UNIVERSITÀ DEGLI STUDI
DI TRENTO

DEPARTMENT OF INFORMATION ENGINEERING AND COMPUTER SCIENCE
ICT International Doctoral School

CONTROLLING THE EFFECT OF CROWD NOISY ANNOTATIONS IN NLP TASKS

Azad Abad

Supervisor
Prof. Alessandro Moschitti
Università degli Studi di Trento

April 2017

Acknowledgment

In the beginning, I practiced the same old story of doing my PhD with the blind eyes and without thinking of why I do need PhD and how this journey is going to change my life.
For the first year of my PhD, the research activities did not go very smoothly and I thought about quitting it every weekend. I was completely frustrated and almost abandoned my PhD till I got an opportunity to work with Prof. Alessandro Moschitti. I thank him very much not only because he is a great supervisor but also a good friend. Definitely, I could not find my path without his support and mentoring during the last 4 years. (I started to like doing research because of you. Thank you very much Alessandro for those simple life lessons you used to share with me.)
University of Trento, provided me with all the possible help and support under the management of Prof. Nicu Sebe. I would like to thank him as well as two great PhD office secretaries, Andrea and Francesca for their wonderful accountability.
I would like to thank my friends and colleagues specially Moin, Arindam, Evgeny and Orkan for fulfilling my life and giving me professional advises and critical comments. I would like to thank my family and my wife Saameh who always encourage me to stay and fight also giving me both support and freedom.
Now, the journey is over and what I can see behind is not regret but the footsteps of wisdom and expertise. Eventually, learning how to learn can make me stronger and a better person to serve my family and the society in the future.

Abstract

Natural Language Processing (NLP) is a sub-field of Artificial Intelligence and Linguistics, with the aim of studying problems in the automatic generation and understanding of natural language. It involves identifying and exploiting linguistic rules and variation with code to translate unstructured language data into information with a schema.
Empirical methods in NLP employ machine learning techniques to automatically extract linguistic knowledge from big textual data instead of hard-coding the necessary knowledge. Such intelligent machines require input data to be prepared in such a way that the computer can more easily find patterns and inferences. This is feasible by adding relevant metadata to a dataset. Any metadata tag used to mark up elements of the dataset is called an *annotation* over the input.
In order for the algorithms to learn efficiently and effectively, the annotation done on the data must be accurate, and relevant to the task the machine is being asked to perform.
In other words, the supervised machine learning methods intrinsically can not handle the inaccurate and noisy annotations and the performance of the learners have a high correlation with the quality of the input data labels. Hence, the annotations have to be prepared by experts.
However, collecting labels for large dataset is impractical to perform by a small group of qualified experts or when the experts are unavailable. This is special crucial for the recent deep learning methods which the algorithms are starving for big supervised data.

Crowdsourcing has emerged as a new paradigm for obtaining labels for training machine learning models inexpensively and for high level of data volume. The rationale behind this concept is to harness the “wisdom of the crowd” where groups of people pool their abilities to show collective intelligence.
Although crowdsourcing is cheap and fast but collecting high quality data from the non-expert crowd requires careful attention to the task quality control management. The quality control process consists of selection of appropriately qualified workers, providing a clear instruction or training that are understandable to non-experts and performing sanitation on the

results to reduce the noise in annotations or eliminate low quality workers.

This thesis is dedicated to control the effect of crowd noisy annotations use for training the machine learning models in variety of natural language processing tasks namely: relation extraction, question answering and recognizing textual entailment.

The first part of the thesis deals with design a benchmark for evaluation Distant Supervision (DS) for relation extraction task. We propose a baseline which involves training a simple yet accurate one-vs-all strategy using SVM classifier. Moreover, we exploit automatic feature extraction technique using convolutional tree kernels and study several example filtering techniques for improving the quality of the DS output.

In the second part, we focused on the problem of the crowd noisy annotations in training two important NLP tasks, i.e., question answering and recognizing textual entailment. We propose two learning methods to handle the noisy labels by (i) taking into account the disagreement between crowd annotators as well as their skills for weighting instances in learning algorithms; and (ii) learning an automatic label selection model based on combining annotators characteristic and the task syntactic structure representation as features in a joint manner.

Finally, we observe that in fine-grained tasks like relation extraction where the annotators need to have some deeper expertise, training the crowd workers has more impact on the results than simply filter-out the low quality crowd workers. Training crowd workers often requires high-quality labeled data (namely, gold standard) to provide the instruction and feedback to the crowd workers.

We conversely, introduce a self-training strategy for crowd workers where the training examples are automatically selected via a classifier. Our study shows that even without using any gold standard, we still can train workers which open doors toward inexpensive crowd training procedure for different NLP tasks.

Keywords

[Crowdsourcing; Distant Supervison; Relation Extraction; Question Answering; Recognizing Textual Entailment; Crowd Self Training; Human Noisy Annotations; NLP; Kernel Methods]

Contents

List of Tables

List of Figures

1

Introduction

1.1 Motivations

Modern society push for processing large volume of information everyday in order to stay up-to-date. There is a change continuously not only in the amount of information, but also in its type over time. As a consequence of the sheer volume and heterogeneous nature of the information it is becoming impossible to analyze this data manually. So, the automatic techniques are being used to facilitate this process.

Recently, the big revolution is how artificial intelligence and cognitive systems are advancing and changing our world. Machine Learning (ML) plays a key role in a wide range of critical applications, such as data mining, natural language processing, visual recognition, and expert systems. ML provides potential solutions to all these domains and more, and is set to be a pillar of our future civilization.

One of the most outstanding results in developing modern Natural Language Processing (NLP) systems was achieved using statistical machine learning methods (Machine-driven) [85]. Prior attempts to deal with language-processing tasks were extremely labour intensive typically requiring to hand-engineer a large set of manual rules (Human-driven).

However, the rule-based systems suffer from several drawbacks such as difficulty to find experts in order to define the rules, challenge in developing, testing and modifying the rules and finally the generalization problem [76].

The most successful ML algorithms used in NLP are supervised methods, which learn to extract knowledge from labeled training data. These methods are heavily used in NLP applications for voice recognition, categorization, sentiment analysis and machine translation [38].

The supervised methods require metadata known as *tag*, *labels* or *class*. Each instance in the training set is a pair consisting of an input object, e.g., a document, a sentence, or a word, and a desired output label, e.g., document category, sentence class or word type. In the training process, machine learning algorithm is telling the computer what patterns are important, and providing examples and counter-examples for each distinction the model should make. The goal is to infer a decision function by analysing the training data, which can be used for mapping unseen test examples to their output labels.

One of the biggest challenges in NLP community is to provide the tagged data for training the supervised methods. Labels for data are often obtained by asking experts to make judgments about a given piece of unlabeled data and creating these labels manually is often too slow, static and expensive. In small datasets with few thousand of examples, a group of experts can handle the annotation task. However, it is impractical for big dataset with millions of the examples. So, manually labeling large training sets is not a feasible option.

In the other side, with the growth of the Internet, Mobile in IOT era, the amount of available data has greatly increased. The new machine learning approaches (a.k.a. Deep Learning) showing promising results in various NLP tasks are big data hungry as well [21, 62, 22, 33, 136, 135].

In addition, It has been shown that there is direct relation between the performance of ML models and the amount of the data [99]. Figure 1.1 shows the relation between the mentioned factors (namely, data size

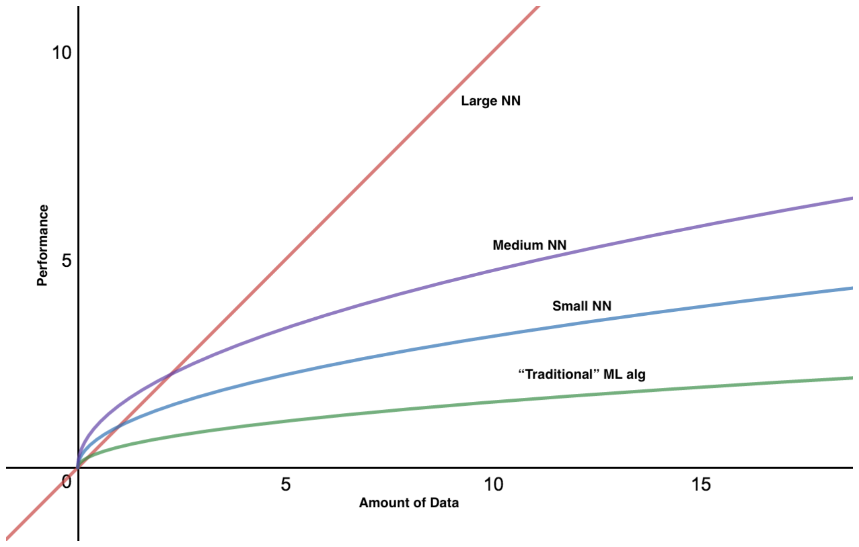


Figure 1.1: The performance of ML methods vs the amount of data

and performance). Thus, achieving the state of the art in NLP tasks without considering the new ML methods and the big data phenomena is not possible.

Following the trend, there are many active areas of research in machine learning that are aimed to assign the label to large unlabeled data to build better and more accurate models for NLP tasks. For instance, Weak-supervised learning attempts to leverage various heuristics and existing knowledgebases (KB)/databases as weak supervision to label the unlabeled data [50, 111, 140]. This gave rise to the terms “weak” or “distant” to indicate that the labels on the training data are imprecise or noisy. The method is attractive because it does not require human supervision; it is efficient for even tera-scale extraction; and it is domain and language independent. However, heuristically labeled training data in DS approach suffers from two major drawbacks:

- KBs are usually incomplete because they are manually constructed, and it is not possible to cover all human knowledge. It causes false negative errors in training, meaning that the DS approach, assigns labels to instances as negative classes due to lack of related knowledge in KB.

- The matched entities from KB can not represent the sentence semantic truly. It causes false positive errors and the semantic drift.

Although, some techniques have been used to decrease the effect of noisy labels in classifier output, this does not necessarily impact the learning aspect of the classifier. The classifier assumes that the labels are precise and often fails to optimize solving two problems jointly (i.e., data trustability and learning the model).

As an alternative solution, crowdsourcing has emerged as a new paradigm to gather text annotations quickly at scale by considering the "widsom of crowd" as the source of supervision. According to Wikipedians, crowdsourcing is "the act of taking a task traditionally performed by an employee or contractor, and outsourcing it to an undefined, generally large group of people, in the form of an open call." [1]

Crowdsourcing is valuable to computational linguists as a source of labeled training data to use in machine learning. Tagging and organizing all different types of big data, be it unstructured, structured or semi structured, require big effort and it can be achieved efficiently by leveraging the power of crowd [134]. Specially, it is an effective approach in collection annotations for NLP tasks because the annotation task requires some degree of common linguistic knowledge which is beyond the power of DS methods. While crowdsourcing offers solicitors of information or services nearly unlimited cheap (or free), the challenge lies in aggregating the multiple, noisy contributor inputs to create a consistent corpus. The output of crowdsourcing, especially in the case of micro-tasks and when monetary incentives are involved, often suffers from low quality. Moreover, the crowd workers are usually not experts and they are of different age, education and ethnics.

For exmaple, in a survey conducted by Ipeirotis [64], information about the demographics of participants on a well-known online crowdsourcing platform (a.k.a, Amazon Mechanical Turk), together with information

[1] https://en.wikipedia.org/wiki/Crowdsourcing

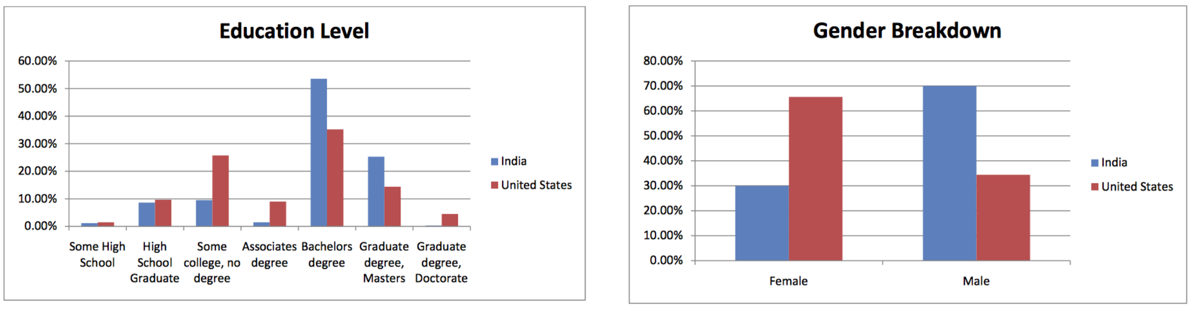


Figure 1.2: Demographic information of Turkers in AMT: right) Gender and left) Education level

about their level of activity and motivation for working on Amazon Mechanical Turk has collected. He focused on the countries with the most number of crowd workers participating in the posted tasks namely: US and India. It shows a high level of diversity among the crowd workers in terms of age, gender and education level which makes the annotations less accurate. Figure 1.2 illustrates the gender and education level of the crowd workers for the both countries .

Broadly speaking, there are two main approaches to improve the quality of the annotations in crowdsourcing tasks. First, pre-processing methods consist of crowd demographic filtering [71], inserting gold questions [166, 14] and preparing a skill modeling to determine workers' proficiency [133]. Second, post-processing methods such as statistical processing to identify outliers and exclude them, both at the level of contributors and contributions. Majority vote is the simplest of such statistical measures or weighting the votes of annotators based on their overall performance [134, 165, 61]. However, the research has shown that it is not the best solution to create high quality annotations [110]. Thus, the more advanced techniques are required to filter out the human noise from crowdsourcing NLP tasks by considering the latent additional information of crowd workers (a.k.a metadata) that can reveal the performance of the workers in desired task.

Then, several interesting questions immediately arise in how to optimally utilize annotations by considering such metadata of crowd workers. For example, how ML techniques handle differences between workers in terms of the quality of annotations they provide. How we can re-weight the noisy annotations for the end task of creating a model. How it is possible to identify genuinely ambiguous examples via annotator disagreements. How we estimate the trustability of crowd workers by considering background and expertise as an additional data. And, what is the contribution of the user data to reveal the pattern of annotation noise in learning NLP tasks is.

The work presented in this thesis is a step towards finding the answers of the above questions.

1.2 Thesis Contributions

The contributions of this thesis are at the intersection of the fields of human-computer interactions, machine learning, and NLP. One overarching goal of this work is to get the advantage of computational approaches in order to control the effect of noisy annotation in NLP tasks when the source of noise is human. The contributions are listed as follow:

1. We propose an approach based on convolution Tree Kernels (TK) and Support Vector Machines (SVMs) to be used as a baseline for applying Distant Supervision in Relation Extraction Task (Chapter 4).
2. We design several instance weighting strategies to help the learning algorithm dealing with the noisy labels of training examples, thus learning a better model in Question Answering (QA) task (Chapter 5).
3. We aim to learn a statistical model to predict the best label from a set of aggregated annotations from a group of annotators and by quantifying their skills. We exploit the contribution of these collected metadata to be used as descriptor features to generate labels to train QA and RTE tasks (Chapter 6).

4. We propose a new framework to improve the accuracy of relation extractor by involving human in the loop. So, we introduce an iterative human-machine collaborative learning method for crowdsourcing. As a first contribution, we introduce a self-training strategy for crowdsourcing. Then, we introduce our iterative human-machine co-training framework for the task of RE (Chapter 7).

1.3 Structure of the Thesis

The thesis outline is as follow:

Chapter 1 - Introduction - This chapter describes the research plan of the thesis. It presents the research approach, theoretical and practical motivations. It also highlights contributions of the thesis, and provides an outline for the rest of the content to follow.

Chapter 2 - Background Work and Concepts - This chapter provides a general overview on supervised learning approaches and Support Vector Machines (SVMs) in particular. Specifically, we focused on classification and re-ranking problems and provide a brief description on how kernels method can solve the nonlinear problems at a reasonable computational cost. In addition, we present the relevant background and the tasks definition in the field of NLP (namely: Relation Extraction, Question Answering and Recognizing Textual Entailment), targeted in this thesis. In addition, a description of the used corpora for each task is elaborated in details. Finally, we explain the standard evaluation metrics that are widely used to evaluate our proposed methods.

Chapter 3 - Annotation Collection for Supervised Methods - In this chapter, we introduce crowdsorucing concept as a cheap solution for collecting the annotations for NLP tasks. We also provide a literature review on various statistical approaches on quality control for crowdsourcing tasks. Then, a whole section is dedicated to present our problem statements, our proposed methods and the contributions briefly.

Chapter 4 - Distant Supervision for Relation Extraction - In this chapter we describe our methodology for creating a baseline for relation extraction task applying distant supervision. The experimental setup and the results are presented and discussed extensively.

Chapter 5 - Instance Weighting for Crowdsourcing QA Task - This chapter provides the description of our solution for decreasing the effect of the noisy labels in question answering task. We also present the detail of our experiments and the result comparison with majority voting-based as baseline.

Chapter 6 - Automatic Label Selector (ALS) - We detailed our proposed classifier to predict the best label from a set of annotations done by a group of annotators considering their profile as a feature set. Moreover, our comprehensive analysis on the experiment results for applying the prediction of ALS classifier to train QA and RTE tasks are visualized and presented.

Chapter 7- Autonomous Crowd Training for Relation Extraction - In this chapter we characterize our iterative human-machine co-training framework used to train crowd workers for relation extraction task. Then, our experimental results is presented and compared with the both methods, using distant supervision or just crowdsourcing without worker training.

Chapter 8 - Conclusion - Finally, the conclusion and future work arising from these studies is discussed in this chapter.

1.4 List of Publications

- *Creating a Standard for Evaluating Distant Supervision for Relation Extraction.* Azad Abad, Alessandro Moschitti. CLIC 2014 - The First Italian Conference on Computational Linguistics, Pisa, Italy, 9-10 December 2014
- *A Multi-task Learning Framework for Time-continuous Emotion Estimation from Crowd Annotations.* Mojtaba Khomam Abadi, Azad Abad, Ramanathan Subramanian, Negar Rostamzadeh, Elisa Ricci, Jagannadan Varadarajan, Nice Sebe. CrowdMM 2014 - International ACM Workshop on Crowdsourcing for Multimedia, Florida, USA, 07 November 2014
- *Distant Supervision for Relation Extraction Using Tree Kernels.* Azad Abad, Alessandro Moschitti. IIR 2014 - 6th Italian Information Retrieval Workshop, Cagliari, Italy, 25-26 May 2015
- *Taking the Best from the Crowd: Learning Question Passage Classification from Noisy Data.* Azad Abad, Alessandro Moschitti. *SEM 2016 - 5th Joint Conference on Lexical and Computational Semantics, Berlin, Germany, 11-12 August 2016
- *Appetitoso: A Search Engine for Restaurant Retrieval based on Dishes.* Gianni Barlacchi, Azad Abad, Emanuele Rossinelli, Alessandro Moschitti. CLiC-IT 2016 - Third Italian Conference on Computational Linguistics, Napoli, Italy, 5-6 December 2016
- *Learning to Automatically Select the Best Labels from Crowd Annotators.* Azad Abad, Alessandro Moschitti. TACL 2017 - to be submitted to Transactions of the Association for Computational Linguistics
- *Self-Crowdsourcing Training for Relation Extraction.* Azad Abad, Moin Nabi, Alessandro Moschitti. ACL 2017 - The 55th annual meeting of the Association for Computational Linguistics,Vancouver, Canada, 30 July-4 August 2017

2

Background Work and Concepts

In this chapter we introduce some important concepts and definitions which the reader of the thesis should be familiar with before proceeding to the rest of the thesis.

First, we briefly introduce the concept of supervised learning and formally define the classification problem with some examples of its application in Natural Language Processing (NLP) tasks related to this thesis.

Then, we formalize the Support Vector Machine (SVM) classifier and the optimization problem, which are central to the work described in this thesis. We show how Tree Kernels (TK) can be used to kernelize the SVM classifier to allow the learning of non-linear decisions in our tasks.

Finally, we introduce the targeted NLP tasks (i.e., Relation Extraction, Question Answering and Recognizing Textual Entailment) which are crowdsourced.

Later on, we describe the used corpora, and the related works for the above mentioned tasks separately. At the end of the chapter, a section is dedicated to introduce the most used evaluation metrics in the thesis.

2.1 Supervised Methods

In this section we provide an overview of some of the concepts and definitions used in supervised learning methods. They consist of a) the decision function and b) the inference problem to assign labels to output variables.

The goal of supervised learning is to learn a decision function $h \in \mathcal{H}$ from some hypothesis space $\mathcal{H}$. This means that given a label training set $D = \{(x_i, y_i)\}_{i=1}^{n}$, the decision function h assigns an input $x \in \mathcal{X}$ to an output label $y \in \mathcal{Y} : h : \mathcal{X} \to \mathcal{Y}$.

Inference: The label assignment to the output variable can be formalized as follows:

$$h(x) = \underset{y \in \mathcal{Y}}{argmax} f_w(x, y), \tag{2.1}$$

where $f : X \times Y \to R$ is a discriminant function parametrized by a vector of weights w that assigns a numerical score to the input-output variables. It can also introduce as a compatibility function which measures the degree of agreement between the input and the output variables. In common practice, function f is a linear function in the vector of model parameters w:

$$f_w(x) = \langle x, \Psi(x, y) \rangle \tag{2.2}$$

Where $\Psi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$ estimates a joint feature representation and $\langle ., . \rangle$ computes the dot product. This estimation is feasible by mapping the input example pairs (x, y) into some feature space $\mathbb{R}^d$.

2.1.1 Supervised Learning in NLP Tasks

In this section we provide a brief overview of the most common strategies used to solve NLP problems - namely a) classification and b) re-ranking. Then, we provide a definition of the specific tasks which are targeted directly in this thesis in separate sections. Here, the main goal is not to provide a final solution for the considered problems but to describe the nature of the problems and trying to categorize them into corresponding classes of machine learning problems. Our solutions to tackle the noisy labels in the targeted tasks are elaborated in details in Chapters 4, 5, 6 and 7.

Classification

Treating NLP tasks as a classification problem is a very well known practice performed by the community whether it is the case of a simple document classification problem or a more complex task like learning entailed texts in Recognizing Textual Entailment.

Problem Formulation The output domain of binary classification is $\mathcal{Y} = \{-1, +1\}$ whereas in multi-class classification $\mathcal{Y} = \{1, .., K\}$, the number of classes are extended to k. The inference problem can be solved by using an extensive list of output variables y and choosing the one that results in the maximum score. The joint feature map for binary classification can be defined as $\Psi(x) = y\Psi(x)$ where $\Psi : \mathcal{X} \to \mathbb{R}^d$ denotes an optional feature set, representing the input variables to the classifier.

In Multi-class $\Psi(x) = \Lambda^k \otimes \Phi(x)$, Λ^k refers to the binary encoding of the k-th label y and $\otimes$ represents the matrix product which forms all the products between the two argument vectors. However, it is possible to simplify the formulae for the binary and multi-class classifications into $max(0, 1 - yf(w, x))$ and $max(0, 1 + \underset{y \neq y^*}{max} f(w_y, x) - f(w_{y^*}, x))$ respectively.

Hence, in order to learn accurate models that produce decision functions with high discriminative powers, the main effort should be dedicated to how well to represent the structure of the input data (a.k.a., feature engineering).

Re-Ranking

Here we describe the problem of reranking text pairs which is heavily used in various NLP tasks like Questions Answering and micro blog retrieval, among others. Several effective solutions exists to the reranking problem, however, we focus on a very successful approach that represents query-document pairs to the ranking algorithm to learn an accurate model.

Problem Formulation We are provided with a set of retrieved enumerations of query-documents, where each query $q_i \in Q$ consists of a list of candidate answers $D_i = \{d_{i1}, d_{i2}, ..., d_{in}\}$. The candidates set is labeled with their relevancy judgments $\{y_{i1}, y_{i2}, .., y_{in}\}$ in a way that the relevant answers are labeled with $y_{ij} = 1$ while 0 otherwise. The aim is to build a model that for each query q_i and its associated candidate passages D_i, generates an optimal ranking $R = (r_{i1}, r_{i2}, .., r_{in})$ for which the relevant answers/documents appears on top of the list.

Formally, the task is to learn a ranking function in the general form:

$$h(w, \Psi(q_i, D_i)) \rightarrow R, \tag{2.3}$$

where $\Psi(.)$ maps the input query-document pairs into appropriate feature representations. Ultimately, the weight vector w is a model parameter vector which is updated during the learning process.

Learning to Rank Approaches Basically, there are three major approaches to learn the function h namely: pointwise, pairwise and listwise methods.

Pointwise approach. This is one of the simplest methods to solve the reranking problem. In this method the ranking problem can be reformulated into a binary classification task where each triple (q_i, d_{ij}, y_{ij}) represents a training instance and the decision function can be presented as: $h(w, \Psi(q_i, d_{ij})) \rightarrow y_{ij}$.

The decision function $h(.)$ simply computes a dot product between the model weight w and the feature generated by $\Psi(.)$ as described in section 2.1.1.

Pairwise approach. In this approach the model is trained to score the correct pairs higher than the incorrect pairs within a certain margin:

$$\langle w, \Psi(q_i, d_{ij}) \rangle \geq \langle w, \Psi(q_i, d_{ik}) \rangle + \epsilon, \tag{2.4}$$

where the document d_{ij} is a relevant answer for query q_i while d_{ik} is not. This method requires more training data compared to the pointwise approach which leads to lower training times.

Listwise approach. The third method treats a query as a list with its associated document candidates to be represented to the ranking function as a single training instance. This method is able to capture considerably more information about the ground truth ordering of the input candidates. However, the inference process is expensive in terms of time complexity.

2.1.2 Support Vector Machines

In this section we introduce the binary Support Vector Machine (SVM) classifier that can handle the classification and the ranking problems. This classifier is heavily employed in this thesis. We also describe syntactic parsers that are used to construct syntactic structures for automatically generating rich features. The generated feature sets can be used in the SVM tree kernel framework for the both classification and ranking problems.

Support Vector Machines (SVM) is one of the most popular classification algorithms among supervised learning methods. It belongs to the family of generalized linear classifiers that enable the use of high-dimension feature spaces via the kernel trick. In the following, we formalize the SVM optimization problem and describe how it can be kernelized to allow for learning non-linear decision boundaries.

Theoretically, the binary SVM deals with the output space that is $y_i \in \{+1, -1\}$. Consider a binary classification task with the training data $\{x_i, y_i\}_{i=1}^{N}, \vec{x} \in \mathbb{R}^d$. SVMs find a hyperplane that correctly separate the training data of two classes while maximizing the distance of the both classes from the hyperplane (marginal distance). Formally, SVMs estimate the following linear discriminant function:

$$f_w(x) = \langle w, x \rangle \,, \tag{2.5}$$

where $w \in \mathbb{R}$ is a vector of model parameters. SVM estimates the weight vector w by solving the following optimization problem:

$$\underset{w}{minimize} \frac{1}{2} \|w\|^2 + C \sum max(0, 1 - y_i \langle w, x_i \rangle) \tag{2.6}$$

where the first part of the the objective function is a regularizer encoding the maximum margin principle and the second part represents the empirical loss obtained on the training set. The margin trade-off parameter C determines the trade-off between maximizing the margin and the number of training data points within that margin. The Eq. (2.6) can reformulated as the following constrained optimization problem:

$$\begin{aligned} &\underset{w,\xi}{minimize} \frac{1}{2} \|w\|^2 + C \sum_i \xi_i \\ &\text{subject to } \underset{\xi_i \geq 0}{y_i} = \langle w, x_i \rangle \geq 1 - \xi_i \end{aligned} \tag{2.7}$$

where the slack variable ξ allows some data points to lie within the margin.

The Kernel Trick One of the most important properties of SVMs is that it can create a non-linear decision boundary by projecting the data through a non-linear function ϕ to a space d' with a higher dimension $\phi(x) : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$:

$$f_\alpha(x) = \sum_i \alpha_i y_i \langle \phi(x_i), \phi(x) \rangle \tag{2.8}$$

Since the outcome of the decision function only relies on the dot-product of the vectors in the feature space d, it is not essential to explicitly project to that space. So, the symmetry function $K(x_i, x_j)$ can be expressed as an inner product:

$$K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle \tag{2.9}$$

This means that all we need to know is how to compute the inner product, since we do not need to know an explicit form of $\phi()$. This method is known as the "kernel trick". The most popular kernel functions are linear, polynomial and sigmoid. Mostly kernel functions are applied to a fixed dimension of features with the real values. However, due to the simple requirements (e.g. an inner product between objects), it is easy to apply the kernel to other types of input objects such as strings, trees and graphs which can be categorized as structural input.

Recently the structural kernels have been used widely in the design of machine learning systems in several domains from data-mining [15, 35, 128, 142, 151, 34, 159] to natural language processing [30, 164, 44, 70, 79].

The biggest advantage of structural learning is to limit the effort of manual feature engineering by automatically generating a huge feature space for a given task. This is a crucial property when the targeted task suffers from the lack of expert knowledge in providing the most appropriate features for a given problem. In the following section we introduce the structural kernel type "Convolutional Tree Kernel" used in this thesis.

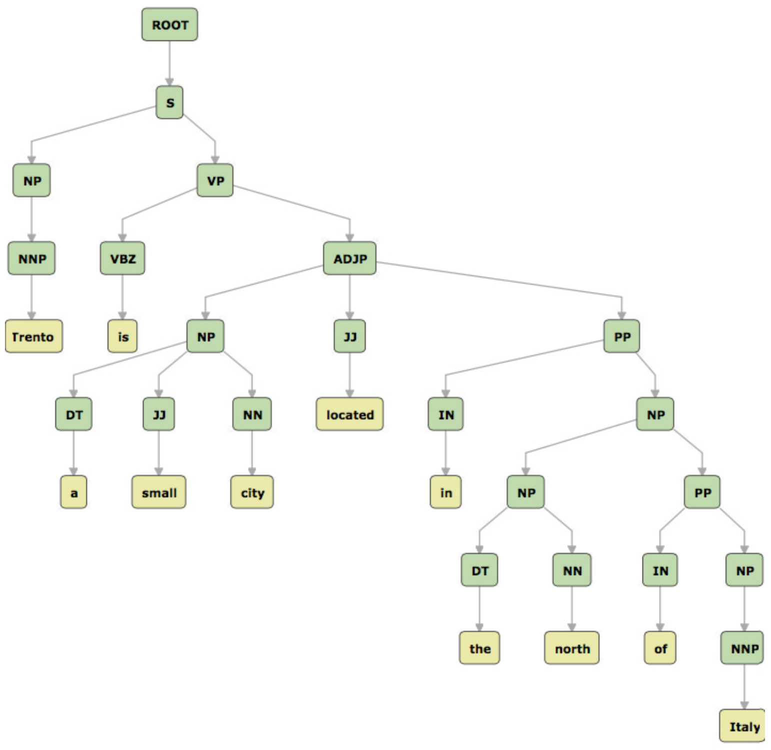


Figure 2.1: An example of Part of Speech (POS) tags and its syntactic tree

Convolutional Tree Kernels (TKs) Basically, Convolutional TKs count the common number of substructures between two trees $T1$ and $T2$. For a given set of substructure $\tau = \{t_1, t_2, ..., t_{|\tau|}\}$, $\chi_i(n)$ denotes an indicator function, which is equal to 1 if the target t_i is rooted at a node n or 0 otherwise. A tree kernel function over $T1$ and $T2$ can be define as:

$$K_{TK}(T1, T2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2) \tag{2.10}$$

where N_{T_1} and N_{T_2} are the sets of the $T1$'s and $T2$'s nodes, respectively and

$$\Delta(n_1, n_2) = \sum_{i=1}^{|\tau|} \chi_i(n_1)\chi_i(n_2) \tag{2.11}$$

which computes the number of common fragments rooted in the n_1 and

n_2 nodes. The most practiced technique to convert the textual data as input for structural kernels like TKs is to hierarchically organize words in a sentence into nested constituents. The constituency parse tree presents several layers of information such as Part-of-Speech (POS), phrases and the syntactic/semantic relations between constituents. Figure 2.1 shows an example of a sentence and its parsed syntactic tree.

In the next sections of this chapter, we introduce the definition of our targeted NLP tasks to valuate our hypothesis using various ML approaches. Each section is dedicated to describe one task and its related dataset that we use in our experimental setup.

2.2 Relation Extraction

2.2.1 Definition

Relation Extraction (RE) is a well-known task in the Natural Language Processing subarea. It aims to extract relation types between two named entities from text. In other words, as defined in ACE 2004[1], the task of RE is, given a set of documents where entities have been previously detected (manually or automatically), to identify the occurrences of relations between such entities (i.e. locating pairs of related entities in text).

For instance, in the sentence:

"The University of Trento is an international university located in Italy. It was founded in 1962 as a Higher University Institute for Social Sciences."

the identified relation types between two entity mentions can be denoted by the triples:

$$Founding - year\ (\textit{University of Trento, 1962})$$

[1]https://www.ldc.upenn.edu/collaborations/past-projects/ace

$$Founding - location\ (University\ of\ Trento,\ Italy)$$

Recently, the Relation Extraction (RE) task has attracted the attention of many researchers due to its wide range of applications such as: (i) Creating new structured knowledge bases, useful for any application; (ii) Enriching available knowledge bases; (iii) Adding new facts to Wikidata[2] or DBPedia[3] and (iv) Improving Question Answering task. For example in the latter, the question " The granddaughter of which actor starred in the movie E.T.?" can be reformulated as a relation extraction problem like: (acted-in ?x "E.T.")(is-a ?y actor)(granddaughter-of ?x ?y).

2.2.2 Related Work on Relation Extraction

Extracting relations from text has become a popular task in the IE community. Nowadays there are plenty of relation extraction systems available for the both industry and the academic use. Also several approches have been proposed for dealing with this problem. Banko et al. [16] classifies all the methods used for relation extraction into three classes:

- supervised methods;
- semi supervised methods;
- self-supervised methods.

In this section, we start with supervised methods which represent the extraction of the relations in the form of a binary classification problem. In addition, we elaborate feature-based [69, 171] and kernel-based approaches [27, 26, 164] more in details in the current Section.

[2]//www.wikipedia.org
[3]http://wiki.dbpedia.org

Supervised Methods

Basically supervised methods are built by using a small set of examples that have been tagged manually [164, 42, 69]. Such systems learn to extract the relations by using machine learning techniques. However, these approaches suffer from some drawbacks :

- Tagging a suitable corpus can take a lot of time and human effort.
- The tagged corpora are usually small and cover few relations. Therefore, the generated models are often domain-dependent.

Mainly two common approaches have been used extensively to train the extractor in a supervised manner. They are:

Feature-based Approaches: In this approach, a set of semantic and syntactic features are extracted and presented to the classifier in the form of a feature vector. For instance, Kambhatla et al. [69] extracted a set of features consisting of (i) the entities themselves, (ii) the types of the two entities, (iii) word sequence between the entities, (iv) number of words between the entities and (v) path in the parse tree containing the two entities using a log-linear model to classify the entities. Zhao et al. [171] trained SVMs by using polynomial kernel with the same feature set for classifying different types of entity relations.

Kernel-based Approaches: The main advantage of kernel-based method is to provide a opportunity for us to explore a large feature space in polynomial computation time. It is feasible even without introducing manually engineered features to the classifier. We will focus on several well known kernels such as the tree kernel [164], the subsequence kernel [25], and the dependency tree kernel [26].

The kernels used for relation-extraction (or relation-detection) are based on string-kernels as proposed by Lodhi et al. [82]. Bunescu et al. [26], proposed three subkernels defined as: one each for matching the before, middle and after portions of the entities context and the combined kernel is simply the sum of all the sub-kernels. Using subsequence kernels in conjunction with SVMs improves both the precision and recall. Compared to the previous approach, Zelenko et al. [164] replace the strings in the kernel with structured shallow parse trees built on the sentence. They use tree kernels and plug them into the SVM. Among others, tree kernels (TKs) have been widely used in supervised settings and have shown promising results.[27, 102, 100, 164]

Semi Supervised Methods

In semi-supervised approach, initially a small number of seed instances are manually annotated and used to extract the patterns from a big corpus [8, 19]. In this method, the annotated instances help the system to extract more instances and new patterns iteratively. However, this method suffers from availability of a limited number of relation types.

Self Supervised Methods

The aim of the self-supervised systems is to make the information extraction process unsupervised. Etzioni et al. [16] proposed the Web IE system called “KnowItAll” as a self-supervised system which learns to label its own training instances. To do that, a small set of domain-independent extraction pattern was used. Another sucessful project was the Intelligence in Wikipedia (IWP) proposed by Weld et al. [158] which bootstraps from Wikipedia corpus exploiting the fact that each article corresponds to a primary object and that many articles contain infoboxes (brief tabular information about the article). The drawback of IWP method is that the amount of relations described in Wikipedia infoboxes is limited. So not all relations can be extracted using this method. In another research, Banko et al. [16, 17] proposed The Open Relation Extraction system under name of “TextRunner”. This method does not pre-suppose a predefined set of relations and is targeted at all relations that can be extracted.

2.2.3 Distant Supervision

Distance Supervision (DS) is an alternative approach to overcome the problem of data annotation [40] as it can automatically generate training data by combining a structured Knowledge Base (KB), e.g., Freebase[4] with a large-scale unlabeled corpus, C.

The basic idea is: given a tuple $r{<}e_1{,}e_2>$ contained in a referring KB, if both e_1 and e_2 appear in a sentence of C, that sentence is assumed to express the relation type r, i.e., it is considered a training sentence for r.

For example, given the KB relation, *president(Obama, USA)*, the following sentence:

Obama has been elected in the USA presidential campaign,

can be used as a positive training example for *president*(e_1,e_2).

However, DS method suffers from two major drawbacks: first, in early studies, [90] assumed that two entity mentions cannot be in a relation with different relation types r_1 and r_2. In contrast, Hoffmann et al. [57] showed that 18.3% of the entities in Freebase that also occur in the New York Times 2007 corpus (NYT) overlap with more than one relation type.

Second, although the DS method has shown some promising results, its accuracy suffers from noisy training data caused by two types of problems [57, 119]:

- Simple string matching of entity pairs does not take into account the meaning and the context of the sentence.
- KBs are incomplete, e.g., a sentence can express relations that are not

[4] http://www.freebase.com/

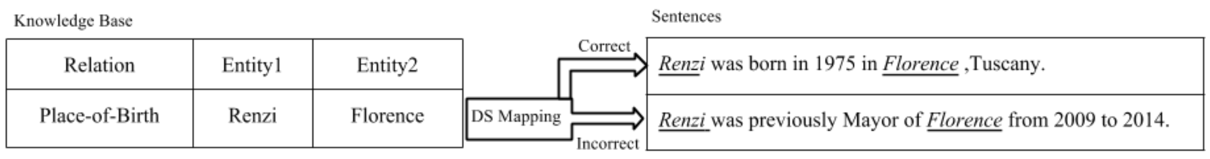

Knowledge Base
Relation
Entity1
Entity2
Place-of-Birth
Renzi
Florence
DS Mapping
Correct
Incorrect
Sentences
Renzi was born in 1975 in Florence ,Tuscany.
Renzi was previously Mayor of Florence from 2009 to 2014.

Figure 2.2: Error in automatic labeling generated through DS

in the KB. Thus DS leads to annotate relations as "No Relation" in case of unseen entities (this generates false negatives).

For example, let us consider the *Place-of-Birth* relation between $<Renzi, Florence>$: the upper sentence of Figure 2.2 supports the relation label whereas the lower sentence does not.

This generates a training instance misleading the classifier.

Several approaches for selecting higher quality training sentences with DS have been studied, but comparing such methods is difficult due to the lack of well-defined benchmarks and models using DS.

Related Work on DS

Distant Supervision (DS) has emerged as a popular method for training semantic relation extractors. It was used for the first time in the biomedical domain [40] and the basic idea was to extract binary relations between protein and cell/tissues by using the Yeast Protein Database (YPD) corpus. This method is getting more and more popular and different types of RE problems are being addressed [24, 90, 119, 101, 58, 120, 139, 57].

As one of the pioneers in using DS for RE task, Mintz et al. [90] used a large semantic database Freebase containing 7,300 relations between 9 million named entities. They extracted a set of syntactic and semantic features to train a relation extraction system. Although they achieved 67,6% of precision using this method, still there is a room for improvement.

The preliminary models of relation extractors (include distant supervision) assumed that a pair of entities can have only one relation. However, as we mentioned before, the assumption was restrictive and often violated by the real scenarios. For example in the following sentences :

Steve Jobs ***founded*** *Apple*
Steve Jobs is the ***CEO*** *of Apple*

the entity pair *(Steve Jobs, Apple)* can be bounded by two relations.

This problem has been identified any addressed in MultiR model by Hoffmann et al. [57] using Multi-Instance Multi-label (MIML) approach. They employ distant learning with Multi-Instance learning with overlapping relations (where two of the same instances may be in two different relations). In another research, Surdeanu et al. [139] addressed the same issue. They discovered 31% of training examples obtained by DS strategy are not valid. Therefore they try to improve distant learning by taking into account Multi-instance Multi-label settings (a.k.a MIML-RE) and using the Bayesian framework which can capture dependencies between labels and learn in the presence of incorrect and incomplete labels.

In addition, Yao et al. [160] proposed a method to combine raw text and a pre-existing structured databases to learn the relations. In the same direction, Riedel et al. [120] tried to reformulate the problem using matrix factorization and collaborative filtering where all the relations between entities are presented as a matrix. They used three approaches namely: (i) latent feature model; (ii) neighborhood model and (iii) entity model, which learns a latent entity representation from data. Moreover a combination of three models also presented in the method using various weights.

Datasets

We use two corpora for our experiments on relation extraction using distant supervision task namely: TAC-KBP 2010 and the New York Times:

TAC-KBP TAC Knowledge Base Population (KBP) consists of more than 1.3 million of newswire and web documents to expand and populate Knowledge Base (KB) with this information. Attributes (a.k.a. "slots") derived from Wikipedia infoboxes are used to create the reference knowledge base (KB). The reference knowledge base includes hundreds of thousands of entities based on articles from an October 2008 dump of English Wikipedia which includes 818,741 nodes.

A named entity tagger has been used on the entire source collection to map the documents with KB [66]. The TAC-KBP dataset is used for performing some experiments extensively presented in Chapter 7.

New York Times-Freebase The original New York Times corpus includes 1.8 million articles written and published by the NYT between January 1987 and June 2007 [124]. The data set consists of two parts for training and the testing, where the first part refers to the years 2005-2006 of the NYT whereas the second refer to the year 2007. In this dataset, Freebase[5] has been used as an external KB.

We used the same corpus provided by Riedel et al. [119], however, in his corpus instead of the entity mentions, their corresponding IDs in Freebase have been tagged (because of previous copyright issues). The old version of Freebase 2007 is not available anymore and in many cases the IDs or entities have changed in Freebase 2014. So, it was not possible to combine NYT with the newer Freebase to apply DS. To deal with this problem, we mapped the old Freebase IDs with the IDs from the Freebase 2014 and, if the entities were not the same, we asked an annotator to manually tag the entity mentions in the sentence. As the result, we created a new dataset that is mapped with Freebase 2014 and it is usable as a stand-alone DS corpus.

Overall, 4,700 relations in the training set and 1,950 in the test set are extracted. The NYT-Freebase corpus is used to perform some experiments

[5]https://developers.google.com/freebase/

which is described in details in Chapter 4.

2.3 Question Answering

2.3.1 Definition

Given a question and a set of candidate passages, the Question Answering (QA) task is to find the best answer among the candidates which can support the answer choice.

Typically QA systems consist of three major modules in the following order: (i) search and retrieve a set of candidate passages based on query (question); (ii) re-rank the retrieved candidate passages; and (iii) extract the best answer/s.

Automated Question Answering (QA) is a complex task requiring rules and syntactic patterns which are manually designed. The goal of the rules is to find the relations between the question and its set of candidate passages. The rules are triggered when the pattern exists in both the question and the passage. However, manually designing these rules is not an easy task since natural language is too complex to be characterized by a finite set of rules. Machine learning has been proposed as an alternative solution to shift the problem of manual rule definition to feature engineering.

In this thesis we focused on factioid QA task which the main purpose is to provide a consice fact for "WH" questions, often start with "WHOM, WHERE, WHEN, HOW and WHY?" question words [94]. QA systems merge information retrieval with information extraction methods to identify a set of likely set of candidates and then to produce the final answers using some ranking scheme [75].

2.3.2 Related Works on QA

TREC had a major impact on interest in question answering task and on the development of evaluation measures that compare the performance of different QA systems [150].

The contemporary question answering from unstructured data sources was instantiated by the TREC Evaluation campaign which is taking place regularly every year since 1999 [148, 149, 29] . The challenge is to provide a concise answer to a natural language question, given a large collection of textual documents. The first TREC evaluation campaign provides a list of 200 questions and a document collection. In the next campaign, TREC-9 held in 2000, the number of questions and size of document collections were increased. In TREC from 2002 to 2007, the list of questions, definition questions, and factoid questions were included in the evaluation campaigns. In TREC 2005, there was a set of 75 topics which contains various types of questions (list, factoid or others).

In recent evaluations two well-known collections of documents were used: AQUAINT with more than 1 million documents and AQUAINT2 with about 907K documents. In TREC-10 in 2001, a new complexity with respect to answers, i.e., answer validation task was included as there was no assurance of all answers to be present in the document collections. The lengths of answers were reduced to 50 words.

Throughout the evaluation campaigns we note a steady increase in question complexity, and as a result more advanced techniques for question answering were developed. Some of the key QA-techniques include: the incorporation of expected answer type as in [53, 93, 167], temporal question answering [91, 51, 9, 125, 112], geographical question answering [49], logic-based representations [92], semantic-role labelling for question answering [98, 105, 129], syntactic and semantic structures for question answering [59, 11, 41, 56], and discourse relationships and textual entailment for question answering [48, 52] .

Regarding QA and in particular answer sentence/passage reranking there has been a large body of work in the recent years, e.g., see [115, 65, 130, 97, 138, 153, 55, 152, 161] which the further exploration is beyond the scope of this thesis.

2.3.3 Dataset

We use the questions from TREC 2002 and 2003 from the large newswire corpus, AQUAINT. We created the Q/AP pairs training BM25 on AQUAINT and retrieving candidate passages for each question.

2.4 Recognizing Textual Entailment

2.4.1 Definition

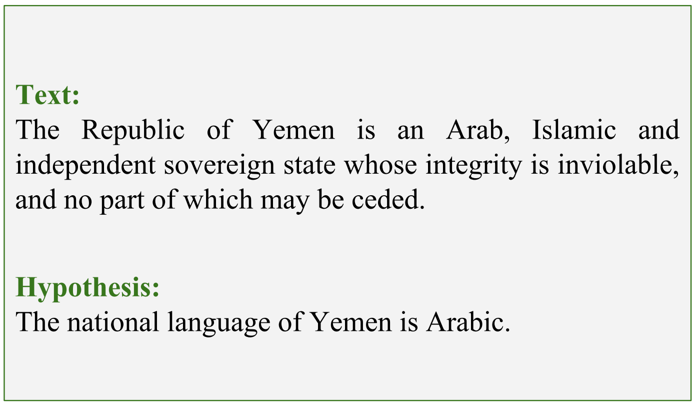
The Recognizing Textual Entailment task was introduced by Dagan et al. [43] for the first time where textual entailment is defined as a directional relationship between pairs of text.

Basically, the entailing "text" denoted by T, and the entailed "Hypothesis" denoted by H . We say that T entails H if the meaning of H can be inferred from the meaning of T. Usually Hypothesis (H) is a short statement whereas Text (T) is a longer span of text.

Figure 2.3 shows a sample of Text and Hypotheses being in entailment relation.

2.4.2 Related works on RTE

In this section we briefly reveiw the background works on the RTE task and its applications. Androutsopoulos et al. [13] provided a comprehensive


Text:
The Republic of Yemen is an Arab, Islamic and independent sovereign state whose integrity is inviolable, and no part of which may be ceded.

Hypothesis:
The national language of Yemen is Arabic.

Figure 2.3: Example of a Text and Hypothesis in entailment relation

survey of paraphrasing and textual Entailment methods.

RTE challenges provide a great platform for Textual Entailment systems and because of it, a wide variety of approaches emerge every year. Some approaches are proposed based on machine learnign methods [6]. Ren et al. [118] proposed a system of classification based on lexical, semantic and syntactic features. In another approach, Harmeling [54] used a probabilistic model which works based on calculus on dependency parse trees. Kouylekov et al. [78] used tree edit distance algorithms which were also used by other researchers [18, 28].

Heilman et al. (2010) proposed tree edit models for representing sequences of transformation and employs tree kernel heuristic in a greedy search routine [55]. Mehdad et al. [89] proposed an approach for cross-lingual textual entailment where bilingual parallel corpora have been used. In the same direction they obtained good results on monolingual corpus as well [87, 88]. Zanzotto et al. [162] employed machine learning models by using distance features models. The distance defined between the text and the hypothesis such as number of words in common, length of longest common subsequence and longest common syntactic sub-tree.

Machine Learning and Probability based approaches are not the only approaches used for RTE rask. Another successful approach was proposed by determining the deep semantic inferences from the text. Approaches based on logical inferences [20] and the application of natural logic [83] yielded good accuracy.

2.4.3 Dataset

The RTE-2 dataset consists of 1,600 text-hypothesis pairs, divided into a development set and a test set, each containing 800 pairs [18]. The pairs were generated by considering different NLP applications, e.g., Information Extraction, Information Retrieval, QA, etc. and the pairs are extracted from different resources such as MUC-4, ACE 2004 and TREC QA. In order to create gold labels, each pair was judged by at least two experts. The RTE-2 dataset is used in running some experiments elaborated in Chapter 6

In the last section of this chapter we introduce the metrics for evaluation the proposed classification and re-reranking models in this thesis.

2.5 Evaluation Metrics

The goal of measuring the performance of an Information Retrieval (IR) system is to evaluate how good the search results satisfied the received query intent. Several metrics have been used widely to measure the performance of IR approaches such as MRR, MAP and P@k.
Besides that Precision and Recall are two fundamental scores for quantifying the quality of predicted labels in classification approaches:

2.5.1 Precision

Precision is a score of showing the proportion of all positive predictions that are correct. Precision is a measure of how many positive predictions were actual positive observations:

$$Precison = \frac{TP}{TP + FP} \tag{2.12}$$

where TP and FP indicate true positive and false positive predictions respectively.

2.5.2 Recall

The recall score reveals the proportion of all real positive observations that are correct:

$$Recall = \frac{TP}{TP + FN} \tag{2.13}$$

which FN refers to false negative predictions.

2.5.3 F1 Measure

The weighted harmonic mean of precision and recall, the balanced F-score is:

$$F_1 = \frac{2 \times Precison \times Recall}{Precison + Recall} \tag{2.14}$$

2.5.4 Precision at Rank k (P@k)

P@k indicates the number of relevant results retrieved by a query (e.g., the correct retrieved documents) found in the first top k positions. it is a common practice to define k with different values such as 1, 5 or 10.

However, in all evaluations related to reranking performance, we keep the k value equal to 1.

2.5.5 Mean Average Precision (MAP)

Average Precision (AP), is a widely used evaluation metric in information retrieval tasks which combines recall and precision for ranked retrieval results. Mean Average Precision (MAP), measures AP across multiple queries/answer passage rankings.
The mean average precision for K queries at position n is the average of the average precision of each query, i.e.,

$$MAP@n = \sum_{i=1}^{K} \frac{AP@n_i}{K} \tag{2.15}$$

Note that, while generally in AP order matters - but not always. Order matters only if there is at least one incorrect prediction. In other words, if all predictions are correct, it doesn't matter in which order they are given.

2.5.6 Mean Reciprocal Rank (MRR)

The Mean Reciprocal Rank is the average of the reciprocal ranks of results for a sample of queries Q:

$$MRR = \frac{1}{|Q|} \sum_{q=1}^{|Q|} \frac{1}{rank(q)} \tag{2.16}$$

where *rank(q)* is the position of the first correct answer in the retrieved list. For a group of queries Q.

3

Annotation Collection for Supervised Methods

Supervised methods in NLP require high quality annotated data to train accurate models. Traditional data annotation methods are expert-intensive. Manually providing large-scale expert-labeled training data is costly in terms of resources and time. So, the corpora created by experts are often small. The expert-driven method has disadvantages such as, (i) a small-size corpus can only contains few classes/label types and (ii) the resulting trained model is usually domain-dependent.

On the other side, ML algorithms face critical challenges such as scalability due to the fact that finding qualified experts in many domains is not easy. For instance, in NLP, it is often hard to find annotators with proficiency in certain languages. This is particularly critical when dealing with widely-spoken languages that are nevertheless considered rare languages in terms of currently available resources. Therefore, it is not easy to improve the ML models by scaling up the datasets.

3.1 Annotation Collection for Big Data

It's a known fact that the recent powerful ML methods are strongly data and supervision demanding like deep learning. Big data enables ML algorithms to uncover more fine-grained patterns and make more accurate predictions than ever before. However, it is expensive and impractical to use only experts to provide such supervised data on a large scale.

Several alternative methods have been suggested to address the challenge of annotating big data. For instance, automatically labeling the big data using Distant Supervison (DS) approach (See Section 2.2.3) or online crowd-generated repositories that serve as a source for big annotated training data, which can capture a large variety in terms of both class number and intra-class diversity [103].

3.1.1 Crowdsourcing

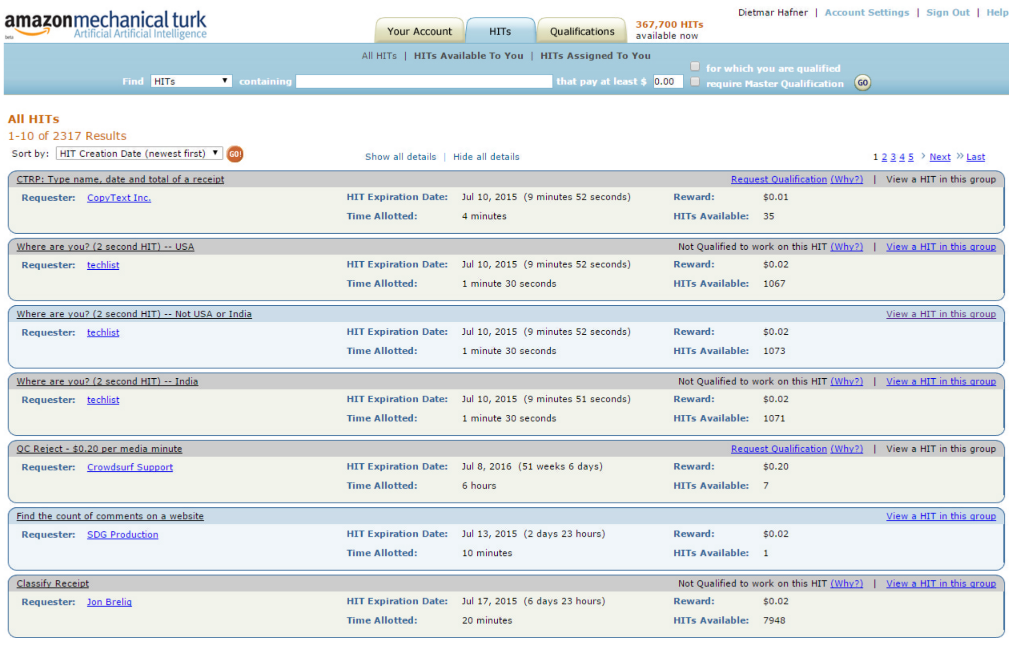
The term "Crowdsourcing" was originally introduced by Howe in 2006 in Wired Magazine [60]. He described it as a web-based movement of ideas and opinions of the crowd for problem-solving and innovation mechanism [154]. However, this idea of using the Internet to facilitate reaching large groups of people to outsource work to them had existed long before Howe coined the term. Recently, crowdsourcing has become a very popular topics in both academia and industry. By hiring workers online, the task assigners can take advantage of the wisdom of the crowd and solve problems that used to be solvable only by experts when the problems are too hard for automated computer algorithms. With the power of crowdsourcing, companies and individuals are no longer limited by their internal resources and can take advantage of the knowledge, energy, and creativity of a global online and offline community. Crowdsourcing provides us an opportunity to harness the power of crowd to create brilliant platforms and applications via voluntary crowd works, such as Wikis [23], Games With A Purpose (GWAP) [146], Captchas [10], and Citizen Science [39, 114, 126].

These platforms include general purpose marketplaces (e.g., Mechanical Turk, oDesk, Freelancer, Crowdflower, MobileWorks, ManPower) as well as markets for specific expertise (e.g., TopCoder, uTest, 99Designs).

Micro-task Crowdsourcing Micro-tasking is a branch of crowdsourcing, where task designers break up a large task into small components in the form of identical microtasks, each requiring online contributors. To crowdsource a task, the task designer, also called the requester, submits the task to a crowdsourcing platform. People who can accomplish the task, called workers, can choose to work on it and devise solutions.
Workers then submit these contributions to the requester via the crowdsourcing platform [45]. Common Microtasking platforms have led to the development of online marketplaces such as Mechanical Turk and Crowdflower.

Amazon Mechanical Turk (MTurk) is the world's leading crowdsourcing venue, however, it is not accessible in Europe. The main motivator of MTurk is profit. Providers create and list batches of small jobs termed Human Intelligence Tasks (HITs) on Amazon's Mechanical Turk website, which may be done by the general public. Workers who fulfill these tasks get credited in micro-payments. While certainly not the only paid labor sourcing environment, Mechanical Turk's current ubiquity make "MTurk" a useful label to refer to this and other forms of computer mediated labor. Figure 3.1 shows the user interface of MTurk platform.

Many factors affect the quality of the collected annotations which change the final decision of the annotators. For example, poor instructions can introduce more confusion than clarit. Low level task design, unfair reward amounts, anonymity of the workers performing tasks, not paying much attention to get rewards faster and programmed bots which submitting random results automatically all affect the quality of results dramatically. Hence, basic and advance quality control mechanisms are using extensively to tackle the noise problem in annotated data.


amazonmechanical turk
Artificial Artificial Intelligence
Your Account
HITs
Qualifications
All HITs | HITs Available To You | HITs Assigned To You
Find
containing
that pay at least $
GO
Show all details | Hide all details
1 2 3 4 5 Next Last

367,700 HITs available now

Dietmar Hafner | Account Settings | Sign Out | Help

Find HITs containing that pay at least $ 0.00 ☐ for which you are qualified ☐ require Master Qualification

All HITs
1-10 of 2317 Results
Sort by: HIT Creation Date (newest first)

CTRP: Type name, date and total of a receipt — Request Qualification (Why?) | View a HIT in this group
Requester: CopyText Inc. HIT Expiration Date: Jul 10, 2015 (9 minutes 52 seconds) Reward: $0.01
Time Allotted: 4 minutes HITs Available: 35

Where are you? (2 second HIT) -- USA — Not Qualified to work on this HIT (Why?) | View a HIT in this group
Requester: techlist HIT Expiration Date: Jul 10, 2015 (9 minutes 52 seconds) Reward: $0.02
Time Allotted: 1 minute 30 seconds HITs Available: 1067

Where are you? (2 second HIT) -- Not USA or India — View a HIT in this group
Requester: techlist HIT Expiration Date: Jul 10, 2015 (9 minutes 52 seconds) Reward: $0.02
Time Allotted: 1 minute 30 seconds HITs Available: 1073

Where are you? (2 second HIT) -- India — Not Qualified to work on this HIT (Why?) | View a HIT in this group
Requester: techlist HIT Expiration Date: Jul 10, 2015 (9 minutes 51 seconds) Reward: $0.02
Time Allotted: 1 minute 30 seconds HITs Available: 1071

QC Reject - $0.20 per media minute — Request Qualification (Why?) | View a HIT in this group
Requester: Crowdsurf Support HIT Expiration Date: Jul 8, 2016 (51 weeks 6 days) Reward: $0.20
Time Allotted: 6 hours HITs Available: 7

Find the count of comments on a website — View a HIT in this group
Requester: SDG Production HIT Expiration Date: Jul 13, 2015 (2 days 23 hours) Reward: $0.02
Time Allotted: 10 minutes HITs Available: 1

Classify Receipt — Not Qualified to work on this HIT (Why?) | View a HIT in this group
Requester: Jon Brelig HIT Expiration Date: Jul 17, 2015 (6 days 23 hours) Reward: $0.02
Time Allotted: 20 minutes HITs Available: 7948

Figure 3.1: Amazon Mechanical Turk (AMT) user interface

3.2 Quality Control in Crowdsourcing

In a microtasking crowdsourcing environment, quality control is one of the biggest issues addressed by people both from academia and industry. Ipeirotis et al. [64] raised the problem of quality control within crowdsourcing platforms like the Amazon Mechanical Turk. Specifically, it is difficult to ensure the quality output when the small tasks are performing by many anonymous crowd workers.

Quality is a subjective issue in general. Some efforts have proposed models and metrics to quantitatively and objectively assess quality along different dimensions of a task, such as reliability, accuracy, relevancy, completeness, and consistency [7].

The overall outcome quality depends on the definition of the task that's being crowdsourced and the contributing workers' attributes [74, 72]. We characterize quality in crowdsourcing systems along two main dimensions: worker profiles and task design. We used the same taxonomy proposed by [107], as Figure 3.2 illustrates in our work.

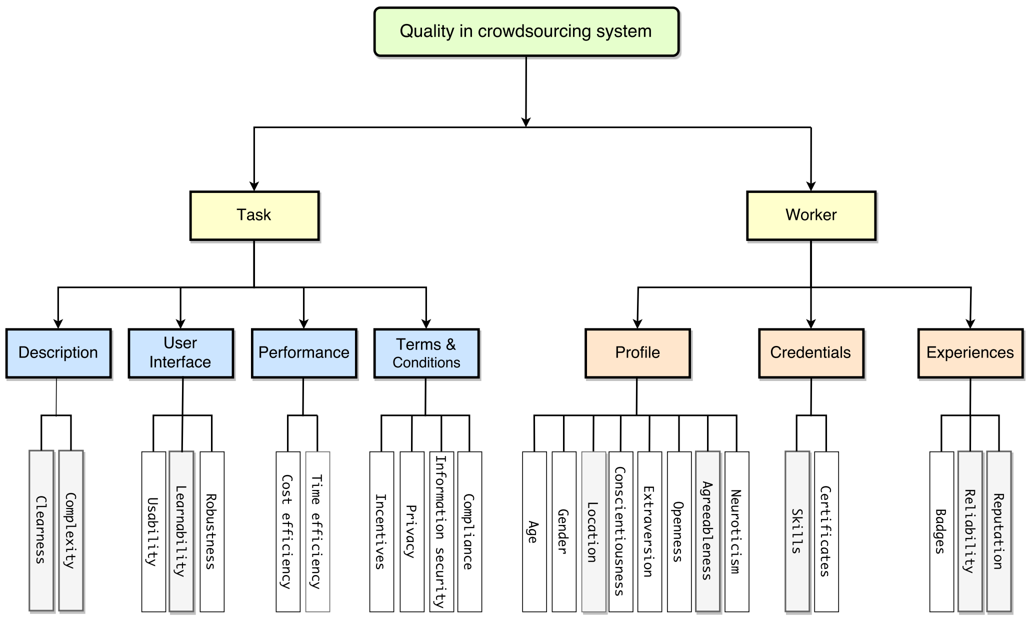


Figure 3.2: Quality control factors in crowdsourcing

We considered several factors in our proposed quality control methods which are highlighted in gray color. These metadata are collected from crowdsourcing platforms or estimated via monitoring the crowdsourced tasks' output.

3.2.1 Related Works on Quality Control in Crowdsourcing

There is already an established literature on this topic, the review of which is beyond the scope of the thesis. Therefore, we briefly review the most important statistical approaches related to quality control mechanisms in crowdsourcing tasks.

Majority voting is the simplest form of quality control on MTurk. Most annotation campaigns involve a small group of untrained annotators who may not always agree on their judgements [163, 127, 144]. The reliability of the annotation is typically assessed by quantifying the level of inter-annotator agreement, while the final annotation to be released is consensuated amongst the annotators. However, the majority voting is still prone to noise.

Kilgarriff [73] proposed a model for generating gold standard datasets for word-sense disambiguation. There are several works [147, 145, 12] which consider relevance judgments for building IR systems. Unlike outsourcing a task (when the targeted crowd is a community of experts), the quality of the collected annotations from ordinary crowd workers is difficult to control and several approaches have been proposed to decrease the level of the noise in the collected annotations.

One of the pioneer in crowdsourcing for NLP tasks is Snow et al. [134] who applied a simple Naive Bayes approach to weigh the labels by considering the accuracy of annotators. In the proposed method, the number of the judgments was fixed. In other work done by Ipeirotis et al. [64], the unrecoverable error rate from bias was separated and the authors used confusion matrix for each annotator to measure the uncertainty and cost associated with each label. Jung [68] applied unsupervised and supervised features to filter out noisy annotators via z-score. He also improved the quality of noisy labels via probabilistic matrix factorization [67].

Nowak et al. [106] filtered out noisy annotations by considering the degree of agreement between crowd workers. In another work done by Sheng et al. [131], they targeted the number of required annotations and asked for a new label when it was beneficial for some of the training examples.

Some other approaches have also been proposed to model the annotators' behavior and increase the accuracy when the labels are noisy [80, 141, 156]. Donmez et al. [47] proposed the use of active learning to deal with noisy annotations and elaborated how to jointly learn the accuracy of labeling sources and obtain the most informative labels for the active learning tasks.

Crowdsourcing has been used for optimizing machine learning performance as well. Quinn et al. [113] presented the CrowdFlow toolkit with which a human and machine can work together to achieve an optimal performance and control the trade-off between speed, cost and quality. Plank et al. [109] proposed a method for taking noise into account when training a classifier. While their approach can help a classifier deal with noisy annotations, it cannot improve the quality of the original data-set.

Raykar et al. [117, 116], proposed a probabilistic approach (EM) which can be used to carry out supervised learning when multiple annotators provide noisy labels, but there is no absolute gold standard. The authors showed that the proposed classifier is superior to the commonly used majority voting baseline. However, their method suffers from major drawbacks as the authors made two unrealistic assumptions: (i) they assumed the crowd workers are equally good and their performance is fixed across different expert sub-groups, and (ii) the annotators' performance have no correlation with the difficulty of the instances which have been annotated.

The most comprehensive survey is the one provided by Sheshadri et al. [132] that compared different approaches to identify relative labels in a collection of labels with a lot of noise. They presented SQUARE, an open source shared task framework for statistical consensus methods including benchmark datasets, defined tasks, standard metrics, and reference implementations with empirical results for several popular frameworks such as ZenCrowd [46], GLAD [157] and CUBAM [155] .

3.3 Our methods for Exploiting Labeled Data

In this section, we provide an overview of the problem statements and the challenges addressed in this thesis. Then, we introduce our proposed methods and the contributions in an itemized fashion to familiarize the readers with an overview of the thesis.

3.3.1 Create Benchmark for Distant Supervision for RE

In this section we introduce our works published in two conferences [2, 3]. In this research, we aim to build a baseline to compare with the other models using Distant Supervision (DS) for relation extraction task. So, we considered the most used corpus in DS, i.e., the combination of NYT and Freebase (NYT-FB). The description of the dataset is provided in Chapter 2.2.3.

Previously, all the RE models which experimented with NYT-FB are based on complex graphical models. This is necessary to encode the dependencies between the overlapping relations.

Additionally, such models use very particular and sparse features, which makes the replicability of the models and results complex, thus limiting the progress of research in DS. For comparing a new DS approach with the previous work using NYT-FB, a researcher is forced to re-implement a very complicated model and its sparse features.

Therefore, we believe that simpler models can be very useful as (i) a much simpler re-implementation would enable model comparisons and (ii) it would be easier to verify whether a DS method is better than another.

Contributions

The contributions of this work can be summarized as below:

- Our proposed approach is based on convolution tree kernels, which can easily exploit syntactic/semantic structures. This is an important aspect in favor of the replicability of our results.
- Our method differs from previous state of the art on overlapping relations [119] as we apply a modification of the simple one-vs-all strategy, instead of the complex graphical models. To make our

approach competitive, we studied several parameters for optimizing SVMs and filtering out noisy negative training examples.

- We mapped the Freebase entity IDs used in NYT-FB from the old version of 2007 to the newer Freebase 2014. Since the entities had changed over time, we asked an annotator to manually tag the entity mentions in the sentence. As the result, we created a new dataset usable as a stand-alone DS corpus to be made available for research purposes.

We described our method, the experimental setup and the results related to this work in Chapter 4.

3.3.2 Instance Weighting

In this section we targeted the problem of noisy human annotation tags in training Question Answering (QA) system. As discussed in Chapter 2.3, one of the most important steps for building accurate QA systems is the selection/reranking of answer passage (AP) candidates typically provided by a search engine. This task requires the automatic learning of a ranking function, which pushes the correct answer passages (i.e., containing the answer to the question) higher in the list. The accuracy of such a function, among others, also depends on the quality of the supervision provided in the training data.

Traditionally, the latter is annotated by experts through a rather costly procedure. Thus, sometimes, only noisy annotations obtained via automatic labeling mechanisms are available.

For example, the Text REtrieval Conference (TREC [1]) provides open-domain QA datasets, e.g., for factoid QA. This data contains a set of questions, the answer keywords and a set of unannotated candidate APs. The labeling of the latter can be automatically carried out by checking

[1]http://trec.nist.gov

if a given passage contains the correct answer keyword or not. However, this method is prone to generate passage labels, i.e., containing the answer keyword but not supporting it. For instance, given the following question, Q, from TREC 2002-03 QA, associated with the answer key *Denmark*:

Q: *Where was Hans Christian Anderson born?*

the candidate passage:

AP: *Fairy Tales written by Hans Christian Andersen was published in 1835-1873 in* ***Denmark****.*

would be wrongly labeled as a correct passage since it contains *Denmark*. Such passages can be both misleading for training and unreliable for evaluating the reranking model, thus requiring manual annotation.

Since the expert work is costly, we can rely on crowdsourcing platforms such as CrowdFlower [2] for labeling data, faster and at lower costs [134].

This method has shown promising results but still produces noisy labels. Thus, a solution consists of (i) using redundant annotations from multiple annotators and (ii) resolving their disagreements with a majority voting approach [131, 168].

However, the consensus mechanism can still produce annotation noise, which (i) depends on the skill of the crowd workers and the difficulty of the given task; and (ii) can degrade the classifier accuracy.

In this work, we study methods to take into account the disagreement among the crowd annotators as well as their skills in the learning algorithms. For this purpose, we design several instance weighting strategies, which help the learning algorithm to deal with the noise of the training examples, thus producing higher accuracy.

[2]http://www.crowdflower.com

Contributions

- We define some weight factors that characterize crowd annotators' skill, namely:

  Prior Confidence, which indicates the previous performance of the crowd worker reported by the crowdsourcing platform;

  Task Confidence, which is determined by the total number of annotations performed by the crowd worker in the target task;

  Consistency Confidence, which quantify the agreements between the annotator and the majority voting labels.

  We used these parameters for building our weighting functions, which aim at reducing the impact of the noisy annotations in learning algorithms.

- We build a passage reranking dataset based on TREC 2002/2003 QA. We used Crowdflower for carrying out an initial noisy annotation and then had an expert manually verify and correct the incorrect labels. This is an important QA resource that we will release to the research community.

Additionally, the accuracy of our models, e.g., classifiers and search engines, tested on such gold standard data can establish new baselines, useful for future research in the field.

Finally, we conducted comparative experiments on our QA dataset using our weighting strategies. The results show that (i) our rerankers improve on the IR baseline, i.e., BM25, by 17.47% and 19.22% in MRR and P@1, respectively; and (ii) our weighting strategy improves the best reranker (using no-weighting model) up to 1.47% and 1.85% on MRR and P@1, respectively. Our methodology [4], experimental setup and the results are described in Chapter 5 extensively.

3.3.3 Optimized Automatic Label Selector (ALS)

This work can be considered as an extension of the previous research we have performed. As we mentioned before, a major problem is that consensus mechanism in crowdsourcing tasks is inaccurate and still produces many incorrect annotations depending on the skill levels of the crowd workers and the difficulty of the given task. These factors are totally ignored in the majority voting mechanism.

Unlike the instance weighting method where the weights are constant for each instance, here we aim at learning a statistical model to predict the best label from a set of annotations done by a group of annotators and by quantifying their skills. More in detail: firstly, we define some annotator descriptors using various parameters characterizing each annotator behavior, namely:

Prior Confidence, which indicates the previous performance of the crowd worker reported by the crowdsourcing platform;

Cardinality Confidence factor, which is determined by the total number of annotations performed by the crowd worker;

Majority Consistency Confidence, which is extracted from the number of agreements between the annotator and the majority voting labels;

Precision and Recall in the current annotation task.

First, We used these parameters in the aggregated voting functions which considered as descriptor features for each example to be labeled positive or negative. In addition, we exploit the syntactic information of the examples to represent the structure of the given tasks. It has performed automatically using Tree Kernel (TK) without considering any feature engineering process. We use such feature representation in Support Vector Machines for learning an Automatic Label Selector (ALS) for assigning the most correct labels out of the annotator labels.

Second, learning and testing our ALS models requires ground truth (gold standard). For this purpose, we used 5 crowd annotators and we had an expert analyze and judge all the cases where disagreement existed. In particular, we built a passage reranking dataset based on TREC 2002/2003 QA. Moreover, we created a dataset based on the RTE task consisting of the judgments of 5 crowd annotators and a set of annotators for future investigation and research.

Third, we conducted comparative experiments on our datasets using different settings of features and parameters. The results show that (i) ALS greatly improves upon the majoring voting by reducing the annotation error by 7% on the passage reranking task and by 20% on the RTE 2.

Our analysis shows that the classifier's optimal performance is learned with a relatively small number of training instances. It means obtaining a limited amount of gold data from experts as seed can enable us to have more accurate annotations using the classifiers instead of the majority voting scheme.

Finally, we experimented on the passage reranking task with the majority voting and the classifier annotated data.

The results show that using the improved data enhances the performance of the reranker by 1.47% and 1.85% on MRR and P@1, respectively. Our results on the RTE task showed that we could improve the accuracy of the task up to 3.75% when the RTE classifier associated with the classifier meta data compared to majority voting. Chapter 6 discusses about the crowdsourcing configuration, ALS proposed framework, the experimental setup and the results in details [1], .

3.3.4 Autonomous Crowd Learning

In this work, we study the effect of workers training on the quality of the collected annotations. So, we targeted Relation Extraction (RE) task as a complex task to be crowdsourced. The primary works on RE was mainly *strongly supervised*, where a manually annotated data is required to train (See Chapter 2.2 for more details). Fully supervised relation extractors are limited to relatively small training sets. However, distant supervision methods provide us an opportunity to use much more data., despite its noisy characteristic. We hope to combine the benefits of supervised and distantly supervised methods by annotating a small subset of the available data using crowdsourcing.

It has been shown that by simply adding a small set of high quality labeled instances (i.e., human-annotated training data) to a larger set of instances annotated by DS, the overall precision of the system increases significantly [57]. Such level of quality of the labels usually can be obtained via a large-scale labor-on-demand crowdsourcing platform at low cost. Specifically, Snow et al. [134] demonstrated that the crowd workers are able to generate high quality labels for various NLP tasks.

However, this does not hold for fine-grained tasks, where the annotators[3] need to have some expertise on it. For instance in RE, several papers have shown that only a marginal improvement can be achieved via crowdsourcing the data [14, 166, 108]. In all the mentioned papers, the well-known gold standard quality control mechanism was used without annotators being trained.

Very recently, despite the previous results, Liu et. al [81] showed that there can be a larger improvement for RE task, by training crowd workers using an interactive tutorial procedure called “Gated Instruction”. This approach, however, requires a set of high-quality labeled data (namely, gold standard) for providing the instruction and feedback to the crowd workers. Thus, a considerable amount of human effort is required for annotating the gold standard.

In this work, we tried to improve the accuracy of the relation extractor by involving human in the loop. So, we introduce an iterative

[3]From now, both the entities :*annotators* and *crowd workers* refer to the same concept.

human-machine collaborative learning method for crowdsourcing. In the proposed method, the classifier (i.e., machine) selects the highest quality examples to train the crowdsource workers (i.e., human). Then, the annotation for lower quality examples are collected to improve the original classifier. And this process can be repeated iteratively. Our experimental results show a significant improvement for the relation extraction task, compared to the method based on distant supervision alone as well as the ones based on the crowdsourcing without user training.

Contributions

Our study shows that even without using any gold standard, we can still train workers and their annotations can achieve comparable results with more costly state-of-the-art methods. In summary our contributions are the following:

- We introduce a self-training strategy for crowdsourcing.
- We propose an iterative human-machine co-training framework for the task of RE.
- We test our approach on a standard benchmark, obtaining a slightly lower performance compared to the gold-based state-of-the-art method.

Our proposed framework [5] and the comparison with state-of-the-art are structured and presented in Chapter7.

4

Distant Supervision for Relation Extraction

4.1 Overview

In this chapter, we aim to build a standard to compare models based on DS: firstly, we considered the most used corpus in DS, i.e., the combination of NYT and Freebase (NYT-FB).

Secondly, we mapped the Freebase entity IDs used in NYT-FB from the old version of 2007 to the newer version of Freebase 2014. Since entities changed, we asked an annotator to manually tag the entity mentions in the sentence. This resulted in a creation of a new dataset usable as a stand-alone DS corpus, which we make available for research purposes.

Finally, all the few RE models experimented with NYT-FB in the past are based on a complex conditional random fields. This is necessary to encode the dependencies between the overlapping relations.

Additionally, such models use very particular and sparse features, which make the replicability of the models and results complex, thus limiting the research progress in DS. Indeed, for comparing a new DS approach with the previous work using NYT-FB, the researcher is forced to re-implement a very complicated model and its sparse features. Therefore, we believe that simpler models can be very useful as (i) a much simpler re-implementation would enable model comparisons and (ii) it would be easier to verify if one DS method is better than another. In this perspective, our proposed approach is based on convolution Tree Kernels (TKs), which can easily exploit syntactic/semantic structures. This is an important aspect to favor replicability of our results.

Moreover, our method differs from previous state of the art on overlapping relations [119] as we apply a modification of the simple one-vs-all strategy, instead of the complex graphical models. To make our approach competitive, we studied several parameters for optimizing SVMs and filtering out noisy negative training examples. Our extensive experiments show that our models achieve satisfactory results.

4.2 DS Baseline using SVMs and TKs

As we described extensilvely in Section 2.1.2, SVMs are linear supervised binary classifiers that separate the class boundaries by constructing hyperplanes in a multidimensional space. They can also be used in non-separable linear space by applying kernel functions. Also, It can be used for multi-class classification by means of one-vs-all strategy [121]. One of the valuable feature of SVM is taking advantage of the kernel function when the classes are not separable linearly in the current feature space by mapping the objects into higher feature space.

Tree kernels (TKs) [37] have been proved to achieve state-of-the-art in relation extraction [170]. Different TKs have been proposed in the past [95]. We model our RE system by using feature vectors along with syntactic/semantic trees (See [169, 102]). The latter information are extracted by applying tree kernel function and syntactic parsing information.

4.2.1 Feature Vectors

In our experiment, we used the features proposed by [90]. It consists of two standard lexical and syntactic feature levels. Lexical/syntactic features extracted from a candidate sentence are decorated with different syntactic features such as: (i) Part of Speech (POS); (ii) the window of k words of the left and right of matched entities; (iii) the sequences of words between them; and (iv) finally, syntactic features extracted in terms of dependency patterns between entity pairs. The proposed features yield low-recall as they appear in conjunctive forms but at the same time they produce a high precision.

4.2.2 Tree Kernels

We use the same convolution tree kernels as described in [169] for syntactic parsing. Generally, given two relation examples R_1 and R_2, a composite kernel *K(*R_1,R_2*)* is computed as:

$$K(R_1, R_2) = \alpha \vec{x_1} \cdot \vec{x_2} + (1 - \alpha) K_T(T_1, T_2), \tag{4.1}$$

$$K_T(T_1, T_2) = \sum_{n_1 \in N1} \sum_{n_2 \in N2} \Delta(n_1, n_2), \tag{4.2}$$

where α is a coefficient multiplying the target kernel and $\vec{x_1} \cdot \vec{x_2}$ is a dot product between two feature vectors of R_1 and R_2. The function $K_T(T_1$,$T_2)$ is a kernel function applied to syntactic trees, where N_1 and N_2 are the set of nodes in the trees T_1 and T_2, respectively and Δ is the number of common sub-trees rooted at n_1 and n_2. Figure 4.1 shows a sentence tree (part a) and its associated tree (part b).

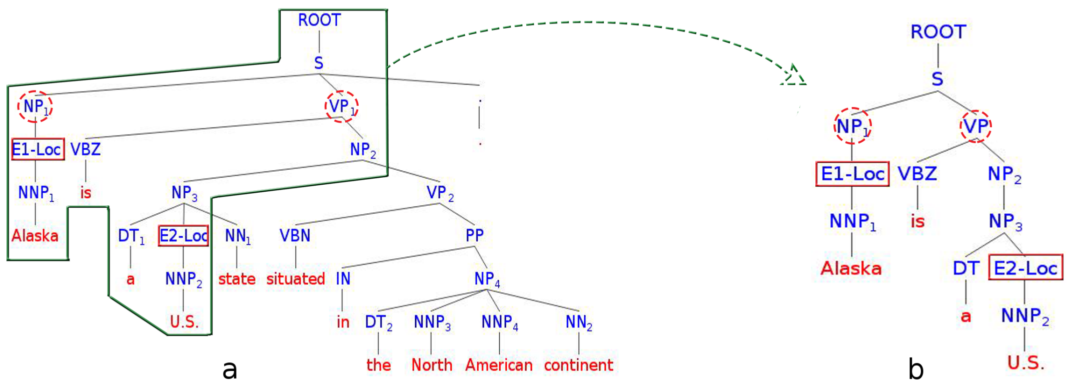


Figure 4.1: a) The constituent parse tree of the example sentence where "E1-Loc" denotes the source entity mentions and "E2-Loc" denotes the target entity. b) PT relation instance space of the sentence.

4.3 Experiments

4.3.1 Corpus

We trained our system on the NYT news wire corpus [124]. The original corpus includes 1.8 million articles written and published by the NYT between January 1987 and June 2007 (See 2.2.3). We used the same subset of data as Riedel et al. [119]. The data set consists of two parts for training and the test, where the first part refers to the years 2005-2006 of the NYT whereas the second refer to the year 2007.

In the corpus provided by [119], instead of the entity mentions, their corresponding IDs in Freebase have been tagged (this because of previous copyright issues). The old version of Freebase 2007 is not available anymore and in many cases the IDs or entities have changed in Freebase 2014. So, it was not possible to combine NYT with the newer Freebase to apply DS. To deal with this problem, we mapped the old Freebase IDs with Freebase 2014 and, when the entities were not the same, we asked an annotator to manually tag the entity mentions in the sentence. As the result, we created a new dataset that is mapped with Freebase 2014 and it is usable as a stand-alone DS corpus.

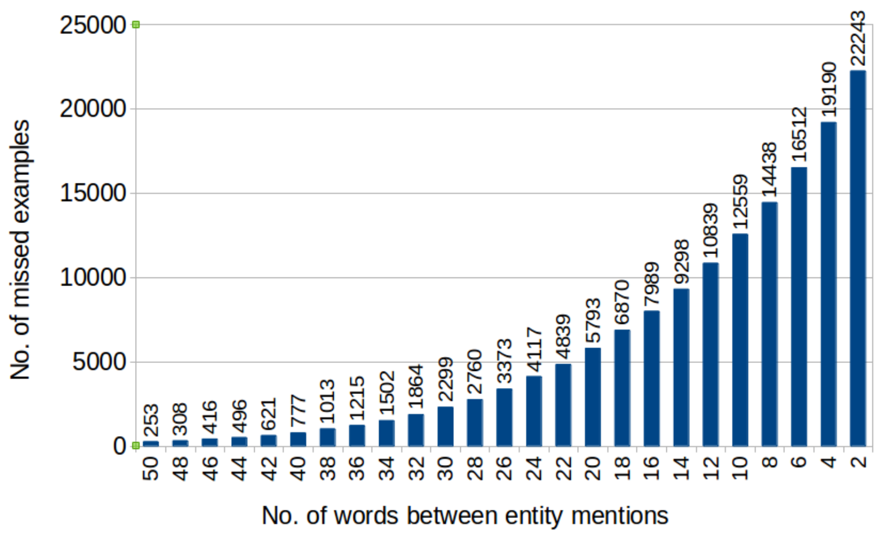


Figure 4.2: Recall of positive examples with respect to word distance between entity mentions.

In overall, we found 4,700 relations in the training set and 1,950 in the test set. The number of positive and negative examples is heavily imbalanced (1:134). So, a simple filtering is applied to discard noisy negative examples from the training set.

4.3.2 Pre-processing

In Section 2.2.3, we pointed out that (i) some sentences containing the target entities may not semantically realize the target relation and (ii) other sentences express a correct relation not in the KB. We tackle such problems by applying sentence filtering and enriching the relations of previous KB.

4.3.3 Sentence Filtering

We used four levels of noise cleaning to remove potential incorrect sentences from the corpus. More specifically, we remove a sentence if:

- The distance between the two target entity mentions is more than k words (e.g., 26). We set the k threshold value equal to 10% of the total number of positive examples as shown in Figure 4.2.

- The number of tagged entities between the entity mentions are greater than a constant h (e.g.,10).

- None of the entity mentions in the sentence appeared in positive examples before, i.e., at least one of the entity in the negative example has to be in a relation with another entity (i.e., it has to be part of previously generated positive examples).

- The same entity pairs were in a relation in positive examples but with different relation type (Overlap Relation). For instance, in the mention *Edmonton, Alberta, one of six Canadian N.H.L. markets, is the smallest in the league.*, the entity mentions <*Edmonton, Alberta*> are in relations with two relation types: *Province/Capital* and *Location/Contains*. Thus, to train Rel. 1, all the instances of Rel. 2 are removed and viceversa.

4.3.4 Knowledge Base Augmentation

We analyzed the entity pairs in the sentences of our corpus with respect to the relations in Freebase 2007. We discovered that many pairs receive no-relation because they did not exist in Freebase 2007. This creates many false negative (FN) errors in the generation of training data. In the new release of Freebase many new relations are added, thus we could recover many of such FNs. However, to keep the compatibility with the previous NYT-FB corpus, we simply discard such examples from the training set (instead of including them as new positive examples). We could match 1,131 new pairs, which are around 1.4% of the total number of the matched pairs in the training set. In overall, 3,373 mentions from the positive examples and 11,818 mentions from negative examples are discarded from the training set.

Relation Type	P%	R%	F1%
company/founders	66.7	11.4	19.5
location/contains	13.5	40.4	20.3
person/company	11.6	60.7	19.5
company/place_founded	20.0	6.7	10.0
person/place_lived	10	20.2	13.46

Table 4.1: Precision and recall of different relation types.

4.3.5 Pipeline Configuration

We use standard NLP tools in our pipeline: we parsed all the sentences using the Charniak parser [32] and tagged the named entities with the Stanford NER toolkit [86] into 4 classes (e.g. Person, Location, Organization and Other). We used SVM-Light-TK[1] for training our classifiers, and employed the one-vs-all strategy for multi-class classification but with some modifications to handle the overlap relations: instead of selecting the class with the highest score assigned by the classifier to sentences, we selected all the labels if the assigned scores are larger than a certain threshold (e.g., 0). Hence, the classifier can select more than one class for each example. We normalize both the tree kernel and the feature vectors.

4.3.6 Parameter Optimization

The SVM accuracy is highly influenced by selecting the suitable values for the cost-factor (option j) and trade-off (option c) parameters. As we mentioned, the dataset is very imbalance thus we tuned the j parameter to outweigh the positive example errors with respect to the negative examples during training. We used 30% of our training set as a development set to optimize the parameters. Then, the best combination of c and j values with the highest F-measure in the development set are used to train the classifier.

[1]http://disi.unitn.it/moschitti/Tree-Kernel.htm

	P%	**R%**	**F1%**
Mintz++	31.28	15.43	20.67
Intxaurrondo et al.	29.79	17.48	22.03
Basic SVM	12.4	7.6	9.5
Our Model	11.3	23.0	15.1
Our Model + filtering	13.2	22.5	16.6

Table 4.2: Comparison of the results for different DS methods

4.3.7 Evaluation

We compared our model with the two recent state-of-the-art algorithms such as: (1) Mintz++ [139], which is an improved version of the original work by [90] and (2) Intxaurrondo et al. [63]. The results for different classes and the overall Micro-average F1 are shown in tables 4.1 and 4.2, respectively. The results show that (i) our model improves the micro-average F1 of the basic RE implementation (basic SVM), i.e., by [170], by more than 7 absolute percent points, i.e., 74% relative; and (ii) applying our simple filtering approach improves our model by 1.5% absolute points. However, our models still outperformed by the state of the art: this is not critical considering that our aim is to build simpler baseline systems.

4.4 Conclusion

In this chapter, we have proposed a standard framework, simple RE models and an upgraded version of NYT-FB for more easily measuring the research progress in DS research.
Our RE model is based on SVMs, can manage overlapping relations and exploit syntactic information and lexical features thanks to tree kernels.
We have also shown that filtering techniques applied to DS data can discard noisy examples and significantly improve the RE accuracy.

5

Instance Weighting for Crowdsourcing QA Task

5.1 Overview

In this chapter, we take into account the disagreement between the crowd annotators as well as their skills in the learning algorithms. For this purpose, we design several instance weighting strategies, which help the learning algorithm to deal with the noise of the training examples, thus producing higher accuracy.

The goal of the first approach is to employ these parameters for building our weighting functions, which aim to reduce the impact of the noisy annotations in learning algorithms. So, we build a passage reranking dataset based on TREC 2002/2003 QA. We use Crowdflowers for carrying out an initial noisy annotation and we have an expert to manually verify and correct incorrect labels. This is an important QA resource that we will release to the research community.

Additionally, the accuracy of our models, e.g., classifiers and search engines, test on such gold standard data establish new baselines, useful for future research in the field. Finally, we conduct comparative experiments on our QA dataset using our weighting strategies. The results show that

(i) our rerankers improve on the IR baseline, i.e., BM25, by 17.47% and 19.22% in MRR and P@1, respectively; and (ii) our weighting strategy improves the best reranker (using no-weighting model) up to 1.47% and 1.85% on MRR and P@1, respectively.

5.2 Crowdsourced Dataset

Initially, we ran a crowdsourcing task on CrowdFlower micro-tasking platform and asked the crowd workers to assign a relevant/not relevant annotation label to the given Q/AP pairs. The crowd workers had to decide whether the given AP supports the raised question or not.

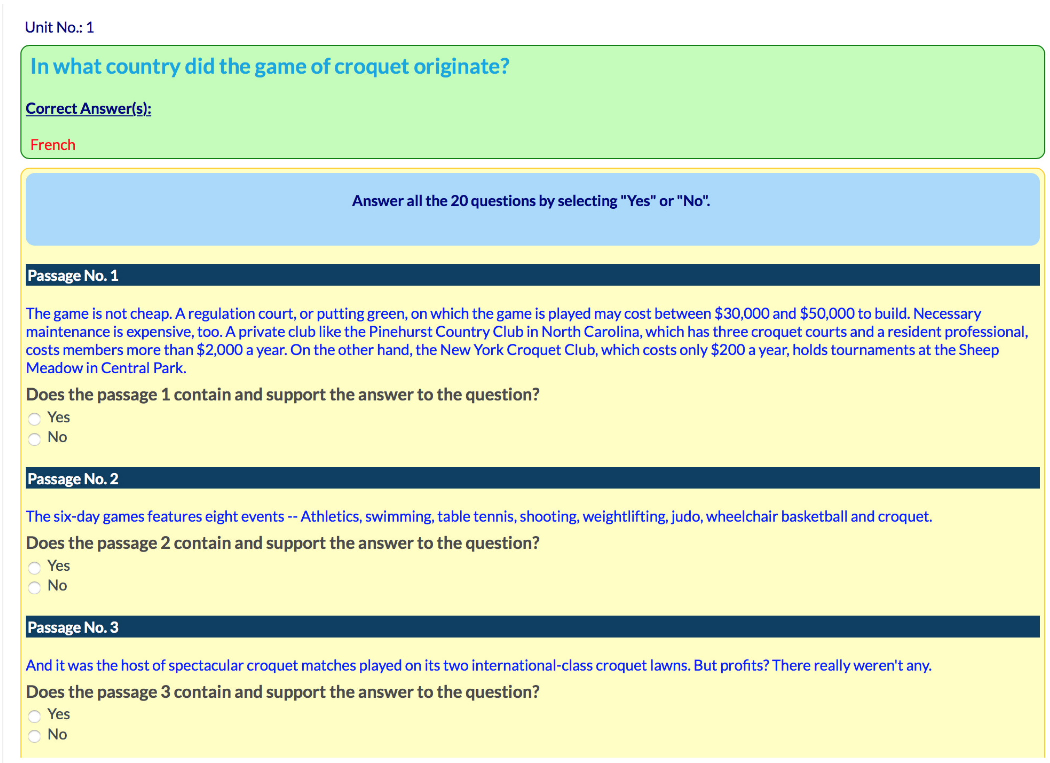
We consider the TREC corpora described in Section 2.3.3 and in particular the first 20 APs retrieved by BM25 search engine for every question. We collect 5 judgments for each AP.

In overall, we crowdsourced 527 questions of the TREC 2002/2003 QA task and collected 52,700 judgments. The number of the participant workers was 108 and the minimum and maximum number of answer passages annotated by a single crowd annotator were 21 and 1,050, respectively.

To obtain an accurate gold standard, we asked an expert to revise the passages labeled by crowd annotators when at least one disagreement was present among the annotations. This *super* gold standard is always and only used for testing our models (not for training). The user interface of the QA crowdsourcing task is illustrated in Figure 5.1.

5.3 Instance Weighting Strategy

In this approach, we define weighing schema for each passage of the training questions. More in detail, each question q is associated with a


Unit No.: 1

In what country did the game of croquet originate?

Correct Answer(s):

French

Answer all the 20 questions by selecting "Yes" or "No".

Passage No. 1

The game is not cheap. A regulation court, or putting green, on which the game is played may cost between $30,000 and $50,000 to build. Necessary maintenance is expensive, too. A private club like the Pinehurst Country Club in North Carolina, which has three croquet courts and a resident professional, costs members more than $2,000 a year. On the other hand, the New York Croquet Club, which costs only $200 a year, holds tournaments at the Sheep Meadow in Central Park.

Does the passage 1 contain and support the answer to the question?

Yes

No

Passage No. 2

The six-day games features eight events -- Athletics, swimming, table tennis, shooting, weightlifting, judo, wheelchair basketball and croquet.

Does the passage 2 contain and support the answer to the question?

Yes

No

Passage No. 3

And it was the host of spectacular croquet matches played on its two international-class croquet lawns. But profits? There really weren't any.

Does the passage 3 contain and support the answer to the question?

Yes

No

Figure 5.1: Crowdsourcing user interface for QA task.

sorted list of answer passages. In turn, each passage p is associated with a set of annotators $\{a_p^1, a_p^2, ..., a_p^k\}$, where a_p^h is the annotator h, $j_p^h \in \{+1, -1\}$ is her/his judgment, and k is the number of annotators per passage. We defined a weighting function, $f(\cdot)$, for scoring the passage p as:

$$f(p) = |\sum_{h=1}^{k} j_p^h W(a^h)|. \tag{5.1}$$

The weighting function consists of a summation of two factors: (i) j_p^h, which indicates the judgment value the annotators, h, have provided for the passage p; and (ii) $W(u)$, which aims at capturing the reliability of the crowd worker u, using the product of three factors:

$$W(u) = P(u)T(u)C(u) \tag{5.2}$$

where *Prior Confidence*, $P(u)$, indicates the prior trust confidence score of the crowd worker, u, provided by the crowdsourcing platform based on the quality of the annotations (s)he has done in the previous tasks. *Task Confidence*, $T(u)$, indicates the total number of annotations performed by the crowd worker u in this task. The score is re-scaled and normalized between (0,1) by considering the maximum and minimum number of annotations the workers have done in this task. *Consistency Confidence*, $C(u)$, indicates the total number of annotation agreements between the annotator u and the majority voting in this task. The score is normalized and re-scaled between (0,1) as well. We use Eq. 5.1 in the optimization function of SVMs:

$$\min \quad \frac{||\vec{w}||^2}{2} + c\sum_i \xi_i^2 f(p_i) \tag{5.3}$$

where $\vec{w}$ is the model, c is the trade-off parameters, ξ_i is the slack variable associated with each training example $\vec{x}_i$, p_i is the passage related to the example x_i (i.e., associated with a constraint), and $f(p_i)$ (Eq. 5.1) assigns a weight to such constraint.

5.4 Experimental Setup

5.4.1 Classifier Feature

We used the rich set of features described in the state-of-the-art QA system [143]. Such features are based on the similarity between question and the passage text: N-gram overlap (e.g., word lemmas, bi-gram, part-of-speech tags and etc.), tree kernel similarity, relatedness between question category and the related named entity types extracted from the candidate answer, LDA similarity between the topic distributions of question and answer passage.

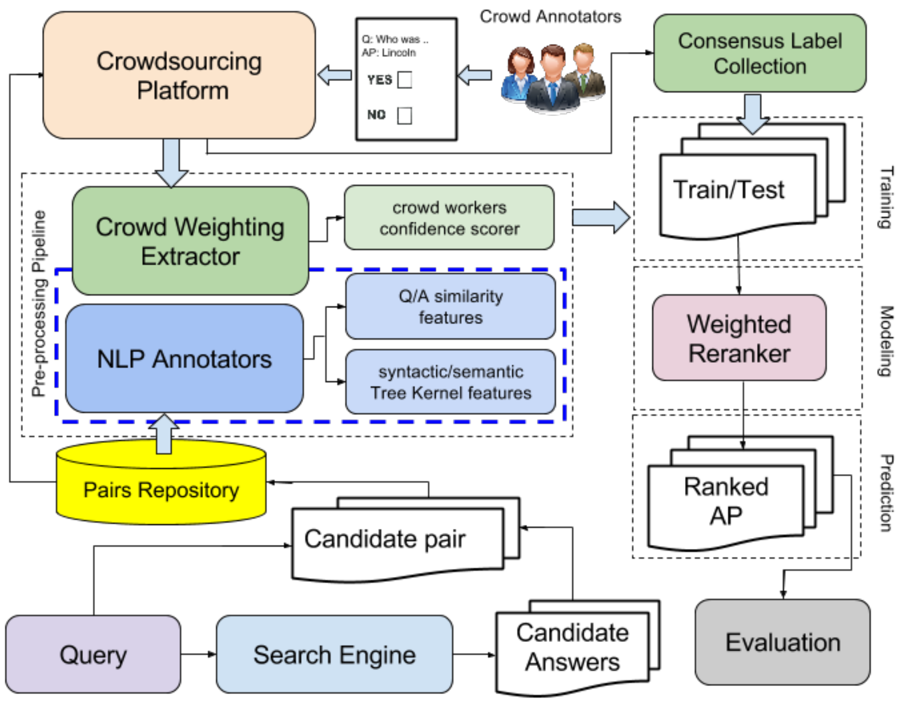


Figure 5.2: Pipeline of QA system. The feature extractor components surrounded by the blue dashed box.

5.4.2 Reranking Model

We used (i) a modified algorithm of SVM-rank [1] using the Eq. 5.3 to train our reranker; (ii) the default cost-factor parameter; and (iii) some other specific values to verify if our results would be affected by different C values. Figure 5.2 illustrates the pipeline of the QA system used in our work.

5.4.3 Baselines

We compared our results with three different baselines, namely:
-BM25: we used Terrier search engine[2], which provides BM25 scoring model to index the answer passages [122]. The APs are extracted from

[1]http://svmlight.joachims.org
[2]http://terrier.org

Model	MRR	MAP	P@1
Baselines			
BM25	41.75 ± 6.56	37.25 ± 4.52	25.57 ± 6.17
RE	57.41 ± 7.31	51.75 ± 6.27	41.38 ± 11.12
CA	57.75 ± 6.77	52.09 ± 5.68	42.94 ± 8.55
Our Weighting Results			
L	58.73 ± 6.88	52.48 ± 6.00	44.12 ± 9.75
P	58.51 ± 5.63	52.07 ± 4.63	43.15 ± 7.32
LP	58.76 ± 6.52	52.60 ± 6.03	44.22 ± 8.72
TC	58.31 ± 5.44	52.09 ± 4.96	42.83 ± 7.69
LTC	58.85 ± 5.85	52.58 ±5.52	43.74 ± 8.50
LPTC	59.22 ± 6.30	52.63 ± 5.96	44.79 ± 8.82

Table 5.1: Results over 5 fold cross validations. Our weighting scheme results are better than the baselines with $p < 0.05$

AQUAINT text corpus and treated as documents. BM25 is used to retrieve 20 candidate answers for each question and rank them by their relevance scores.
-RE (regular expression): we trained a classifier with the noisy annotations produced by labels automatically derived with RE applied to answer keys (no weighting strategy).
-CA (crowd annotations): we train a classier with the same configuration as RE but using majority voting as a source of supervision.

5.5 Weighting Experiments

In these experiments, we used the labels provided by crowd annotators using majority voting for training and testing our models. Most interestingly, we also assign weights to the examples in SVMs with the weighting schemes below:

- Labels Only (L), i.e., we set $P(u) = T(u) = C(u) = 1$ in Eq. 5.2. This means that the instance weight (Eq. 1) is just the sum of the labels j_p^h.

- Prior Only (P): to study the impact of prior annotation skills, we set $C(u) = T(u) = 1$ in Eq. 5.2, and we only use $P(u)$ (crowdflower trust), i.e., we do not account for the sign of annotations, j_p^h.

- Labels & Prior (LP): the previous model but we also used the sign of the label, j_p^h.

- Task & Consistency (TC): we set $P(u) = 1$ such that Eq. 5.2 takes into account both annotator skill parameters for the specific task, i.e., task and consistency confidence, but only in the current task and no sign of j_p^h.

- L & TC (LTC): same as before but we also take into account the sign of the annotator decision.

- LPTC: all parameters are used.

All our results are computed with 5-folds cross validations.
Table 5.1 shows the evaluation of the different baselines and weighting schemes proposed in our work (using the default c parameter of SVMs). We note that: firstly, the accuracy of BM25 is lower than the one expressed by rerankers trained on noisy labels (-15.66% in MRR, -14.5% in MAP, -15.81 in P@1%).

Secondly, although there is some improvement using crowd annotations for training[3] compared to the noisy training labels (RE), the improvement is not significant (+0.34% in MRR, +0.34% in MAP, +1.56% in P@1). This is due to three reasons:

(i) the crowdsourcing annotation suffers from a certain level of noise as well (only 27,350 of the answer passages, i.e., 51.80%, are labeled with “crowd fully in agreement”);

(ii) although the RE labels may generate several false positives, these are always a small percentage of the total instances as the dataset is highly

[3]The test labels are always obtained with majority voting and we removed questions that have no answer in the first 20 passages retrieved by BM25.

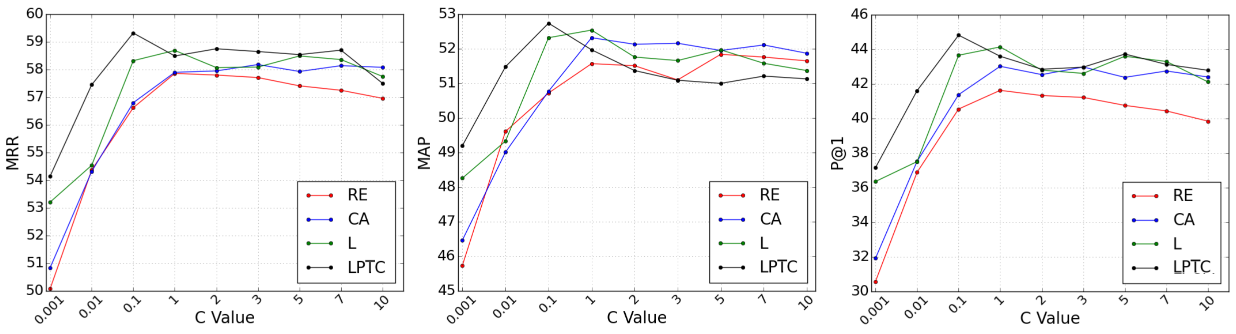


Figure 5.3: The impact of the C values on different models with (LPTC, L) and without (CA, RE)

unbalanced (9,535 negative vs. 1,005 positive examples);

(iii) RE does not generate many false negatives as they are precise.

Thirdly, the table clearly shows the intuitive fact that it is always better to take into account the sign of the label given by the annotator, i.e., LP vs. L and LTC vs. TC.

Next, when we apply our different weighting schema, we observe that the noise introduced by the crowd annotation can be significantly reduced as the classifier improves by +1.47% in MRR, +0.54% in MAP and +1.85% in P@1, e.g., when using LTC & LPTC compared to CA, which does not provide any weight to the reranker.

Finally, as the trade-off parameter, c, may alone mitigate the noise problem, we compared our models with the baselines according to several value of the parameter. Fig. 5.3 plots the rank measures averaged over 5-folds: our weighting methods, especially LPTC (black curve), is constantly better than the baseline, CA, (blue curve) in MRR and P@1.

5.6 Conclusion

Our study shows that we can effectively exploit the implicit information of crowd workers and apply it to improve the QA task.

We demonstrated that (i) the best ranking performance is obtained when the combination of different weighting parameters is used; and (ii) the noise of annotations, present in crowdsourcing data, can be reduced by considering weighting scores extracted from crowd worker performance.
In the future, we will explore better weighting criteria to model the noise that is induced by annotations of crowd workers.

6

Automatic Label Selector (ALS)

6.1 Overview

In this chapter, we aim at learning an statistical model to predict the best label from a set of annotations done by a group of annotators and by quantifying their skills.

We used several parameters in aggregated voting functions which considered as descriptor features for each example to be labeled positive or negative. In addition, we exploit the syntactic information of the examples to represent the structure of the given tasks. It has been done in an automatic fashion without considering any feature engineering process thanks to Tree Kernel (TK). We use such feature representation in Support Vector Machines for learning an Automatic Label Selector (ALS) for assigning the most correct labels out of the annotator labels.

We conducted comparative experiments on our datasets in different setting of features and parameters. The results show that ALS greatly improves on the majoring voting by reducing the annotation error by 7% on the passage reranking task and 20% on the RTE 2.

6.2 Pilot Experiments

Before running the main crowdsourcing task, we evaluated the effect of the initial configurations of the platform on the quality of the collected annotations. We conducted two pilot crowdsourcing experiments with different scenarios:

In the first experiment, we crowdsourced 15 Q/APs and for each Q/AP pairs, judgments were requested. We also set the maximum number of annotations a unique crowd worker can perform to 70 judgments. In overall, 83 crowd workers participated in the task and the maximum number of annotation per individual annotator was 45.

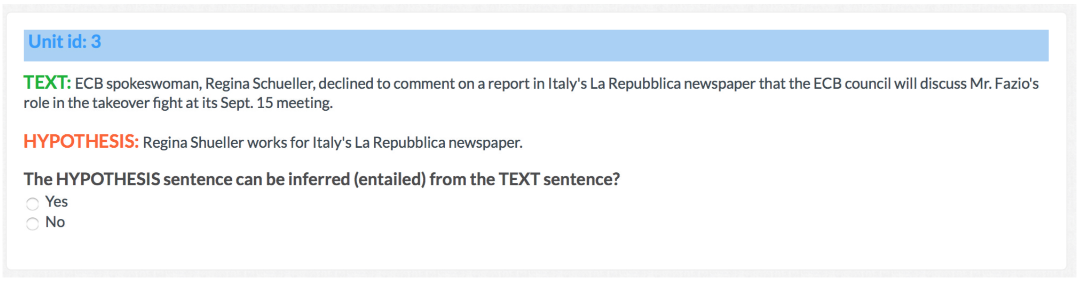
In the second scenario, we removed the constrain of the quota limitation and repeated the data collection procedure. However, the task was finalized by only 37 crowd workers and the maximum number of annotations provided by a annotator was 300.

The intuition behind the idea is:

a crowd worker is more reliable for a given task if (s)he annotates more examples since s(he) has seen more number of gold questions. The statistical test has shown that in the both conditions, the collected sets of annotations have a high level of agreement (0.769) calculated with the Kappa test [31].

6.3 Crowdsourced Datasets

We ran two different crowdsourcing tasks on CrowdFlower micro-tasking platform for QA and the RTE datasets:



Unit id: 3

TEXT: ECB spokeswoman, Regina Schueller, declined to comment on a report in Italy's La Repubblica newspaper that the ECB council will discuss Mr. Fazio's role in the takeover fight at its Sept. 15 meeting.

HYPOTHESIS: Regina Shueller works for Italy's La Repubblica newspaper.

The HYPOTHESIS sentence can be inferred (entailed) from the TEXT sentence?

- Yes
- No

Figure 6.1: Crowdsourcing user interface for RTE task.

6.3.1 QA Task

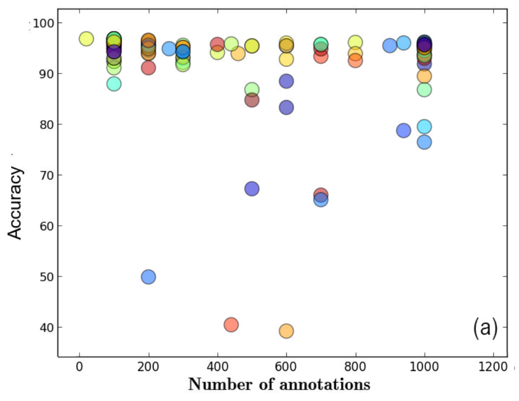
Like the previous experiment, a set of Q/APs pairs are provided and the annotators were asked to assign a relevant/not relevant annotation label to the given pairs. We used exactly the same dataset and the annotations described in Section 5.2. In overall, 527 questions and its associated passage candidates are crowdsouced. As a result, 52,700 judgments from 108 crowd workers are collected and analyzed.

6.3.2 RTE Task

In the second annotation task, we crowsdourced development set from the second challenge on Recognizing Textual Entailment (RTE2) defined by Dagan et al. [43]. The characteristics of this dataset described in the Section 2.4.3. We asked for 5 judgments for each sentence pair. Unlike the TREC corpus, the ground truth labels are provided by the community in advance.

In overall, 4,000 annotations are collected. The number of participants was 82 and the maximum number of sentences annotated by a unique annotator was 99 while there was a annotator with only 8 annotations. Figure 6.1 shows a unit of crowdsourced T/H pair designed in CF platform.

Additionally, we removed the maximum quota of annotations an annotator can perform in the both tasks. We demonstrated that this (i) does not affected the quality of the annotations in Section 6.2; and (ii)



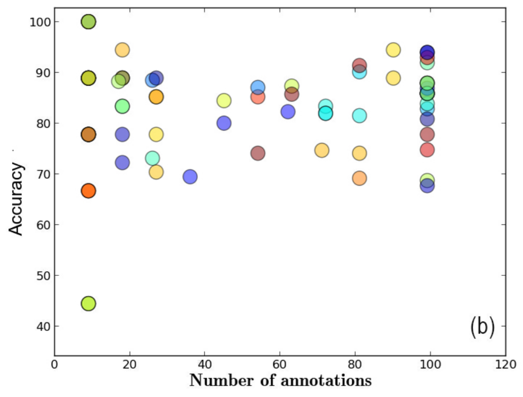


Figure 6.2: Workers accuracy vs. number of annotations: a) QA and b) RTE tasks. Each colored circle is a representative of an annotator.

allows us to collect reliable statistics about the annotators since they can participate extensively to our annotation projects. Eventually, we used this factor as a feature in our artificial labeler classifier.

Figure 6.2 shows the accuracy of annotators versus the number of annotations each worker contributed to the main annotation tasks. The results confirmed that our assumption was correct and removing the quota limitation has not affected the quality of the collected annotations.

6.4 The Automatic Label Selector Framework(ALS)

The model for learning to select the most probable annotator annotation is simple from the learning algorithm perspective: given an example to be labeled, the set of annotators constitute the learning objects whereas the correct labels for the example is the output.

Therefore, the complexity of the approach rely on representing accurately a set of annotator trustability criteria and the task information in a from of features. For this purpose, we first define parameters for characterizing the individual annotator and aggregate them to create features that represent the annotations over the examples. Then, applying Tree Kernels(TK), a set of syntactic features from the examples are extracted to train ALS. The latter represents the hidden syntactic structure of the examples without explicit feature design.

Interestingly, selecting the most relevant substructures useful for the task is assigned to kernel machines themselves. The details of the used TK described in Section 2.1.2.

6.4.1 Annotator Parameters

We define feature function, $f(a)$, for each annotator a, which aims at capturing his/her, using the product of three factors:

$$f(a) = N(a)M(a)C(a) \tag{6.1}$$

where $N(a)$ is the *Cardinality Confidence*, which indicates the total number of annotations performed by annotator a in this task. This is re-scaled and normalized between (0,1) by considering the maximum and minimum number of annotations the annotator have done in this task. $M(a)$ is the *Majority Consistency Confidence*, which indicates the total number of annotations agreement between the annotator a and the majority voting in this task. This is normalized and re-scaled between (0,1) as well. $C(a)$ is the *Prior Confidence*, which indicates the prior trust confidence score of the crowd worker, a, provided by the crowdsourcing platform based on the quality of the annotations (s)he has done in the previous tasks.

6.4.2 Aggregated Features for ALS

Each example x is associated with a set of annotators $\{a_1^x, a_2^x, ..., a_k^x\}$, $l_h^x \in \{+1, -1\}$ is her/his label assigned to x, and k is the number of annotators per example.

We defined an aggregated feature, $F(\cdot)$, for the example x as:

$$F(x) = \sum_{h=1}^{k} l_h^x f(a_h). \tag{6.2}$$

This consists of a summation of two factors: (i) l_h^x, which indicates the judgment value the annotators, h, have provided for the passage x; and (ii) the annotator feature $f(a_h)$ defined by Eq. 6.1. $F(x)$ suggests that each ASL aggregated feature is basically a majority voting score in which each annotator vote is weighted according to one specific feature.

By changing the value of $f(a_h)$, we can generate several features as for example outlined below:

- Labels Only (L), assign $N(a) = M(a) = C(a) = 1$, i.e., the majority voting.

- Prior Only (P): to specifically account for the crowd workers annotation skills, we set $N(a) = M(a) = 1$ in Eq. 6.1, i.e., we only use $C(a)$ (crowdflower trust) for weighing our instances.

- Cardinality & Majority (CM): we weight the agreement across crowd workers, thus we utilized the majority consistency confidence $M(a)$ along with $N(a)$ for normalization purpose $C(a) = 1$.

- Pr: we set $f(a)$ as the Precision of a in her/his annotation task of the current dataset.

- Rc: we set $f(a)$ as the Recall of a in her/his annotation task of the current dataset.

- T$_i$: this is the only non-aggregated feature: we use the set of annotator IDs annotating the example.

6.4.3 Task Parameters

We exploit the syntactic information of the examples using Tree Kernel (TK) in order to get the advantage of high number of syntactic features in ALS learner [36].

More in details: we applied Subset Trees (SSTs) proposed by [96] and count the number of common tree fragments between two trees T_1 and T_2. The kernel function $K(T_1, T_2)$ defined as :

$$K(T_1, T_2) = \sum_{n_1 \in N_{T1}} \sum_{n_2 \in N_{T2}} \Delta(n_1, n_2) \tag{6.3}$$

where N_{T1} and N_{T2} are a set of T_1's and T_2's nodes, respectively and $\Delta(n_1, n_2)$ are number of common fragments rooted in the n_1 and n_2 nodes. In QA task, T_1 and T_2 are associated with question/passage parse trees respectively. Also, in RTE task, the syntactic properties of Text/Hypothesis pairs are exploit. In the both tasks, we parsed the examples using Stanford parser [86].

6.5 Experimental Setup

6.5.1 ALS Experiments

To generate the feature vectors of Q/AP pairs annotated by a set of annotators, as it is mentioned in Section 4.1, the aggregation of some parameter weights of the annotators has used as a feature. Basically these features can reveal the performance of the annotators over the time by considering the accuracy of annotators in the previous tasks and their precision level in the current task. As a result, the ALS learns to model the annotation decision of annotators for a specific instance and predict the label of unseen instances based on training data.

We evaluated the accuracy of ALS on two different NLP tasks, namely, QA and RTE tasks. for the both tasks, we used 5 fold cross validations strategy to evaluate the accuracy of the classifier.

Table 6.1 shows the contribution of the different features and weighting schema proposed in this paper (ablation test). The final results presented in the last column of the table shows that ALS classier performs better that majority voting (baseline) and decreases the accuracy errors by 0.71% and 3.5% in QA and RTE tasks respectively. In addition, by combining the annotators characteristics features and the syntactic features derived from the tasks, the accuracy of automatic labeler selector increases up to 0.12% and 1.13% in the QA and RTE tasks.

6.5.2 Trade-off Parameter

The number of features are small and the feature space is sparse. Although the instances distribution for the both positive and negative classes are balance (vs) in RTE dataset; however, the QA dataset is highly skewed to negative examples population (1005 vs 9535).

We studied the impact of the trade-off value C, in the both tasks by evaluation the accuracy with several C values. Fig. 6.3 shows the accuracy of ALS classifier in the both tasks when the C value is increased.

6.5.3 Learning Curves

Fig. 6.4 plots the learning curves in the both QA and RTE tasks which indicates that by small increments in supervision of the both tasks (e.g., 84 training examples in QA and 128 training examples in RTE task), the ALS classifier outperforms the unsupervised baseline(majority voting) significantly.

	QA Task				RTE Task			
Feature	Precision	Recall	F1-Measure	Accuracy	Precision	Recall	F1-Measure	Accuracy
Baseline	86.45 ± 5.49	95.81 ± 2.69	90.85 ± 3.98	98.33 ± 0.60	84.91± 4.03	96.77 ± 1.31	90.40 ± 2.28	89.75 ± 2.44
L	96.14 ± 2.24	91.98 ± 3.24	94.00 ± 2.38	98.98 ± 0.30	92.01± 3.23	92.01 ± 1.18	91.98 ± 1.63	92.00 ± 1.61
M	95.93 ± 2.38	92.26 ± 3.36	94.05 ± 2.58	98.99 ± 0.33	92.98 ± 2.41	92.96 ± 1.64	92.95 ± 1.44	93.00 ± 1.20
C	95.93 ± 2.38	92.17 ± 3.25	94.00 ± 2.53	98.98 ± 0.32	92.50 ± 3.09	92.95 ± 5.91	92.60 ± 2.88	92.62 ± 2.77
P	96.05 ± 2.20	91.98 ± 3.24	93.95 ± 2.33	98.97 ± 0.29	92.09 ± 3.0	93.44 ± 6.18	92.62 ± 2.77	92.62 ± 2.63
Pr	94.17 ± 3.11	92.92 ± 3.77	93.51 ± 3.0	98.89 ± 0.40	91.58 ± 3.16	92.34 ± 2.27	91.89 ± 0.74	91.87 ± 0.76
Rc	95.43 ± 2.48	92.59 ± 3.37	93.97 ± 2.52	98.97 ± 0.32	92.73 ± 2.49	93.17 ± 1.57	92.94 ± 1.72	93.00 ± 1.35
Ti	93.86 ± 4.70	92.50 ± 2.62	93.16 ±3.53	98.83 ± 0.50	91.62 ± 3.39	92.57 ± 1.90	92.05 ± 1.75	92.00 ± 1.89
TK	95.36 ± 3.61	92.37 ± 1.01	93.82 ± 2.16	98.92 ± 0.33	92.45 ± 1.43	91.86 ± 4.57	92.08 ± 1.80	92.12 ± 1.80
Our Model	96.68 ± 2.81	92.41 ± 0.95	94.49 ± 1.80	99.04 ± 0.32	92.58 ± 2.86	93.96 ± 1.28	93.24 ± 1.59	93.25 ± 1.35

Table 6.1: Ablation test for Q/A and RTE tasks.The results are estimated over 5 fold cross validation

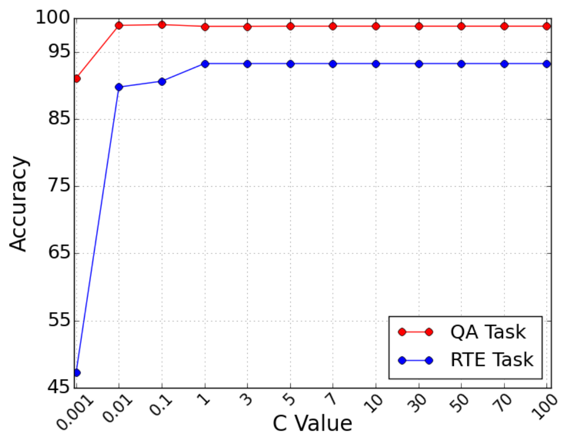


Figure 6.3: ALS classifier accuracy in QA and RTE tasks setting several C values.

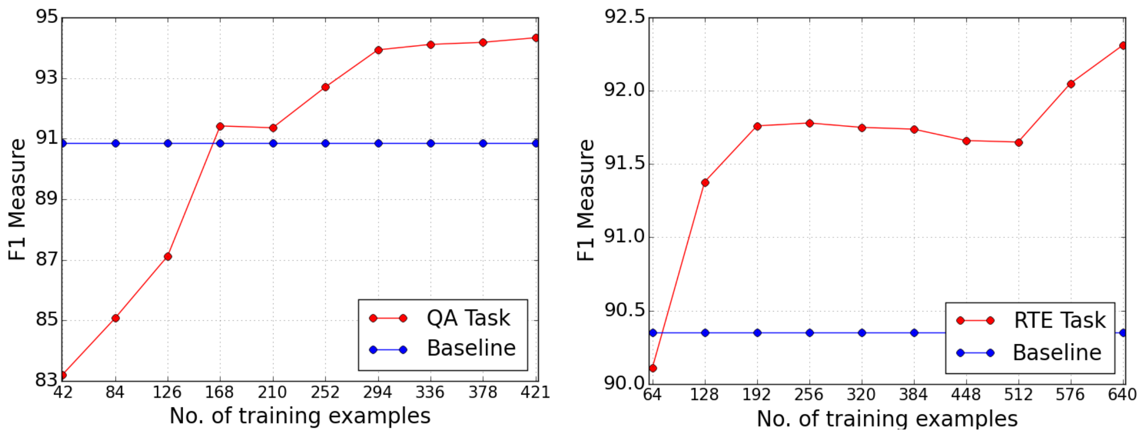


Figure 6.4: Learning curves under different number of training examples.

6.5.4 Quality of Annotations

We investigated the relationship between the number of labels and the overall quality of the labeling using multiple annotators. Eventually, we compared the quality of the majority voting with ALS when the number of annotators increases.

Fig. 6.5 demonstrates the analytic relationship between the number of annotators and the quality of the collected labels. As expected, increasing the number of annotators leads to produce labels with the higher quality. Moreover, the plots shows that in any number of annotators aggregation, ALS performs better that majority voting. Note that, for the sake of simplicity and avoid of biased evaluation, we keep the accuracy fixed when the number of annotators was even in majority voting.

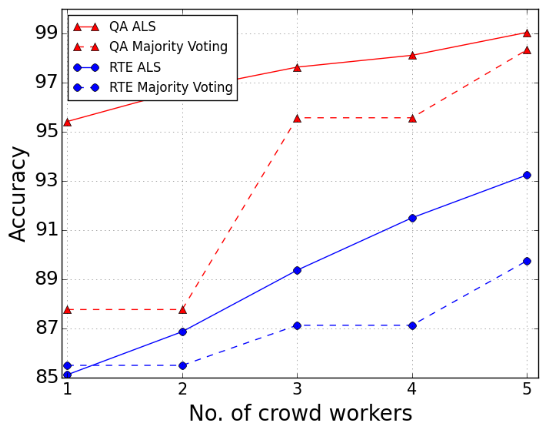


Figure 6.5: Improvement in tasks accuracy by increasing the number of annotators.

6.6 The Impact of ASL on Target Tasks

In the second set of experiments, we studied the effect of labels quality on target task performance. Hence, we used the ALS predicted labels to train the classifier for passage re-ranking in QA and recognizing textual entailment, two important NLP tasks. Then, the results compared with unsupervised and fully supervised models, when classifiers were trained with majority voting and gold standard labels respectively.

6.6.1 Classifier Features

We used the rich set of features described in the state-of-the-art QA system [143]. Such features are based on the similarity between question and the passage text: N-gram overlap (e.g., word lemmas, bi-gram, part-of-speech tags and etc.), tree kernel similarity, relatedness between question category and the related named entity types extracted from the candidate answer, LDA similarity between the topic distributions of question and answer passage.

For RTE task, we used a set of annotation components to generate the syntactic and lexical features [77], namely; Tokenizer, POS tagger, dependency parser. Moreover, several external knowledge resources such as Wikipedia and Wordnet are used to recognize cases where T and H use different textual expressions.

6.6.2 Classifier Models

For passage reranking (QA task), we used (i) SVM-rank[1] to train our rerankers; (ii) the default cost-factor parameter; and (iii) some other specific values to verify if our results would be affected by different C values.

For the second task (e.g. RTE), we used fixed weight token edit distance [84], which is a token-based distance algorithm, with edit operations defined over sequences of tokens of T and H. We ran Excitement Open Platform (EOP) pipeline version 1.2.3 [2] where Entailment Decision Algorithm(EDA) optimized on RTE2 development set and evaluated on RTE2 test set.

6.6.3 Baselines

We evaluated the accuracy of our approach, by training a QA classifier supervised by ALS predicted labels with three different baselines namely:

- BM25: we used Terrier search engine[3], which provides BM25 scoring model to index the answer passages [122]. The APs are extracted

[1] http://svmlight.joachims.org
[2] http://hltfbk.github.io/Excitement-Open-Platform/
[3] http://terrier.org

from AQUAINT text corpus and treated as documents. BM25 is used to retrieve 20 candidate answers for each question and rank them by their relevance scores.

- RE (Regular Expression): we trained the QA classifier with the noisy annotations produced by labels automatically derived with RE applied to answer keys.
- MV (Majority Voting): we trained the QA classier with the same configuration as above but using majority voting of crowdsourced annotations as a source of supervision.
- GS (Gold Standard) and finally, as a upper bound of the task, we compared the QA classifier results when it was trained with gold standard labels.

In the RTE task, the last two baselines mentioned above are considered as lower and upper bounds of the task respectively.

6.6.4 Evaluation Metrics

Like the previous experiment, we used the same set of metrics mentioned in Section 2.5, to evaluate the performance of the classifier for QA tasks: the Mean Reciprocal Rank (MRR), which computes the reciprocal of the rank at which the first relevant passage is retrieved, Precision at rank 1 (P@1), which reports the percentage of question with the correct answer at rank 1, and Mean Average Precision (MAP), which measures the average of precision of the correct passages appearing in the ranked AP list.

All our results are computed with 5-folds cross validations, thus the above metrics are averaged over 5 folds. In the RTE task, the standard evaluation metrics, e.g., Precision (Pr), Recall (Rec), F1 score and the accuracy is considered.

QA Task			
Model	MRR	MAP	P@1
BM25	41.75 ± 6.56	37.25 ± 4.52	25.57 ± 6.17
RE	57.41 ± 7.31	51.75 ± 6.27	41.38 ± 9.12
MV	57.75 ± 6.77	52.09 ± 5.68	42.94 ± 8.55
ALS	58.93 ± 4.17	52.54 ± 4.07	43.83 ± 5.06
GS	60.25 ± 4.38	52.92 ± 4.41	45.69 ± 5.57

Table 6.2: The results over 5 fold cross validation

RTE Task							
	ENTAILMENT			NONENTAILMENT			
Model	Precision	Recall	F1	Precision	Recall	F1	Accuracy
MV	51.94	96.75	67.59	76.36	10.50	18.46	53.62
ALS	55.51	74.25	63.52	61.13	40.50	48.72	57.37
GS	54.79	85.75	66.86	67.24	29.25	40.76	57.50

Table 6.3: The results of RTE Task evaluated per class (entailment and non-entailment). The last column is the macro average accuracy of the both classes.

6.6.5 Experimental Results

Table 6.2 shows the evaluation of the different baselines and We note that: firstly, the accuracy of BM25 is lower than the one expressed by rerankers trained on noisy labels (-15.66% in MRR, -14.5% in MAP, -15.81% in P@1).

Secondly, although there is some improvement using crowd annotations for training[4] compared to the noisy training labels (RE), the improvement is not significant (+0.34% in MRR, +0.34% in MAP, +1.56% in P@1).

Finally, training the passage reranker with ALS predicted labels, improves the MMR and P@1 up to 1.18% and 0.89% respectively compared to majority voting (MV).

The results of RTE classifier described in Table 6.3. The accuracy of the task increases significantly by 3.75% when ALS labeler output used as

[4]The test labels are always obtained with majority voting and we removed questions that have no answer in the first 20 passages retrieved by BM25.

a source of supervision in RTE task.

Although, the trained system with consensus labels has a higher recall in entailment class, it suffers from low recall in non-entailment class extensively (entail. 22.5% vs. non-entail. -30%). We observed that the annotators have tended to assign more positive labels (entailment) to the examples incorrectly (false positive: 69 vs. false negative: 13). It would due to caused by difficulty of the task, e.g., high similarities between T and H in various pairs or low skill level of annotators in the given task.

However, the ALS classifier avoids of majority bias interpretation and behaving more like the experts(lower variance in precision and recall compared to GS). It was feasible by considering the annotators characteristics aggregation and the examples difficulties together to predict the most probable correct labels.

6.7 Conclusion

In this work, we proposed a learning framework (namely, ALS) to select the best labels out of a set of annotations provided by crowd workers for QA and RTE tasks. This is made possible by taking into account to model the both users and the task jointly. From the crowd workers side, we extracted the metadata of the crowd workers implicitly provided in the users profile and also their performance in the assigned task.
From the task side, the syntactic structural of the tasks presented in a form of feature descriptor using Tree Kernels to train the ASL classifier as well. We show that ASL classifier can improve the both tasks by providing more precise training labels compared to automatic generated noisy labels or the majority voting.

7 Autonomous Crowd Training for RE

7.1 Overview

In this chapter, instead of using gold standard, we propose to alternatively use *silver standard*, i.e., a high-quality automatic annotated data, to train the crowd workers. Specifically, we introduce a self-training strategy for crowd-sourcing, where the workers are trained with simpler examples (which we assume to be less noisy) first and gradually are presented with more difficult ones. This is biologically inspired by the common human process of gradual learning, starting with the simplest concepts.

Moreover, we propose an iterative human-machine co-training framework for the task of RE. The main idea is that a subset of “less-noisy” examples is automatically selected by an automatic system in each iteration, and for training the annotators only using this subset. The educated crowd workers can then provide higher quality annotations, which the system can use in the next iteration to improve the quality of its classification (See Fig. 7.1).

In other words, this cycle gradually improves both system and human annotators. This is in line with the studies in human-based computational

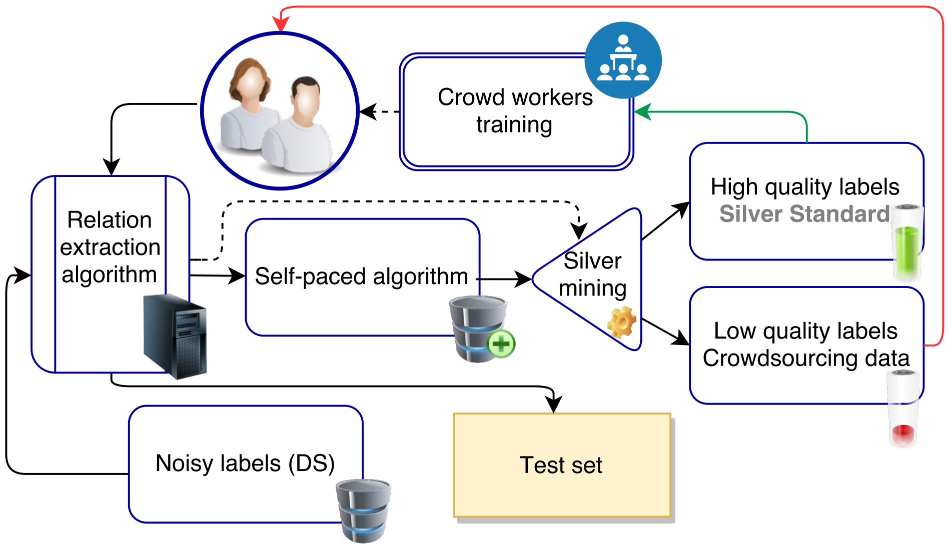


Figure 7.1: Human-machine co-training framework

approaches which showed that the crowd intelligence can effectively alleviate the drifting problem in auto-annotation systems [137, 123].

7.2 Self-Crowdsourcing Training

In this section we first explain, our proposed method for automatically identifying high-quality examples (i.e., Silver Standard) to train the crowd workers and collect annotations for the lower-quality examples. Then, we explain the scheme designed for crowd worker training and annotation collection.

7.2.1 Silver Standard Mining

The main idea of our approach to self-crowdsourcing training is to use classifier's score for gradually training the crowd workers, such that the examples and labels associated with the highest prediction values (i.e., the most reliable) will be used as silver standard.

More in detail, our approach is based on a noisy-label dataset, DS, whose labels are extracted in a distant supervision fashion and CS a dataset to be labeled by the crowd. The first step it to divide CS into three parts CS_I, which is used to create the instruction for the crowd workers, CS_Q, which used for asking questions about sentence annotations, and CS_A, which is used to collect the labels from annotators, after they have been trained.

To select CS_I, we train a classifier C on DS, and then used it to label CS examples. In particular, we used MultiR framework [57] to train C, as it is a widely used framework for RE. Then, we sort CS in a descending order according to the classifier prediction scores and select the first N_i elements, obtaining CS_I.

Next, we select the N_q examples of $CS \setminus CS_I$ with highest score to create the set CS_Q. Note that the latter contains highly-reliable classifier annotations but since the scores are lower than for CS_I examples, we conjecture that they may be more difficult to be annotated by the crowd workers.

Finally, CS_A is assigned with the remaining examples, i.e., $CS \setminus CS_I \setminus CS_Q$. These have the lowest confidence and should therefore be annotated by crowd workers. N_i and N_q can be tuned on the task, we set both to 10% of the data(See algorithm 1).

7.2.2 Training Schema

We conducted *crowd worker training* and *annotation collection* using the well-known Crowdflower platform[1]. Given CS_I and CS_Q (See Section 7.2.1), we train the annotators in two steps:

(i) **User Instruction**: first, a definition of each relation type (borrowed from TAC-KBP official guideline) is shown to the annotators. This initial

[1]www.crowdflower.com

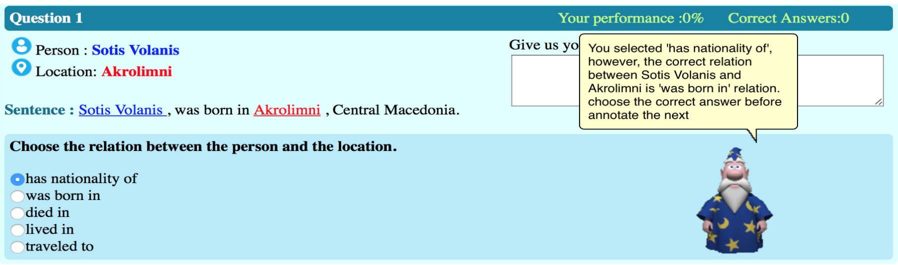
Algorithm 1 Collaborative Crowdsourcing Traning

```
**Input:** $DS$, $CS$, $N_i$, $N_q$, $MaxIter$
**Output:** Trained classifier $C_t$
$C_0 \leftarrow$ Train *MultiR* on $DS$
**For** $t := 1$ **to** $MaxIter$:
    $P \leftarrow \emptyset$
    For each $E \in CS$:
        Compute $(E_{relation}, E_{score})$ using $C_{t-1}$
        $P \leftarrow P \cup \{(E_{relation}, E_{score})\}$
    $CS_{sorted} \leftarrow$ Sort $CS$ using the scores $E_{score}$ in $P$
    $CS_I \leftarrow N_i$ topmost elements in $CS_{sorted}$
    $CS_Q \leftarrow N_q$ topmost elements in $\{CS_{sorted} \setminus CS_I\}$
    $CS_A \leftarrow$ remaining elements in $\{CS_{sorted} \setminus CS_I \setminus CS_Q\}$
    *User Instruction* using $CS_I$
    *Interactive QA* using $CS_Q$
    $T_{CS} \leftarrow$ *Crowdsourcing* $CS_A$
    $C_t \leftarrow$ Train *MultiR* on $\{DS \cup T_{CS}\}$
```

training step provides the crowd workers with a big picture of the task. We then train the annotators showing them a set of examples from CS_I (see Fig. 7.3). The latter are presented in order of difficulty level. The ranked list of examples provided by self-training strategy facilitates the gradual educating of the annotators [104]. This give us a benefit of training the annotators with any level of expertise. It is a crucial property of crowdsourcing, where we have absolutely no clue about the workers' expertise in advance.

(ii) **Interactive QA**: after the initial step, we challenge the workers in an interactive QA task with multiple-choice questions over the sentence annotation (see Fig. 7.2). To accomplish that, we adapted an interactive java-based agent [2] that can provide feedback for crowd workers: it corrects their mistakes by knowing their given answer and also the correct answer provided by the classifier. Then, the feedback is enriched with a shallow level of rule-based reasoning to help the crowd workers revise their mistakes. Note that: (a) To have a better control of the worker training, we performed a selection of the sentences in CS_Q to be used for questioning

[2]https://www.smore.com/clippy-js



Question 1 Your performance :0% Correct Answers:0

Person : Sotis Volanis
Location: Akrolimni

Sentence : Sotis Volanis , was born in Akrolimni , Central Macedonia.

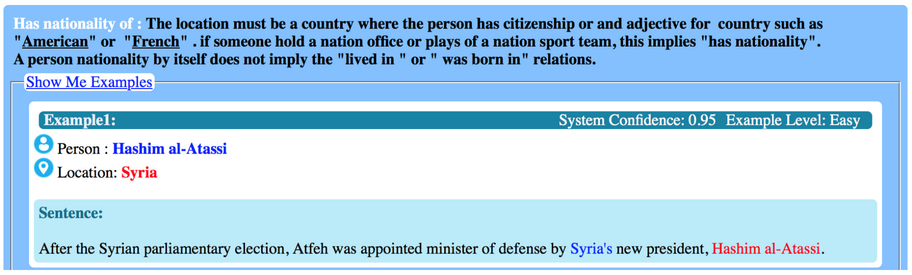
Give us yo

You selected 'has nationality of', however, the correct relation between Sotis Volanis and Akrolimni is 'was born in' relation. choose the correct answer before annotate the next

Choose the relation between the person and the location.

[✓ has nationality of]
was born in
died in
lived in
traveled to

Figure 7.2: User Interface of crowd worker training: interactive QA phase



Has nationality of : The location must be a country where the person has citizenship or and adjective for country such as "American" or "French" . if someone hold a nation office or plays of a nation sport team, this implies "has nationality". A person nationality by itself does not imply the "lived in " or " was born in" relations.

Show Me Examples

Example1: System Confidence: 0.95 Example Level: Easy

Person : Hashim al-Atassi
Location: Syria

Sentence:

After the Syrian parliamentary election, Atfeh was appointed minister of defense by Syria's new president, Hashim al-Atassi.

Figure 7.3: User Interface of crowd worker training: instruction phase

in a category-wise fashion. Meaning that, we select the subsets of examples for each class of relation separately. We observed in practice that initially a lot of examples are classified as “No Relation”. This is due to a difficulty of the task for the DS-based model. Thus, we used them in CS_A.

7.3 Experimental Setup

In this section, we first introduce the details of the used corpora, then we explain the feature extraction and RE pipeline and finally the experiments in details and discuss the results is presented.

7.3.1 Corpora

We used TAC-KBP newswires, one of the most well-known corpus for RE task (See Section 2.2.3). As DS, we selected 700K sentences automatically annotated using Freebase as an external KB. We used the active learning framework proposed by Angeli [14] to select CS. This allowed us to select the best sentences to be annotated by humans (sampleJS). As a result, we obtained 4,388 sentences. We divided the CS sentences in CS_I, CS_Q and CS_A, with 10%, 10% and 80% split, respectively. We requested at least 5 annotations for each sentences. Similarly to [81], we restricted our attention to 5 relations between *person* and *location*[3].

For both DS and CS, we used the publicly available data provided by [81]. Ultimately, 221 crowd workers participated to the task with minimum 2 and maximum 400 annotations per crowd worker.

To evaluate our model, we randomly selected 200 sentences as test set and had domain expert manually tagging them using TAC-KBP annotation guidelines.

7.3.2 Relation Extraction Pipeline

We used the relation extractor, MultiR [57] along with lexical and syntactic features proposed by Mintz et al. [90] such as: (i) Part of Speech (POS); (ii) windows of k words around the matched entities; (iii) the sequences of words between them; and (iv) finally, dependency structure patterns between entity pairs. These yield low-recall as they appear in conjunctive forms but at the same time they produce a high precision.

7.3.3 Experimental Results

In the first set of experiments, we verified the quality of our silver standard set used in our self-training methods. For this purpose, we trained MultiR on CS_I, CS_Q and CS_A and evaluate them on our test set. Figure 7.4 illustrates the results in terms of precision, recall and F1 for each partition separately. They suggest that, the extractor trained on CS_I and CS_Q is significantly better than the one trained on the lower part of the CS, i.e., data CS_A, even if the latter is much larger than the other

[3] *Nationality, Place-of-birth, Place-of-resident, Place-of-death, Traveled-to*

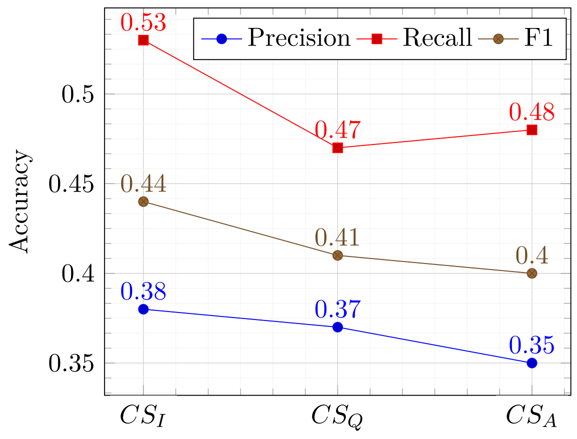


Figure 7.4: Accuracy of different CS partitions

two (80% vs. 10%).

In the next set of experiments, we evaluated the impact of adding a small set of crowdsourced data to a large set of instances annotated by Distant Supervision. We conducted the RE experiments in this setting, as this allowed us to directly compare with Liu et al. [81]. Thus, we used CS_A annotated by our proposed method along with the noisy annotated DS to train the extractor. We compared our method with (i) the *DS-only* baseline (ii) the popular active learning based method (a.k.a., *SampleJS* [14]) and also (iii) the state of the art, *Gated Instruction* (GI) strategy [81].

We emphasize that the same set of examples (both DS and CS) are used in this experiment and just replaced the previous annotations with the annotations collected using our proposed framework.

Note that in the sampleJS baseline, the annotations have been collected through crowdsourcing, but without any explicit user training stage. The results are shown in Table 1. Our method improves both the *DS-only* and the *SampleJS* baselines, in F1 by 5% and 4% respectively. These improvements clearly confirm the benefits of both the crowdsourcing and the crowd worker training in the RE task respectively.

Additionally, our model shows just 3% lower in F1 than the GI method. In both our method and GI, the crowd workers are trained before enrolling

Model	Pr.	Rec.	F1
DS-only	0.43	0.52	0.47
SampleJS [1]	0.46	0.51	0.48
Gated Instruction [6]	0.53	0.57	0.55
Our Method	0.50	0.54	0.52

Table 7.1: Evaluation of the impact of the CS_A label quality in the RE task.

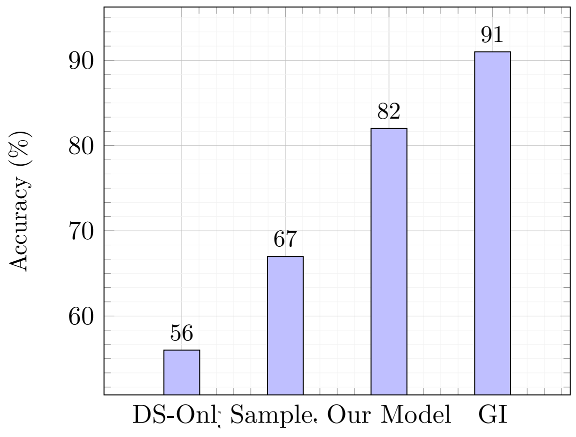


Figure 7.5: Crowd workers annotation accuracy

in the main task. However, GI trains annotators using a Gold Standard, which involves a higher level of supervision with respect to our method. This suggests that our Self-training method is potentially effective and an inexpensive alternative to GI.

We also analyzed the accuracy of the crowd workers in terms of the quality of their annotations. For this purpose, we randomly selected 100 sentences from CS_A and then had them manually annotated by an expert. We compared the accuracy of the annotations collected with our proposed approach with those provided by DS-only baseline and the GI method.

Figure 7.5 shows the results. The annotations performed by annotators trained with our method are just slightly less accurate than the annotations produced by annotators trained with GI. These results are inline with our previous results showing the effect of quality of annotations on the performance of the final RE task.

7.4 Conclusion

In this chapter, we have proposed a self-training strategy for crowdsourcing, as an effective alternative to train annotators with Gold Standard.
Our experimental results show that the annotation carried out with our method is accurate for the high-level semantic task of RE. Such property suggests that training annotators can be a replacement for popular consensus-based filtering scheme.
Our method achieves this goal through an inexpensive training procedure. Analyzing the capability of our method for other fine-grained tasks is surely an interesting direction of future work.

8 Conclusion

Research to date has shown that crowdsourcing is slowly revolutionising NLP research by significantly reducing the cost of acquiring linguistic resources, as well as by directly supporting algorithms and their evaluation. This has come at the expense of new, more complex resource creation methodologies, in particular for contributor management, result aggregation and quality control.
As it is described in Chapter 4, we made contributions to create a benchmark for Relation Extraction Task using Distant Supervison [2, 3] and also introducing two approaches in the Chapters 5 and 6 for improving the accuracy of classifiers in various NLP tasks namely: Question Answering and RTE [4, 1] when the training labels are collected through crowdsourcing.
Also we have proposed a self-training strategy for training the crowd workers before enrolling into the main task without using any gold standard questions. Such property suggests that training annotators can be a replacement for popular consensus-based filtering scheme [5]. We believe that our contributions leave a significant footprint in the domain of quality control in micro-task crowdsourcing for NLP tasks.

We itemize our contributions in this thesis briefly:

- we proposed a standard framework, simple RE models and an upgraded version of NYT-FB for more easily measuring the research progress in DS research.

- we proposed a framework to exploit the implicit information of the crowd workers and apply it to improve the QA task considering weighting scores inferred from crowd worker performance.
- we proposed a statistical learner namely, ALS to select the best labels among the annotations generated by a group of annotators and by considering the implicit information of crowd workers and the given task jointly as a features set.
- we proposed a self-training strategy for crowdsourcing, as an effective alternative to train annotators without gold standard examples.

8.1 Limitations and Future Work

Our work has a set of limitations, which we consider as areas for future improvement. These limitations are discussed below.

Platforms: We conducted our experiments on CrowdFlower. However, still there is a variety of other platfroms such as Amazon Mechanical Turk [1] and CickWorker [2]. Different platforms provide different metadata of the crowd workers we used partially in our experiments. Moreover, the different platforms, attract different demographics of the crowd resulting in different workers behavior. Each platform provides a certain level of freedom for requesters (Task Designer) to design the task, define quality control mechanism and select the workers.
Therefore the reported results and conclusions we identified in our experiments might be not completely valid for other crowdsourcing platforms.

[1] https://www.mturk.com/mturk/welcome
[2] https://www.clickworker.com

Task types: We conducted our experiments based on several NLP tasks (e.g. relation extraction, question answering and recognizing textual entanglements). Crowds workers acting differently annotating different tasks. Thus, the results of the proposed methods in this thesis could be different in new tasks. The effectiveness of our methods for other NLP tasks have not investigated yet and left as a future work.

Dataset size: We performed our experiments with small datasets in QA and RTE tasks. Moreover we collected the minimum number of judgments per example to estimate the majority voting (e.g. between 3 and 5). This is not due to limitation of our methods but rather financial constrains. We believe that having thousands of data units, could show the robustness of our methods, However, the results might be vary with a small variance with what we have reported in this thesis. Generally speaking, scaling up is still a major challenges, as are the following challenges and future trends, which are valid for crowdsourcing in science as a whole.

8.2 Final Remarks

Linguistic resources such as tagged corpora or lexicons are core to the development of the NLP field. Indeed, over the past ten years, NLP research has been driven forward by a growing volume of annotated corpora. These corpora have been used widely to train NLP algorithms and also for domain adaptation. In addition, they make the algorithms comparable as well as let the experimental setups replicable among the scientific researches.

Traditional expert-driven corpus creation methodologies tend to be very expensive to implement in terms of annotation time and price per annotation. Crowdsourcing also changed significantly some scientific practices in NLP. This phenomena helps the community to ask for bigger and more complex high-quality data but still harnesses the linguistic knowledge of the crowd as a source of supervision. However, involving human has its own consequences and expenses as well. This has come at the expense of new, more complex resource creation methodologies, in particular for contributor management, result aggregation and quality control. This is what we presented as a big picture in this thesis.

From a different perspective, the crowd has an ability to learn the task and improve themselves if they have trained in advance. Training mechanisms ensure that the selected contributors understand the task at hand and acquire a basic skill for performing it. We believe that it is a alternative solution for complex pre/post processing quality control mechanism.

Bibliography

[1] Azad Abad and Alessandro Moschitti. Learning to automatically select the best labels from crowd annotators.

[2] Azad Abad and Alessandro Moschitti. Creating a standard for evaluating distant supervision for relation extraction. In *The First Italian Conference on Computational Linguistics*, CLIC '14, 2014.

[3] Azad Abad and Alessandro Moschitti. Distant supervision for relation extraction using tree kernels. In *6th Italian Information Retrieval Workshop*, IIR '14, 2014.

[4] Azad Abad and Alessandro Moschitti. Taking the best from the crowd:learning question passage classification from noisy data. In *The Fifth Joint Conference On Lexcial and Computational Semantics*, 2016.

[5] Azad Abad, Moin Nabi, and Alessandro Moschitti. Self-crowdsourcing training for relation extraction. In *55th annual meeting of the Association for Computational Linguistics*, ACL '17.

[6] Eugene Agichtein, Walt Askew, and Yandong Liu. Combining lexical, syntactic, and semantic evidence for textual entailment classification. In *TAC*. NIST, 2008.

[7] Eugene Agichtein, Carlos Castillo, Debora Donato, Aristides Gionis, and Gilad Mishne. Finding high-quality content in social media. In *Proceedings of the 2008 International Conference on Web Search and*

Data Mining, WSDM '08, pages 183–194, New York, NY, USA, 2008. ACM.

[8] Eugene Agichtein and Luis Gravano. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 85–94. ACM, 2000.

[9] David Ahn, Steven Schockaert, Martine De Cock, Etienne Kerre, et al. Supporting temporal question answering: Strategies for offline data collection. In *Proceedings of the 5th International Workshop on Inference in Computational Semantics. Buxton, UK: ACL*, pages 127–32. Citeseer, 2006.

[10] Luis Von Ahn, Manuel Blum, Nicholas J. Hopper, and John Langford. Captcha: Using hard ai problems for security. In *Proceedings of the 22Nd International Conference on Theory and Applications of Cryptographic Techniques*, EUROCRYPT'03, pages 294–311, Berlin, Heidelberg, 2003. Springer-Verlag.

[11] Enrique Alfonseca, Marco De Boni, José-Luis Jara-Valencia, and Suresh Manandhar. A prototype question answering system using syntactic and semantic information for answer retrieval. In *TREC*, 2001.

[12] Omar Alonso and Stefano Mizzaro. Using crowdsourcing for trec relevance assessment. *Inf. Process. Manage.*, 48(6):1053–1066, November 2012.

[13] Ion Androutsopoulos and Prodromos Malakasiotis. A survey of paraphrasing and textual entailment methods. *J. Artif. Int. Res.*, 38(1):135–187, May 2010.

[14] Gabor Angeli, Julie Tibshirani, Jean Y. Wu, and Christopher D. Manning. Combining distant and partial supervision for relation extraction. In *In Proceedings of EMNLP*, pages 1556–1567, 2014.

[15] Tatsuya Asai, Kenji Abe, Shinji Kawasoe, Hiroki Arimura, Hiroshi Sakamoto, and Setsuo Arikawa. Efficient Substructure Discovery

from Large Semi-structured Data. In Robert L Grossman, Jiawei Han, Vipin Kumar, Heikki Mannila, and Rajeev Motwani, editors, *SDM*. SIAM, 2002.

[16] Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. Open information extraction from the web. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 2670–2676, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[17] Michele Banko and Oren Etzioni. The tradeoffs between open and traditional relation extraction. In *Proceedings of ACL-08: HLT*, pages 28–36, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[18] Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. The second PASCAL recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, 2006.

[19] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100. ACM, 1998.

[20] Johan Bos and Katja Markert. Recognising textual entailment with logical inference. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 628–635, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[21] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. *CoRR*, abs/1508.05326, 2015.

[22] Samuel R. Bowman, Christopher Potts, and Christopher D. Manning. Recursive neural networks for learning logical semantics. *CoRR*, abs/1406.1827, 2014.

[23] Susan L. Bryant, Andrea Forte, and Amy Bruckman. Becoming wikipedian: Transformation of participation in a collaborative online encyclopedia. In *Proceedings of the 2005 International ACM SIGGROUP Conference on Supporting Group Work*, GROUP '05, pages 1–10, New York, NY, USA, 2005. ACM.

[24] Razvan Bunescu and Raymond Mooney. Learning to extract relations from the web using minimal supervision. In *Annual meeting-association for Computational Linguistics*, volume 45, page 576, 2007.

[25] Razvan Bunescu and Raymond J Mooney. Subsequence kernels for relation extraction. In *NIPS*, pages 171–178, 2005.

[26] Razvan Bunescu and Raymond J. Mooney. Subsequence kernels for relation extraction. In Y. Weiss, B. Schoelkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems, Vol. 18: Proceedings of the 2005 Conference (NIPS)*, 2006.

[27] Razvan C. Bunescu and Raymond J. Mooney. A shortest path dependency kernel for relation extraction. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 724–731, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[28] Aljoscha Burchardt, Nils Reiter, Stefan Thater, and Anette Frank. A semantic approach to textual entailment: System evaluation and task analysis. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, RTE '07, pages 10–15, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

[29] John Burger, Claire Cardie, Vinay Chaudhri, Robert Gaizauskas, Sanda Harabagiu, David Israel, Christian Jacquemin, Chin-Yew Lin,

Steve Maiorano, George Miller, et al. Issues, tasks and program structures to roadmap research in question & answering (q&a). In *Document Understanding Conferences Roadmapping Documents*, pages 1–35, 2001.

[30] Nicola Cancedda, Eric Gaussier, Cyril Goutte, and Jean Michel Renders. Word sequence kernels. *J. Mach. Learn. Res.*, 3:1059–1082, March 2003.

[31] Jean Carletta. Assessing agreement on classification tasks: The kappa statistic. *Comput. Linguist.*, 22(2):249–254, June 1996.

[32] Eugene Charniak. A maximum-entropy-inspired parser. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 132–139. Association for Computational Linguistics, 2000.

[33] Danqi Chen and Christopher Manning. A Fast and Accurate Dependency Parser using Neural Networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar, October 2014. Association for Computational Linguistics.

[34] Yun Chi, Yi Xia, Yirong Yang, and Richard R Muntz. Mining closed and maximal frequent subtrees from databases of labeled rooted trees. *IEEE Transactions on Knowledge and Data Engineering*, 17(2):190–202, 2005.

[35] Yun Chi, Yirong Yang, and Richard R. Muntz. Mining frequent rooted trees and free trees using canonical forms, 2003.

[36] Michael Collins and Nigel Duffy. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 263–270, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[37] Michael Collins, Nigel Duffy, et al. Convolution kernels for natural language. In *NIPS*, volume 2001, pages 625–632, 2001.

[38] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537, 2011.

[39] Seth Cooper, Firas Khatib, Adrien Treuille, Janos Barbero, Jeehyung Lee, Michael Beenen, Andrew Leaver-Fay, David Baker, Zoran Popovic, and Foldit Players. Predicting protein structures with a multiplayer online game. *Nature*, 466(7307):756–760, August 2010.

[40] Mark Craven and Johan Kumlien. Constructing biological knowledge bases by extracting information from text sources. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*. AAAI Press, 1999.

[41] Hang Cui, Renxu Sun, Keya Li, Min-Yen Kan, and Tat-Seng Chua. Question answering passage retrieval using dependency relations. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 400–407. ACM, 2005.

[42] Aron Culotta and Jeffrey Sorensen. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 423. Association for Computational Linguistics, 2004.

[43] Ido Dagan, Oren Glickman, and Bernardo Magnini. *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment: First PASCAL Machine Learning Challenges Workshop, MLCW 2005, Southampton, UK, April 11-13, 2005, Revised Selected Papers*, chapter The PASCAL Recognising Textual Entailment Challenge, pages 177–190. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[44] Hal Daumé and D. Marcu. A tree-position kernel for document compression. In *Proceedings of the Fourth Document Understanding Conference (DUC 2004)*, pages 6 – 7, 2004/// 2004.

[45] R. Dawson and S. Bynghall. *Getting Results from Crowds.* Advanced Human Technologies, 2012.

[46] Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. Zencrowd: Leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 469–478, New York, NY, USA, 2012. ACM.

[47] Pinar Donmez, Jaime G. Carbonell, and Jeff Schneider. Efficiently learning the accuracy of labeling sources for selective sampling. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 259–268, New York, NY, USA, 2009. ACM.

[48] Mark Cravena Dan DiPasquoa Dayne Freitagb, Andrew McCalluma, Tom Mitchella Kamal Nigama, and Se an Slatterya. Learning to construct knowledge bases from the world wide web.

[49] Fredric Gey, Ray Larson, Mark Sanderson, Hideo Joho, Paul Clough, and Vivien Petras. Geoclef: the clef 2005 cross-language geographic information retrieval track overview. *Accessing Multilingual Information Repositories*, pages 908–919, 2006.

[50] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(12), 2009.

[51] Sanda Harabagiu and Cosmin Adrian Bejan. Question answering based on temporal inference. In *Proceedings of the AAAI-2005 workshop on inference for textual question answering*, pages 27–34, 2005.

[52] Sanda Harabagiu and Andrew Hickl. Methods for using textual entailment in open-domain question answering. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association*

for Computational Linguistics, pages 905–912. Association for Computational Linguistics, 2006.

[53] Sanda M Harabagiu, Dan I Moldovan, Marius Paşca, Rada Mihalcea, Mihai Surdeanu, Răzvan Bunescu, Corina R Gîrju, Vasile Rus, and Paul Morărescu. Falcon: Boosting knowledge for answer engines. 2000.

[54] Stefan Harmeling. Inferring textual entailment with a probabilistically sound calculus. *Natural Language Engineering*, 15(4):459–477, 2009.

[55] Michael Heilman and Noah A. Smith. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 609–617, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[56] Andrew Hickl, John Williams, Jeremy Bensley, Kirk Roberts, Ying Shi, and Bryan Rink. Question answering with lcc's chaucer at trec 2006. In *TREC*, 2006.

[57] Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. Knowledge-based weak supervision for information extraction of overlapping relations. In *ACL 2011*, HLT '11.

[58] Raphael Hoffmann, Congle Zhang, and Daniel S. Weld. Learning 5000 relational extractors. In *ACL '10*, pages 286–295, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[59] Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Chin-Yew Lin, and Deepak Ravichandran. Toward semantics-based answer pinpointing. In *Proceedings of the first international conference on Human language technology research*, pages 1–7. Association for Computational Linguistics, 2001.

[60] Jeff Howe. The rise of crowdsourcing. *Wired Magazine*, 14(6):1–4, January 2006.

[61] Pei-Yun Hsueh, Prem Melville, and Vikas Sindhwani. Data quality from crowdsourcing: A study of annotation selection criteria. In *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, HLT '09, pages 27–35, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[62] Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 873–882, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[63] Ander Intxaurrondo, Mihai Surdeanu, Oier Lopez De Lacalle, and Eneko Agirre. Removing noisy mentions for distant supervision. *Procesamiento de Lenguaje Natural*, 51:41–48, 9 2013.

[64] Panagiotis G. Ipeirotis, Foster Provost, and Jing Wang. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*, HCOMP '10, pages 64–67, New York, NY, USA, 2010. ACM.

[65] Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, CIKM '05, pages 84–90, New York, NY, USA, 2005. ACM.

[66] Heng Ji, Ralph Grishman, Hoa Trang Dang, Kira Griffitt, and Joe Ellis. Overview of the tac 2010 knowledge base population track. In *Third Text Analysis Conference (TAC 2010)*, volume 3, pages 3–3, 2010.

[67] Hyun Joon Jung. Quality Assurance in Crowdsourcing via Matrix Factorization based Task Routing. In *Proceedings of World Wide Web (WWW) Ph.D. Symposium, Companion Publication*, pages 3–8, 2014.

[68] Hyun Joon Jung and Matthew Lease. Improving Consensus Accuracy via Z-score and Weighted Voting. In *Proceedings of the 3rd Human Computation Workshop (HCOMP) at AAAI*, pages 88–90, 2011.

[69] Nanda Kambhatla. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 22. Association for Computational Linguistics, 2004.

[70] Rohit J. Kate and Raymond J. Mooney. Using string-kernels for learning semantic parsers. In *ACL 2006: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, pages 913–920, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[71] Gabriella Kazai, Jaap Kamps, and Natasa Milic-Frayling. The face of quality in crowdsourcing relevance labels: Demographics, personality and labeling accuracy. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 2583–2586, New York, NY, USA, 2012. ACM.

[72] Roman Khazankin, Daniel Schall, and Schahram Dustdar. *Predicting QoS in Scheduled Crowdsourcing*, pages 460–472. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[73] Adam Kilgarriff. Gold standard datasets for evaluating word sense disambiguation programs. 12(4):453 – 472, 1998.

[74] Aniket Kittur, Ed H. Chi, and Bongwon Suh. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 453–456, New York, NY, USA, 2008. ACM.

[75] Jeongwoo Ko, Luo Si, and Eric Nyberg. Combining evidence with a probabilistic framework for answer ranking and answer merging in question answering. *Information processing & management*, 46(5):541–554, 2010.

[76] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. Supervised machine learning: A review of classification techniques, 2007.

[77] Milen Kouylekov and Bernardo Magnini. Combining lexical resources with tree edit distance for recognizing textual entailment. In *Machine Learning Challenges, Evaluating Predictive Uncertainty, Visual Object Classification and Recognizing Textual Entailment, Machine Learning Challenges Workshop, MLCW 2005, Southampton, UK, April 11-13, 2005, Revised Selected Papers*, pages 217–230, 2005.

[78] Milen Kouylekov and Bernardo Magnini. Recognizing textual entailment with tree edit distance algorithms. In *PASCAL Challenges on RTE*, pages 17–20, 2005.

[79] Taku Kudo and Yuji Matsumoto. Fast methods for kernel-based text analysis. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 24–31, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[80] Abhimanu Kumar and Matthew Lease. Modeling annotator accuracies for supervised learning. In *Proceedings of the Workshop on Crowdsourcing for Search and Data Mining (CSDM) at the Fourth ACM International Conference on Web Search and Data Mining (WSDM)*, pages 19–22, Hong Kong, China, February 2011.

[81] Angli Liu, Xiao Ling Stephen Soderland, Jonathan Bragg, and Daniel S Weld. Effective crowd annotation for relation extraction. In *Association for Computational Linguistics*, NAACL-HLT 2016, 2016.

[82] Huma Lodhi, John Shawe-taylor, and Nello Cristianini. Text classification using string kernels. *Journal of Machine Learning Research*, 2:563–569, 2002.

[83] Bill MacCartney and Christopher D. Manning. Natural logic for textual inference. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, RTE '07, pages 193–200,

Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

[84] Bernardo Magnini, Roberto Zanoli, Ido Dagan, Kathrin Eichler, Günter Neumann, Tae-Gil Noh, Sebastian Pado, Asher Stern, and Omer Levy. The excitement open platform for textual inferences. In *Proceedings of the ACL 2014 System Demonstrations*. ACL, 6 2014.

[85] Christopher D Manning, Hinrich Schütze, et al. *Foundations of statistical natural language processing*, volume 999. MIT Press, 1999.

[86] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60, 2014.

[87] Yashar Mehdad, Alessandro Moschitti, and Fabio Massimo Zanzotto. Semker: Syntactic/semantic kernels for recognizing textual entailment. In *TAC*. NIST, 2009.

[88] Yashar Mehdad, Alessandro Moschitti, and Fabio Massimo Zanzotto. Syntactic/semantic structures for textual entailment recognition. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 1020–1028, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[89] Yashar Mehdad, Matteo Negri, and Marcello Federico. Using bilingual parallel corpora for cross-lingual textual entailment. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1336–1345, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[90] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the*

ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2, pages 1003–1011. Association for Computational Linguistics, 2009.

[91] Dan Moldovan, Christine Clark, and Sanda Harabagiu. Temporal context representation and reasoning. In *International Joint Conference on Artificial Intelligence*, volume 19, page 1099. LAWRENCE ERLBAUM ASSOCIATES LTD, 2005.

[92] Dan Moldovan, Christine Clark, Sanda Harabagiu, and Steve Maiorano. Cogex: A logic prover for question answering. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 87–93. Association for Computational Linguistics, 2003.

[93] Dan Moldovan, Marius Paşca, Sanda Harabagiu, and Mihai Surdeanu. Performance issues and error analysis in an open-domain question answering system. *ACM Transactions on Information Systems (TOIS)*, 21(2):133–154, 2003.

[94] Paloma Moreda, Hector Llorens, Estela Saquete, and Manuel Palomar. Combining semantic information in question answering systems. *Inf. Process. Manage.*, 47(6):870–885, November 2011.

[95] Alessandro Moschitti. Efficient convolution kernels for dependency and constituent syntactic trees. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *Proceedings of the 17th European Conference on Machine Learning (ECML 2006), September 18-22, 2006, Berlin, Germany*, volume 4212 of *Lecture Notes in Computer Science*, pages 318–329. Springer, Berlin–Heidelberg, Germany, 2006.

[96] Alessandro Moschitti. Making tree kernels practical for natural language learning. In *EACL*, 2006.

[97] Alessandro Moschitti, Silvia Quarteroni, Roberto Basili, and Suresh Manandhar. Exploiting syntactic and shallow semantic kernels for

question answer classification. In *In Proc. of ACL-07*, pages 776–783, 2007.

[98] Srini Narayanan and Sanda Harabagiu. Question answering based on semantic structures. In *Proceedings of the 20th international conference on Computational Linguistics*, page 693. Association for Computational Linguistics, 2004.

[99] Andrew Ng. Bay area deep learning school day 2 at cemex auditorium, stanford. https://www.youtube.com/watch?v=9dXiAecyJrY, September 2016.

[100] Truc-Vien T. Nguyen and Alessandro Moschitti. End-to-end relation extraction using distant supervision from external semantic repositories. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, pages 277–282, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[101] Truc-Vien T Nguyen and Alessandro Moschitti. Joint distant and direct supervision for relation extraction. In *IJCNLP*, pages 732–740, 2011.

[102] Truc-Vien T Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. Convolution kernels on constituent, dependency and sequential structures for relation extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, pages 1378–1387. Association for Computational Linguistics, 2009.

[103] Long-Van Nguyen-Dinh, Mirco Rossi, Ulf Blanke, and Gerhard Tröster. Combining crowd-generated media and personal data: Semi-supervised learning for context recognition. In *Proceedings of the 1st ACM International Workshop on Personal Data Meets Distributed Multimedia*, PDM '13, pages 35–38, New York, NY, USA, 2013. ACM.

[104] Robert M Nosofsky. The generalized context model: An exemplar model of classification. *Formal approaches in categorization*, pages 18–39, 2011.

[105] Adrian Novischi and Dan Moldovan. Question answering with lexical chains propagating verb arguments. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 897–904. Association for Computational Linguistics, 2006.

[106] Stefanie Nowak and Stefan Rüger. How reliable are annotations via crowdsourcing: A study about inter-annotator agreement for multi-label image annotation. In *Proceedings of the International Conference on Multimedia Information Retrieval*, MIR '10, pages 557–566, New York, NY, USA, 2010. ACM.

[107] Kucherbaev Pavel. *Quality Assurance Strategies in Microtask Crowdsourcing*. PhD dissertation, University of Trento, 2016.

[108] Maria Pershina, Bonan Min, Wei Xu, and Ralph Grishman. Infusion of labeled data into distant supervision for relation extraction. In *ACL 2014)*, Baltimore, US, June 2014. Association for Computational Linguistics.

[109] Barbara Plank, Dirk Hovy, and Anders Sogaard. *Learning part-of-speech taggers with inter-annotator agreement loss*, pages 742–751. Association for Computational Linguistics, 2014.

[110] Dražen Prelec, H Sebastian Seung, and John McCoy. A solution to the single-question crowd wisdom problem. *Nature*, 541(7638):532–535, 2017.

[111] Matthew Purver and Stuart Battersby. Experimenting with distant supervision for emotion classification. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 482–491. Association for Computational Linguistics, 2012.

[112] James Pustejovsky, Roser Saurí, José M Castaño, Dragomir R Radev, Robert J Gaizauskas, Andrea Setzer, Beth Sundheim, and Graham Katz. Representing temporal and event knowledge for qa systems. In *New directions in question answering*, pages 99–112, 2004.

[113] Er J. Quinn, Benjamin B. Bederson, Tom Yeh, and Jimmy Lin. Crowdflow: Integrating machine learning with mechanical turk for speed-cost-quality flexibility. Technical report, 2010.

[114] J. Raddick, C. Lintott, S. Bamford, K. Land, D. Locksmith, P. Murray, B. Nichol, K. Schawinski, A. Slosar, A. Szalay, D. Thomas, J. Vandenberg, and D. Andreescu. Galaxy Zoo: Motivations of Citizen Scientists. In *American Astronomical Society Meeting Abstracts #212*, volume 40 of *Bulletin of the American Astronomical Society*, page 240, May 2008.

[115] Filip Radlinski and Thorsten Joachims. Query chains: Learning to rank from implicit feedback. *CoRR*, abs/cs/0605035, 2006.

[116] Vikas C. Raykar and Shipeng Yu. Eliminating spammers and ranking annotators for crowdsourced labeling tasks. *J. Mach. Learn. Res.*, 13:491–518, February 2012.

[117] Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. *J. Mach. Learn. Res.*, 11:1297–1322, August 2010.

[118] Han Ren, Donghong Ji, and Jing Wan. WHU at TAC 2009: A tri-categorization approach to textual entailment recognition. In *Proceedings of the Second Text Analysis Conference, TAC 2009, Gaithersburg, Maryland, USA, November 16-17, 2009*, 2009.

[119] Sebastian Riedel, Limin Yao, and Andrew McCallum. Modeling relations and their mentions without labeled text. In *ECML PKDD'10 : Part III*, 2010.

[120] Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. Relation extraction with matrix factorization and universal schemas. 2013.

[121] Ryan Michael Rifkin. *Everything old is new again: a fresh look at historical approaches in machine learning.* PhD thesis, MaSSachuSettS InStitute of Technology, 2002.

[122] S. E. Robertson and S. Walker. On relevance weights with little relevance information. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '97, pages 16–24, New York, NY, USA, 1997. ACM.

[123] Olga Russakovsky, Li-Jia Li, and Li Fei-Fei. Best of both worlds: Human-machine collaboration for object annotation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[124] Evan Sandhaus. The new york times annotated corpus ldc2008t19. philadelphia: Linguistic data consortium. 2008.

[125] Estela Saquete, Jose Luis Vicedo, Patricio Martínez-Barco, Rafael Muñoz, and Fernando Llopis. Evaluation of complex temporal questions in clef-qa. In *Workshop of the Cross-Language Evaluation Forum for European Languages*, pages 591–596. Springer, 2004.

[126] Neil Savage. Gaining wisdom from crowds. *Commun. ACM*, 55(3):13–15, March 2012.

[127] Asad B. Sayeed, Jordan L. Boyd-Graber, Bryan Rusk, and Amy Weinberg. Grammatical structures for word-level sentiment detection. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 3-8, 2012, Montréal, Canada*, pages 667–676, 2012.

[128] Dennis Shasha, Jason Tsong-Li Wang, and Sen Zhang. Unordered tree mining with applications to phylogeny. In *Data Engineering, 2004. Proceedings. 20th International Conference on*, pages 708–719. IEEE, 2004.

[129] Dan Shen and Mirella Lapata. Using semantic roles to improve question answering. In *EMNLP-CoNLL*, pages 12–21, 2007.

[130] Dan Shen and Mirella Lapata. Using semantic roles to improve question answering. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 12–21, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[131] Victor S. Sheng, Foster Provost, and Panagiotis G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 614–622, New York, NY, USA, 2008. ACM.

[132] Aashish Sheshadri and Matthew Lease. Square: A benchmark for research on computing crowd consensus. In *HCOMP*. AAAI, 2013.

[133] Elena Smirnova. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 1191–1192, New York, NY, USA, 2011. ACM.

[134] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 254–263, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[135] Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng, and Christopher D. Manning. Parsing natural scenes and natural language with recursive neural networks. In *ICML*, pages 129–136. Omnipress, 2011.

[136] Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank.

[137] Chong Sun, Narasimhan Rampalli, Frank Yang, and AnHai Doan. Chimera: Large-scale classification using machine learning, rules, and crowdsourcing. *Proc. VLDB Endow.*, 7(13):1529–1540, August 2014.

[138] Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. Learning to rank answers on large online qa collections. In *In Proceedings of the 46th Annual Meeting for the Association for Computational Linguistics: Human Language Technologies (ACL-08: HLT*, pages 719–727, 2008.

[139] Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. Multi-instance multi-label learning for relation extraction. In *EMNLP-CoNLL 12*, 2012.

[140] Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. Reducing wrong labels in distant supervision for relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 721–729. Association for Computational Linguistics, 2012.

[141] Wei Tang and Matthew Lease. Semi-supervised consensus labeling for crowdsourcing. In *ACM SIGIR Workshop on Crowdsourcing for Information Retrieval (CIR)*, pages 36–41, 2011.

[142] Alexandre Termier, Marie-Christine Rousset, and Michèle Sebag. DRYADE: A new approach for discovering closed frequent trees in heterogeneous tree databases. In *Proceedings of the 4th IEEE International Conference on Data Mining (ICDM 2004), 1-4 November 2004, Brighton, UK*, pages 543–546, 2004.

[143] Kateryna Tymoshenko, Alessandro Moschitti, and Aliaksei Severyn. *Encoding semantic resources in syntactic structures for passage reranking*, pages 664–672. Association for Computational Linguistics (ACL), 1 2014.

[144] Noortje J. Venhuizen, Valerio Basile, Kilian Evang, and Johan Bos. Gamification for word sense labeling. In *Proceedings of the 10th International Conference on Computational Semantics, IWCS 2013,*

March 19-22, 2013, University of Potsdam, Potsdam, Germany, pages 397–403, 2013.

[145] Timo Volkmer, James A Thom, and Seyed MM Tahaghoghi. Modeling human judgment of digital imagery for multimedia retrieval. *Multimedia, IEEE Transactions on*, 9(5):967–974, 2007.

[146] Luis von Ahn and Laura Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 319–326, New York, NY, USA, 2004.

[147] Ellen M. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. *Inf. Process. Manage.*, pages 697–716, 2000.

[148] Ellen M. Voorhees. The trec question answering track. *Nat. Lang. Eng.*, 7(4):361–378, December 2001.

[149] Ellen M Voorhees and L Buckland. Overview of the trec 2003 question answering track. In *TREC*, volume 2003, pages 54–68, 2003.

[150] Ellen M. Voorhees and Donna K. Harman. *TREC: Experiment and Evaluation in Information Retrieval (Digital Libraries and Electronic Publishing)*. The MIT Press, 2005.

[151] Frane Šarić, Goran Glavaš, Mladen Karan, Jan Šnajder, and Bojana Dalbelo Bašić. Takelab: Systems for measuring semantic text similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, SemEval '12, pages 441–448, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[152] Mengqiu Wang and Christopher D. Manning. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *Proceedings of the 23rd*

International Conference on Computational Linguistics, COLING '10, pages 1164–1172, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[153] Mengqiu Wang, Noah A Smith, and Teruko Mitamura. What is the jeopardy model? a quasi-synchronous grammar for qa. *EMNLP-CoNLL*, 7:22–32, 2007.

[154] Kelly A. Way, Michael C. Ottenbacher, and Robert J. Harrington. Is crowdsourcing useful for enhancing innovation and learning outcomes in culinary and hospitality education? *Journal of Culinary Science & Technology*, 9(4):261–281, 2011.

[155] Peter Welinder, Steve Branson, Pietro Perona, and Serge J. Belongie. The multidimensional wisdom of crowds. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 2424–2432. Curran Associates, Inc., 2010.

[156] Jacob Whitehill, Ting fan Wu, Jacob Bergsma, Javier R. Movellan, and Paul L. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 2035–2043. Curran Associates, Inc., 2009.

[157] Jacob Whitehill, Ting fan Wu, Jacob Bergsma, Javier R. Movellan, and Paul L. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 2035–2043. Curran Associates, Inc., 2009.

[158] Fei Wu and Daniel S. Weld. Autonomously semantifying wikipedia. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 41–50, New York, NY, USA, 2007. ACM.

[159] Liang Huai Yang, Mong Li Lee, Wynne Hsu, and Xinyu Guo. 2pxminer: An efficient two pass mining of frequent xml query patterns. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 731–736, New York, NY, USA, 2004. ACM.

[160] Limin Yao, Sebastian Riedel, and Andrew McCallum. Universal schema for entity type prediction. In *Proceedings of the 2013 Workshop on Automated Knowledge Base Construction*, AKBC '13, pages 79–84, New York, NY, USA, 2013. ACM.

[161] Xuchen Yao, Benjamin Van Durme, Chris Callison-burch, and Peter Clark. Answer extraction as sequence tagging with tree edit distance. In *In North American Chapter of the Association for Computational Linguistics (NAACL*, 2013.

[162] Fabio massimo Zanzotto, Marco Pennacchiotti, and Alessandro Moschitti. A machine learning approach to textual entailment recognition. *Nat. Lang. Eng.*, 15(4):551–582, October 2009.

[163] Alessandra Zarcone and Stefan Rued. Logical metonymies and qualia structures: an annotated database of logical metonymies for german. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may 2012. European Language Resources Association (ELRA).

[164] Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. Kernel methods for relation extraction. *The Journal of Machine Learning Research*, 3:1083–1106, 2003.

[165] Haijun Zhai, Todd Lingren, Louise Deleger, Qi Li, Megan Kaiser, Laura Stoutenborough, and Imre Solti. Web 2.0-based crowdsourcing for high-quality gold standard development in clinical natural language processing. *J Med Internet Res*, 15(4):e73, Apr 2013.

[166] Ce Zhang, Feng Niu, Christopher Ré, and Jude Shavlik. Big data versus the crowd: Looking for relationships in all the right places. In

ACL, pages 825–834, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[167] Dell Zhang and Wee Sun Lee. Question classification using support vector machines. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 26–32. ACM, 2003.

[168] Jing Zhang, Victor S. Sheng, Jian Wu, Xiaoqin Fu, and Xindong Wu. Improving label quality in crowdsourcing using noise correction. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, pages 1931–1934, New York, NY, USA, 2015. ACM.

[169] Min Zhang, Jie Zhang, and Jian Su. Exploring syntactic features for relation extraction using a convolution tree kernel. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 288–295. Association for Computational Linguistics, 2006.

[170] Min Zhang, Jie Zhang, Jian Su, and Guodong Zhou. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 825–832. Association for Computational Linguistics, 2006.

[171] Shubin Zhao and Ralph Grishman. Extracting relations with integrated information using kernel methods. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 419–426, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.