\xrightarrow{\text{čas}}$ [včeraj].

Metoda na vhodu pričakuje dokumente iz domene (npr. novice o bombnih napadih), pa tudi negativne primere, t.j. dokumente, ki ne spadajo v domeno. Vse



Slika B.3: Primer usmerjenega acikličnega grafa, kot ga skonstruira metoda CT. Vsak okvirček prikazuje trojico in število njenih pojavitev v domenskih dokumentih f_+. Domena v tem primeru so bombni napadi. Sivi okvirčki predstavljajo trojice, ki se v dokumentih pojavljajo neposredno, ter stavke, ki so jih porodili; znak ✓predstavlja domenske, znak ✠ pa izvendomenske dokumente. Puščice kažejo od manj do bolj specializiranih trojic. Odebeljeni okvirček je tista trojica, ki bo izbrana v končno domensko predlogo. Vrednosti okvirov na sliki niso prikazane, vendar korelirajo s f_+.

trojice iz vseh vhodnih dokumentov nato povežemo v usmerjen acikličen graf: trojice so točke, dve trojici pa sta povezani, če so vsi elementi prve trojice nadpomenke elementov druge trojice (ali pa so si med seboj enaki). Primer takega grafa je na sliki B.3. Za vsako trojico si nato zapomnimo število njenih pojavitev v domenskih in izvendomenskih dokumentih. Števila pojavitev propagiramo po grafu navzgor (t.j. proti nadpomenkam) in za vsako trojico izračunamo njeno *vrednost*; ta pozitivno korelira s številom pojavitev v domenskih dokumentih in negativno s številom pojavitev v izvendomenskih dokumentih. Trojice z najvišjo vrednostjo na koncu proglasimo za domensko predlogo.

Intuitivno se bodo preveč specializirane trojice (npr. $\boxed{\text{eksplozija}} \xrightarrow{\text{akter}} \boxed{\text{avtobomba}}$) pojavile v premajhnem številu vhodnih dokumentov, zato bo njihova vrednost nizka; presplošne (npr. $\boxed{\text{učinkovanje}} \xrightarrow{\text{akter}} \boxed{\text{fizičen objekt}}$) bodo nastopale v izvendomenskih dokumentih, kar bo spet škodovali njihovi vrednosti; tiste ravno prav splošne, ki si jih želimo v domenski predlogi (npr. $\boxed{\text{poškodovanje}} \xrightarrow{\text{akter}} \boxed{\text{eksplozivno sredstvo}}$)), pa bodo uravnovesile oba faktorja, ki nastopata v formuli za vrednost.

Rezultati Da bi ocenili kvaliteto obeh metod, smo za pet domen (novice o avionskih nesrečah, bombnih napadih, potresih, diplomatskih obiskih, ter kazenskih obsodbah) najprej razvili zlato evaluacijsko množico, t.j. nabor desetih atributov, ki naj jih ima idealna predloga za posamezno domeno. Nato smo algoritem pognali na nekaj sto dokumentih iz vsake domene. Rezultati so v tabeli B.1

Domain	FVM	FGS	CT
airplane	0.53	0.24	**0.57**
bomb	**0.52**	0.26	0.44
earthquake	0.38	0.50	**0.54**
visit	—	0.15	**0.30**
sentence	—	0.15	**0.59**

Tabela B.1: Recall@20, t.j. delež atributov iz zlate evaluacijske množice, vsebovanih tudi v izhodu algoritmov. Primerjamo se tudi s sodobno konkurenčno metodo FVM [37].

Čeprav je variacija precejšnja, je razvidno, da algoritem FGS vsaj ni slabši od konkurenčne FVM. Pri tem FGS za izhodne atribute predpiše natančen tip – kvalitativna prednost, ki iz tabele zgoraj ni očitna. Algoritem CT se obnese slabše predvsem na račun večje količine zavrženih informacij pri pretvorbi semantičnih okvirov v graf, ter premočne predpostavke o strukturni regularnosti vhodnih grafov.

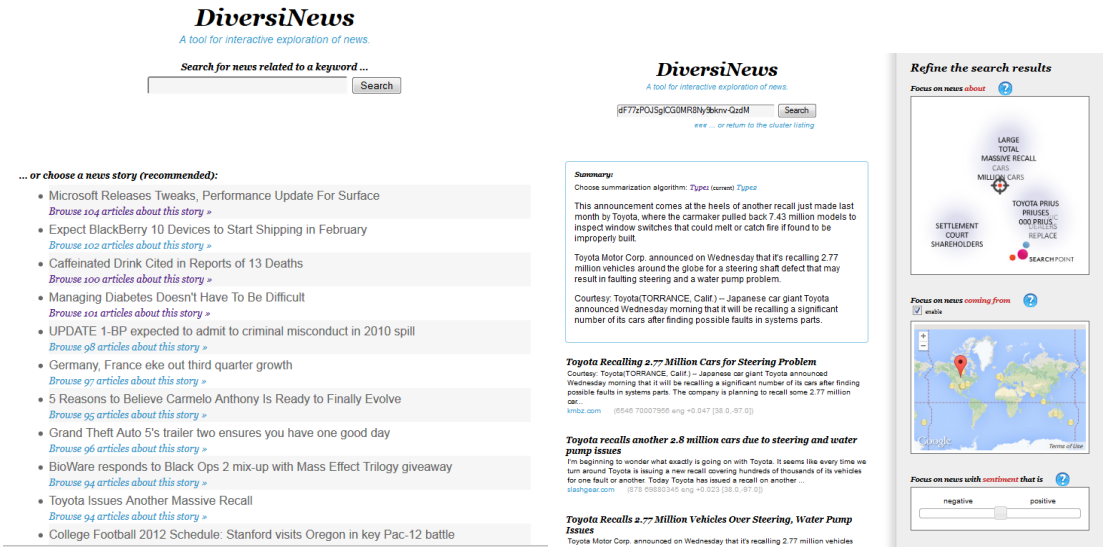
Evaluacijsko množico in natančno metodologijo primerjave izhodnih podatkov algoritma z evaluacijsko množico smo javno objavili (glej prilogo A), s čimer spodbujamo hitrejši razvoj in transparentnejšo primerjavo algoritmov na tem področju v bodoče.

B.4 Izpostavljanje raznolikosti mnenj

V 5. poglavju semantične tehnike uporabimo za iskanje ključnih lastnosti, v katerih se dokumenti med seboj *razlikujejo*. Problema se lotimo z utilitarnega vidika; zasnujemo sistem za zajem in analizo novic ter pripadajoč uporabniški vmesnik, namenjen raziskovanju in lažjemu razumevanju le-teh. DiversiNews, kot smo poimenovali vmesnik, za popularne novice, o katerih poroča veliko število virov, omogoča vpogled v razlike med njihovimi stališči. Takšno orodje je lahko v veliko pomoč takrat, ko želimo razumeti novico v detajle: na primer zato, ker smo medijski strokovnjak (novinar), ali na primer zato, ker novice poročajo o podjetju, v katerega smo veliko investirali, ali pa na primer zato, ker zgodba govori o intrigi v svetu profesionalne košarke in nas košarka preprosto zanima.

Orodje najlažje predstavimo kar s sliko B.4. Na uvodnem zaslonu vidimo seznam nedavnih popularnih zgodb. Na ta del se ne osredotočamo, saj gre za ločen problem *odkrivanja* novic, ne pa njihovega razumevanja. Ko zgodbo izberemo, se prikaže glavni zaslon (slika B.4b). Na levi so našteti vsi članki o zgodbi in na vrhu njihov

povzetek, na desni pa uporabnik lahko izbere, kakšne članke želi izpostaviti. Izbiro lahko opravi s tremi krmilniki, t.j. na podlagi treh lastnosti člankov. Od vrha navzdol so to: izpostavljena (pod)tematika (ko izide nov MacBook, se bodo eni članki osredotočili na dizajn, drugi pa na tehnične značilnosti), lokacija založnika (Američani bodo o bližnjevzhodni krizi pisali drugače kot Iranci) in naklonjenost (angl. *sentiment*; o pomembnem Janševem nagovoru bodo eni pisali pohvalno, drugi grajajoče). Vsakič, ko uporabnik spremeni položaj kakšnega od krmilnikov, se članki na levi v hipu preuredijo tako, da so na vrhu prikazani najrelevantnejši. A člankov je še vedno veliko, zato istočasno tvorimo tudi nov povzetek, ki zdaj v treh stavkih povzema samo relevantne članke.



a) Uvodni zaslon b) Glavni zaslon

Slika B.4: Zaslonske slike programa DiversiNews.

Tehnično ozadje. DiversiNews sintetizira veliko količino podatkov in jih prikaže na obvladljiv način na eni sami strani. Podatki povzemajo informacijo o člankih, ki iz tekstovne reprezentacije ni razvidna, zato jih pojmujemo kot semantične, vendar gre za drugačne tipe semantičnih podatkov kot v prejšnjih dveh poglavjih.

Za analizo naklonjenosti uporabimo Štajnerjevo [127] metodo in implementacijo. Podatke o geografski lokaciji založnikov dobimo z uporabo predznanja o založnikih, dosegljivega na internetu. Razbitje člankov na (pod)tematike dobimo s hierarhičnim grozdenjem, krmilnik pa nato grozde na zaslonu prikaže s pomočjo MDS (multidimensional scaling [110]). Rangiranje člankov ob spremembi krmilnikov temelji na preprosti linearni kombinaciji relevantnostnih mer, ki jih podajo posamezni krmilniki.

Ponovno pa semantične okvirje, razvite v razdelku B.2, uporabimo za inovativno sprotno generiranje povzetkov relevantnih člankov. Razvili smo algoritem za

sumarizacijo (razdelek 5.2.7), ki deluje v prostoru semantičnih okvirov: najprej z uporabo predznanja za vsak par okvirov določi številsko mero podobnosti, nato pa v povzetek požrešno enega za drugim jemlje okvirje, ki so čim bolj podobni vsem še neuporabljenim okvirjem in čim manj podobni vsem že uporabljenim.

Rezultati. Sistem smo evalvirali z dveh stališč: 1) pravilno delovanje in intuitivnost uporabniškega vmesnika, ter 2) uporabnost sistema pri raziskovanju novic. Posebej smo evalvirali tudi sumarizator. Rezultati so spodbudni – velika večina uporabnikov je nedvoumno potrdila uporabnost sistema kot celote. Največ izboljšav potreben je po mnenju uporabnikov možnih pri krmliniku podtematik. Ker gre za neobičajen krmilnik, ki predstavlja veliko količino podatkov, je zahtevna že sama navigacija, nato pa se izkaže, da se tudi generirani povzetki najslabše odzivajo prav na spremembe tega krmilnika. V pozitivnem smislu pa so uporabniki izpostavili predvsem uporabnost geografskega krmilnika ter preglednost, ki jo prinaša sistem.

Evaluacija sumarizatorja pokaže, da se ta odreže primerljivo s konkurenčnim sodobnim sumarizatorjem na osnovi sintaktičnih značilk, vendar se slabše odziva na spremembe krmilnika. Podrobnosti vseh evaluacij najdete v poglavju 5.3

B.5 Zaključek

Ogledali smo si dve različni metodi za semantizacijo teksta in uporabo semantičnih značilk v praksi. Glede na ekstrinzično evaluacijo v poglavjih 4 in 5 lahko zaključimo, da se je s semantičnimi metodami moč približati sintaktičnim in jih morda ponekod celo preseči. Videli smo tudi, da predznanje lahko močno pomaga algoritmom, saj na primer algoritma SDP (razdelek B.3) in FrameSum (razdelek B.4) dosežeta dostojne rezultate kljub temu, da v izhodišču zavržeta veliko večino vhodnih podatkov in ohranita le izluščene semantične okvirje – ki pa jih nato povežeta s predznanjem.

Hkrati smo videli, da imajo semantični pristopi še vedno vrsto težav. Te so očitne predvsem pri začetni semantizaciji besedila, kjer je kvaliteta očitno še daleč od idealne. Pokažejo pa se tudi pasti in pomanjkljivosti semantičnih reprezentacij med uporabo v poglavjih 4 in 5.

Glavne opažene **prednosti** semantičnih metod so:

- *Avtomatska izbira značilk.* Semantizacija obdrži le "najbolj informativne" dele besedila. Semantične reprezentacije so zato pogosto kompaktnejše, s čimer potencialno zmanjšamo šum in poenostavimo nadaljnje procesiranje (npr. ocenjevanje primernosti fragmenta za domenske predloge ali ocenjevanje podobnosti dveh stavkov za potrebe povzemanja)

- *Normalizacija naravnega jezika.* Semantizacija nas reši skrbi s skloni, spregatvami, časi, sopomenkami ipd.; skratka, inteligentno skrči prostor značilk.

- *Dostop do predznanja.* Ker za koncepte uporabljamo ustaljene oznake, brez dodatnega truda dobimo dostop do potencialno zelo dragocenih rezultatov

drugih raziskovalcev, npr. taksonomije nadpomenk v WordNetu.

Precej pa je tudi **pomanjkljivosti in presenečenj**:

- *Krhkost in seštevajoče se napake.* Cevovodi za semantizacijo besedil sestojijo iz mnogo *zaporednih* faz, in njihove napake se akumulirajo. Sorodna težava je, da vsaka od faz deluje pod določenimi predpostavkami, in če so predpostavke le ene od teh faz kršene, lahko kakovost celotnega cevovoda močno trpi. Izpostavimo lahko npr. predpostavko o "pravilnosti" jezika, ki jo naredi razčlenjevalnik – na podatkih s Twitterja bi metode verjetno delovale mnogo slabše.

- *Računska zahtevnost.* Predvsem razčlenjevanje besedila je računsko dolgotrajno. Sicer ga je trivialno paralelizirati, a iz logističnih in finančnih razlogov je količina teksta, ki ga lahko v praksi obdelamo, vseeno omejena.

- *Omejen priklic.* Strojno branje v polnem pomenu besede je še daleč; trenutno se moramo pri izluščanju osredotočiti le na posamezne dele besedila (tiste, ki jih zajamejo semantični okvirji) in upati, da s tem nismo zavrgli preveč informacij. V povprečju to kolikor toliko drži, zelo lahko pa najdemo primere, kjer naši (in sorodni) postopki zavržejo tudi bistvene informacije.

- *Potrebno predznanje.* V tej disertaciji smo uporabili številne vire, npr. WordNet, FrameNet, Cyc, skladenjski razčlenjevalniki ipd. Če želimo metode prenesti na besedila z drugačnimi jezikovnimi lastnostmi (npr. v drugem jeziku, ali pa celo samo v drugačnem stilu ali z drugačnimi domenskimi poudarki), potrebujemo nove ali prilagojene vire, ki pa morda za naš jezik ali domeno sploh še ne obstajajo. Gradnja teh virov je izredno zamudna in draga.

- *Zahtevnost implementacije.* Čeprav zahtevnost implementacije ne vpliva na končni rezultat, je v pragmatičnem smislu pomembna. Obvladovanje in povezovanje velikega števila orodij v delujoč cevovod je zamudnejše kot uporaba konceptualno preprostejših metod. Situacija se na srečo izboljšuje, saj popularnost tehnik globokega procesiranja narašča, s tem pa tudi število dosegljivih elegantnih orodij in meta-paketov.

Glede na vse zgoraj našteto smo mnenja, da semantične metode pri procesiranju besedil prinesejo v povprečju le majhne napredke, zato za široko rabo še niso zrele. Hkrati pa smo videli, da tudi niso slabše od bolj sintaktično naravnanih metod – vsaj na nekaterih področjih že zdaj dosegajo ali presegajo njihove rezultate. Tako lahko zaenkrat semantične metode priporočimo za probleme, ki so jim v dobršni meri pisani na kožo. Pomembno pa se je zavedati, je opisu "na kožo pisan" ustreza vsako leto več problemov, saj se na področju semantičnega procesiranja (razčlenjevanje, ekstrakcija informacij, itd.) veliko dogaja, in orodja so sposobna kvalitetno semantizirati vedno večji in raznolikejši delež informacij z vedno večjo natančnostjo na vedno širšem naboru besedil.

B.5.1 Uporabnost metod za druge jezike

Pomembna prednost semantičnih reprezentacij je, da temeljijo na *konceptih*, in ti so po definiciji neodvisni od jezika. Ko imamo besedilo enkrat predstavljeno v semantični obliki, je vseeno, kako je bilo originalno zapisano – na primer, celotno 4. in 5. poglavje te disertacije načeloma ne potrebuje nobene spremembe, če bi semantični okvirji izhajali iz slovenskega besedila.

Problematični del pa je seveda semantizacija, pretvorba iz teksta v enotno semantično obliko. Naše metode (SDP in MSRL, razdelek B.2), pa tudi metode drugih raziskovalcev, se za to zanašajo na vrsto orodij in podatkovnih baz. Sem spadajo slovarji, razčlenjevalniki, označevalniki skladenjskih vlog itd. Kot pokažemo na nekaj primerih v razdelku 6.2.2, so ti viri na voljo tudi za številne neangleške jezike, vendar večinoma zaostajajo za angleškimi po kvaliteti in obsegu.

Poglejmo konkreten primer: metodo SDP, ki jo uporabljamo v tej disertaciji, in slovenščino. Na voljo so vsa potrebna orodja: označevalnik skladenjskih vlog [119], semantični razčlenjevalnik [120], in slovarji, s katerimi lahko slovenske besede približno preslikamo na WordNet. Metode iz pričujoče disertacije bi torej lahko uporabili tudi na slovenskih tekstih. Vendar pa bi za to potrebovali več dela (orodja niso enako dobro podprta in enako zrela kot angleška), predvsem pa bi na vsakem od korakov delali večje napake, kot jih delamo pri angleščini. Vprašljivo je, ali bi bil končni rezultat še uporaben. Podobno velja za druge jezike; večji kot je jezik, bolje je podprt in lažje bi se bilo približati rezultatom na angleščini.

B.5.2 Izvirni prispevki znanosti

Osrednji izvirni prispevki znanosti so sledeči:

- *Metodi za semantizacijo besedil.* Predstavimo dve novi metodi (SDP in MSRL) za semantizacijo besedil, ki naredita bistveno drugačen kompromis med priklicem in natančnostjo kot obstoječe metode. Evalviramo ju intrinzično in ekstrinzično.

- *Metodi za konstrukcijo domenskih predlog.* Predstavimo dve novi metodi za konstrukcijo domenskih predlog in smo prvi, ki raziščemo in opišemo, kako se pri reševanju tega problema obnese uporaba semantičnih značilk. Metoda CT je po kvaliteti vsaj primerljiva z obstoječim stanjem tehnike ("state of the art"), dodatno pa za polja predlog proizvede podrobne tipološke omejitve, česar obstoječe metode niso sposobne.

- *Formalna evaluacija in testni podatki za konstrukcijo domenskih predlog.* Evaluacija metod s tega področja je težavna, in doslej ni bilo na voljo nobene jasno dokumentirane metodologije za evaluacijo ali javnih evaluacijskih podatkov, s čimer področje težje napreduje. Tu ponudimo oboje.

- *Izpostavljanje raznolikih mnenj.* Predstavimo integriran, samozadosten sistem za zajem, procesiranje, agregacijo in brskanje novic ter odkrivanje mnenj v

njih. Z združitvijo podatkov različnih modalnosti (geografski, tematski, in o naklonjenosti) uporabnikom omogočimo bistveno drugačen vpogled v izbrane problematike, kot ga omogočajo obstoječa orodja, z eksplicitnejšim in enostanejšim dostopom do raznolikih mnenj.