申请上海交通大学硕士学位论文

Indirect Supervision in Information Extraction

论文作者 许方正

学　　号 116033910084

导　　师 朱其立教授

专　　业 计算机技术

答辩日期 2019 年 1 月 7 日

Submitted in total fulfillment of the requirements for the degree of Master in Computer Science

Indirect Supervision in Information Extraction

FANGZHENG XU

Advisor
Prof. KENNY QILI ZHU

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING,
SCHOOL OF ELECTRONIC INFORMATION AND ELECTRICAL ENGINEERING
SHANGHAI JIAO TONG UNIVERSITY
SHANGHAI, CHINA

Jan. 7th, 2019

Indirect Supervision in Information Extraction

摘 要

在本文中，我们探讨了自然语言处理子任务信息提取中的各种研究问题，特别注重利用间接监督来利用丰富的无标注数据和内部语义信息，来减少对大量人工标注文本的依赖。神经网络可以构建没有手工特征的可靠模型。然而，在许多情况下，我们很难获得足够的标注数据来训练这些模型。在本研究中，从命名实体识别任务开始，我们开发了一个神经框架，用于从原始文本中提取知识并且提高序列标注任务的准确度。除了预训练的词向量中包含的单词级信息之外，我们还结合了字符级别的知神经语言模型，并进一步地采用迁移学习技术引导语言模型。与以前的方法相比，这些特定于任务的知识使我们能够采用更简洁的模型并进行更有效的训练。与大多数迁移学习方法不同，我们提出框架不依赖于任何额外的监督，可以从训练序列的自包含顺序信息中提取有帮助的监督信息。模型在基准数据集的实验证明了利用字符级信息和共同训练的有效性和高效率。在 CoNLL03 NER 任务中，模型训练在单个 GPU 上完成大约 6 小时，在不使用任何额外标注的情况下达到 91.71 ± 0.10 的 F1 分数。除了通用文本的命名实体识别之外，我们还提出了用于识别社交媒体文本中新兴命名实体的多通道神经网络架构。我们提出了一种将综合的单词表示与多通道信息和条件随机场（CRF）结合到一个传统的双向长短期记忆（BiLSTM）神经网络中，且使用任何其他特征工程。与一同参赛的其他系统相比，我们的系统在两个评估指标的平均值方面获得了第二名。

拥有了良好的命名实体提取模型作为前提，为了以更加有效的方式解释大量文本语料，对感兴趣的类型进行自动关系提取是很重要的。传统的关系抽取模型在训练的时候严重依赖于人工标注的数据，人工生产标签数据的成本是很高的，而且人工标签会成为处理多种类型关系时的障碍。因此，更多的关系提取系统转向建立在基于通过和知识库链接自动获取的训练数据（远监督方法）。然而，因为知识库的不完整和语境不可知的自动标签的原因，通过远监督得到的训练数据含有很多噪声。在最近几年，解决问答任务越来越受关注，这类任务的用户反馈和数据集都容易获得了。我们提出了一个新颖的框架来利用问答对作为关系提取的一个间接监督源，还研究了如何使用这种监督来减少从知识库中产生的噪声。我们的模型将关系提述、类型、问答实体提述对以及文本特征联合地嵌入到了两个低维空间中（关系提取和问答），在这个低维空间中，具有相同关系类型或者语义相似的问答对会拥有相似的表征，共享的特征将这两个空间连接起来，

从两个源中传递更加清晰的语义知识。然后使用这些学习到的向量去估计测试集的关系提述的类型。我们构造了一个全局目标函数，采用一个新型的边际问答损失指标，通过利用问答数据集中的语义特征去降低知识库所产生的噪声。结合两个公开的关系提取数据集 TREC QA 数据集，我们的实验结果在 F1 上达到了 11% 的提升。

除了提取命名实体之间关系的传统任务外，我们还将问题扩展到一般对象及它们之间的关系。具体而言，位置关系是一种常识性知识，描述了两个在现实生活中通常彼此相邻的物理对象。我们研究如何通过句子级关系分类器自动提取这种关系，并且通过大型语料库中打分，分类，聚合实体对来为 ConceptNet 常识数据集进行自动知识抽取扩展。此外，我们还发布了两个基准数据集，用于评估未来的类似研究。和社会科学相结合，另一方面作为信息提取的新扩展，跨文化差异和相似性在跨语言自然语言理解，特别是对于社交媒体的研究中很常见。例如，不同文化背景的人通常对一个命名实体持有不同的意见。此外，理解跨语言的俚语需要跨文化相似性的知识。我们研究计算这种跨文化差异和相似性的问题，并提出了一种轻量级但有效的方法，并对两项新任务进行评估：1）挖掘命名实体的跨文化差异; 2）找到跨语言俚语的类似解释。

为了将信息提取任务从闭域和特定目标类型集解放出来，我们研究开放信息提取系统，以从句子中挖掘关系元组，并且不限定事先定义好的关系类型。然而，当前开放信息抽取系统专注于对句子中的局部上下文信息进行建模以提取关系元组，而忽略了可以集体利用大型语料库中的全局统计来识别高质量句子级提取的事实。我们将局部语句的信息和全局结构信号整合到一个具有远监督学习的统一框架中。新系统可以有效地应用于不同的文本领域，因为它使用来自外部知识库的事实作为非直接监督; 并且可以基于语料库统计有效地对句子级元组提取进行评分。与其他开放式信息抽取系统相比，我们利用不同领域的两个实际语料库的实验证明了其有效性和稳健性。

关键词： 信息抽取　非直接监督学习　自然语言处理　文本挖掘

INDIRECT SUPERVISION IN INFORMATION EXTRACTION

ABSTRACT

In this thesis we explore various research problems in information extraction of NLP with a special focus on using indirect supervision to leverage the abundant unlabeled data and exploit internal semantic information without relying on heavy annotations. Recent advances in neural networks (NNs) make it possible to build reliable models without handcrafted features. However, in many cases, it is hard to obtain sufficient annotations to train these models. In this study, starting with the named entity recognition (NER) task, we develop a neural framework to extract knowledge from raw texts and empower the sequence labeling task. Besides word-level knowledge contained in pre-trained word embeddings, character-aware neural language models are incorporated to extract character-level knowledge. Transfer learning techniques are further adopted to mediate different components and guide the language model towards the key knowledge. Comparing to previous methods, these task-specific knowledge allows us to adopt a more concise model and conduct more efficient training. Different from most transfer learning methods, the proposed framework does not rely on any additional supervision. It extracts knowledge from self-contained order information of training sequences. Extensive experiments on benchmark datasets demonstrate the effectiveness of leveraging character-level knowledge and the efficiency of co-training. On the CoNLL03 NER task, model training completes in about 6 hours on a single GPU, reaching F1 score of 91.71±0.10 without using any extra annotations. Besides NER for general domain text, we present our multi-channel neural architecture for recognizing emerging named entity in social media messages. We propose a novel approach, which incorporates comprehensive word representations with multi-channel information and Conditional Random Fields (CRF) into a traditional Bidirectional Long Short-Term Memory (BiLSTM) neural network without using any additional hand-crafted features such as gazetteers. In comparison with other systems participating in the shared task, our system won the 2nd place in terms of the average of two evaluation metrics.

With state-of-the-art models to extract named entities, we investigate that relation extraction

(RE) for types of interest is of great importance for interpreting massive text corpora in an efficient manner. Traditional RE models have heavily relied on human-annotated corpus for training, which can be costly in generating labeled data and become obstacles when dealing with more types. Thus, more recent relation extraction systems have shifted to be built upon training data automatically acquired by linking to knowledge bases (distant supervision). However, due to the incompleteness of knowledge bases and the context-agnostic labeling, the training data collected via distant supervision (DS) can be very noisy. In recent years, as the increasing attention has been brought to tackling question-answering (QA) tasks, user feedbacks or datasets of such tasks become more accessible. We propose a novel framework to leverage question-answer pairs as an indirect source of supervision for relation extraction, and study how to use such supervision to reduce noise induced from DS. Our model jointly embeds relation mentions, types, QA entity mention pairs and text features in two low-dimensional spaces (RE and QA), where objects with same relation types or semantically similar question-answer pairs have close representations. Shared features connect these two spaces, carrying clearer semantic knowledge from both sources. We then use these learned embeddings to estimate the types of test relation mentions. We formulate a global objective function and adopt a novel margin-based QA loss to reduce noise in DS by exploiting semantic evidence from the QA dataset. Our experimental results achieve the average 11% improvement in F1 score on two public RE datasets combined with TREC QA dataset.

Besides the traditional task of extracting relations between named entities, we also extend the problem to general objects and the relations between them. Specifically, LocatedNear relation is a kind of commonsense knowledge describing two physical objects that are typically found near each other in real life. We study how to automatically extract such relationship through a sentence-level relation classifier and aggregating the scores of entity pairs from a large corpus. Also, we release two benchmark datasets for evaluation and future research. As a new extension in information extraction, cross-cultural differences and similarities are common in cross-lingual natural language understanding, especially for research in social media. For instance, people of distinct cultures often hold different opinions on a single named entity. Also, understanding slang terms across languages requires knowledge of cross-cultural similarities. We study the problem of computing such cross-cultural differences and similarities. We present a lightweight yet effective approach, and evaluate it on two novel tasks: 1) mining cross-cultural differences of named entities and 2) finding similar terms for slang across languages.

To free information extraction tasks from closed domain and target set, we study open information extraction systems to mine relation tuples from sentences, and do not confine to a pre-defined schema for the relations of interests. However, current open IE systems focus on modeling local context information in a sentence to extract relation tuples, while ignoring the fact that global statistics in a large corpus can be collectively leveraged to identify high-quality sentence-level extractions. We integrate local context signal and global structural signal in a unified framework with distant supervision. The new system can be efficiently applied to different domains as it uses facts from external knowledge bases as supervision; and can effectively score sentence-level tuple extractions based on corpus-level statistics. Experiments on two real-world corpora from different domains demonstrate the effectiveness and robustness when compared to other open IE systems.

KEY WORDS: information extraction, indirect supervision, natural language processing, text mining

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1 Introduction

Massive text corpora are emerging worldwide in different domains and languages. The sheer size of such unstructured data and the rapid growth of new data pose grand challenges on making sense of these massive corpora. Information extraction (IE)[1] – extraction of relation tuples in the form of (*head entity*, relation, *tail entity*) – is a key step towards automating knowledge acquisition from text. In Fig. 7–1, for example, the relation tuple (*Louvre-Lens*, build, *new satellites*) can be extracted from sentence S_2 to represent a piece of factual knowledge in text with structured form. Relation tuples so extracted have a variety of downstream applications, including serving as building blocks for knowledge base construction[2] and facilitating question answering systems[3, 4].

Named entity recognition (NER) is one of the first and most important steps in Information Extraction pipelines. Generally, it is to identify mentions of entities (persons, locations, organizations, etc.) within unstructured text. After identifying all the named entities in the large text corpora, we could do tasks like relation extraction for named entities.

Relation extraction is an important task for understanding massive text corpora by turning unstructured text data into relation triples for further analysis. For example, it detects the relationship "president_of" between entities "*Donald Trump*" and "*United States*" in a sentence. Such extracted information can be used for more downstream text analysis tasks (e.g. serving as primitives for information extraction and knowledge base (KB) completion, and assisting question answering systems).

However, most research focused on knowledge graphs for named entities and their relations, yet commonsense knowledge is lesser researched at the time. ConceptNet is one of the few examples but its content is human-curated and limited. Commonsense knowledge is an important ingredient in machine comprehension and inference. Artificial intelligence systems can benefit from incorporating commonsense knowledge as background, such as *ice is cold* (HasProperty), *chewing is a sub-event of eating* (HasSubevent), *chair and table are typically found near each other* (LocatedNear), etc. These kinds of commonsense facts have been used in many downstream tasks, such as textual entailment[5, 6] and visual recognition tasks[7]. The commonsense knowledge is often represented as relation triples in commonsense knowledge bases, such as *ConceptNet*[8], one of the largest commonsense knowledge graphs available today.

However, most commonsense knowledge bases are manually curated or crowd-sourced by community efforts and thus do not scale well. With reasoning capability and model interpretability in mind, the commonsense knowledge question came to me. I then constructed dataset and models to automatically extract, aggregate and populate commonsense spatial knowledge (the most limited relation type in ConceptNet) from literature texts (novels usually have descriptions of physical scenes). It may help other tasks (object detection) through knowledge reasoning and logical rules.

Apart from these well-defined research problems in information extraction, we also shifted our focus onto the intersection of natural language processing with computational social science. We propose new task of cross-lingual, cross-cultural word embeddings and internet slang translation from social media under different cultural backgrounds. Many current NLP systems like machine translation or dialogue systems are not aware of the user’s cultural background and its implications. With the help of socio-linguistic features to model cultural differences, it could potentially make downstream tasks socially and culturally aware.

However, one of the most important issue in current state of the art methods in these information extraction task is the heavy reliance of human annotated training data or noise and error prone. Typically, RE systems rely on training data, primarily acquired via human annotation, to achieve satisfactory performance. However, such manual labeling process can be costly and non-scalable when adapting to other domains (e.g. biomedical domain). In addition, when the number of types of interest becomes large, the generation of handcrafted training data can be error-prone. To alleviate such an exhaustive process, the recent trend has deviated towards the adoption of distant supervision (DS). DS replaces the manual training data generation with a pipeline that automatically links texts to a knowledge base (KB). The pipeline has the following steps: (1) detect entity mentions in text; (2) map detected entity mentions to entities in KB; (3) assign, to the candidate type set of each entity mention pair, all KB relation types between their KB-mapped entities. However, the noise introduced to the automatically generated training data is not negligible. There are two major causes of error: incomplete KB and context-agnostic labeling process. If we treat unlinkable entity pairs as the pool of negative examples, false negatives can be commonly encountered as a result of the insufficiency of facts in KBs, where many true entity or relation mentions fail to be linked to KBs (see example in Figure 4–1). In this way, models counting on extensive negative instances may suffer from such misleading training data. On the other hand, context-agnostic labeling can engender false positive examples, due to the inaccuracy of the DS assumption that if a sentence contains any two entities holding a relation in the KB, the sentence must be expressing such relation between them. For example, entities "*Donald Trump*" and "*United States*" in the sentence "*Donald Trump flew back to United States*" can be labeled as "`president_of`" as well as "`born_in`", although only an out-of-interest relation type "`travel_to`" is expressed explicitly.

To alleviate such exhaustive process, two main lines of work have emerged: weak supervision and distant supervision (DS). Weak supervision relies on a small set of manually-specified seed instances (or patterns) that are applied in bootstrapping learning to identify more instances of each type. This assumes seeds are unambiguous and sufficiently frequent in the corpus, which requires careful seed selection by human. The recent trend has deviated towards the adoption of distance supervision (DS). DS generates training data automatically by aligning texts and a knowledge base(KB). The typical workflow is : (1) detect entity mentions in text; (2) map detected entity mentions to entities in KB; (3) assign, to the candidate type set of each entity mention pair, all KB relation types between their KB-mapped entities. The automatically labeled training corpus is then used to infer types of the remaining candidate entity mentions and relation mentions (i.e., unlinkable candidate mentions).

Towards the goal of diminishing the negative effects by noisy DS training data, distantly supervised RE models that deal with training noise, as well as methods that directly improve the automatic training data generation process have been proposed. These methods mostly involve designing distinct assumptions to remove redundant training information[9–12]. For example, method applied in[10, 11] assumes that for each relation triple in the KB, at least one sentence might express the relation instead of all sentences. Moreover, these noise reduction systems usually only address one type of error, either false positives or false negatives. Hence, current methods handling DS noises still have the following challenges:

1. Lack of trustworthy sources: Current de-noising methods mainly focus on recognizing labeling mistakes from the labeled data itself, assisted by pre-defined assumptions or patterns. They do not have external trustworthy sources as guidance to uncover incorrectly labeled data, while not at the expense of excessive human efforts. Without other separate information sources, the reliability of false label identification can be limited.
2. Incomplete noise handling: Although both false negative and false positive errors are observed to be significant, most existing works only address one of them.

While traditional IE systems require people to pre-specify the set of relations of interests, recent studies on *open-domain information extraction* (Open IE)[13–15] rely on *relation phrases* extracted from text to represent the entity relationship, making it possible to adapt to various domains (*i.e.*, open-domain) and different languages (*i.e.*, language-independent). Thus we also try to extend the information extraction to open domain with minimum supervision.

In this thesis we explored various research problems in *information extraction*, with a focus on utilizing **indirect supervision** by exploiting outside supplementary data or the data itself inherent traits. We first start with tackling the named entity recognition problem, then relation extraction problem. We extend to open domain information extraction and also propose novel tasks related to extracting cultural differences in the social media domain.

In the second chapter, we proposed a sequence labeling framework, which effectively leverages the language model to extract character-level knowledge from the self-contained order information. Highway layers are incorporated to overcome the discordance issue of the naive co-training Benefited from the effectively captured such task-specific knowledge, we can build a much more concise model, thus yielding much better efficiency without loss of effectiveness (achieved the state-of-the-art on three benchmark datasets) . In the third chapter, we present a novel multi-channel BiLSTM-CRF model for emerging named entity recognition in social media messages. We find that BiLST-CRF architecture with our proposed comprehensive word representations built from multiple information are effective to overcome the noisy and short nature of social media messages. In the fourth chapter, we present a novel study on indirect supervision (from question-answering datasets) for the task of relation extraction. We propose a framework, ReQuest, that embeds information from both training data automatically generated by linking to knowledge bases and QA datasets, and captures richer semantic knowledge from both sources via shared text features so that better feature embeddings can be learned to infer relation type for test relation mentions despite the noisy training data. Our experiment results on two datasets demonstrate the effectiveness and robustness of ReQuest. In the fifth chapter, we propose to identify LocatedNear relation from literature text and construct a knowledge base of object pairs that would commonly appear near each other in real world. We present a novel study on enriching LocatedNear relationship from textual corpora. Based on our two newly-collected benchmark datasets, we propose several methods to solve the sentence-level relation classification problem. We show that existing methods do not work as well on this task and discovered that LSTM-based model does not have significant edge over simpler feature-based model. Whereas, our multi-level sentence normalization turns out to be useful. In the fifth chapter, we present the SocVec method to compute cross-cultural differences and similarities, and evaluate it on two novel tasks about mining cross-cultural differences in named entities and computing cross-cultural similarities in slang terms. Through extensive experiments, we demonstrate that the proposed lightweight yet effective method outperforms a number of baselines, and can be useful in translation applications and cross-cultural studies in computational social science. In the final chapter, we study the task of open information extraction and proposes a principled framework, ReMine, to unify local contextual information and global structural cohesiveness for effective extraction of relation tuples. ReMine leverages distant supervision in conjunction with existing knowledge bases to provide automatically-labeled sentence and guide the entity and relation segmentation. The local objective is further learned together with a translating-based objective to enforce structural cohesiveness, such that corpus-level statistics are incorporated for boosting high-quality tuples extracted from individual sentences. We develop a joint optimization algorithm to efficiently solve the proposed unified objective function and can output quality extractions by taking into account both local and global information. Experiments on two real-world corpora of different domains demonstrate that ReMine system achieves superior precision when outputting same number of extractions, compared with several state-of-the-art open IE systems. As a byproduct, ReMine also demonstrates competitive performance on detecting mentions of entities from text when compared to several named entity recognition algorithms.

Chapter 2 Named Entities Recognition with Language Models

2.1 Introduction

Linguistic sequence labeling is a fundamental framework. It has been applied to a variety of tasks including part-of-speech (POS) tagging, noun phrase chunking and named entity recognition (NER)[16, 17]. These tasks play a vital role in natural language understanding and fulfill lots of downstream applications, such as relation extraction, syntactic parsing, and entity linking[18, 19].

Traditional methods employed machine learning models like Hidden Markov Models (HMMs) and Conditional Random Fields (CRFs), and have achieved relatively high performance. However, these methods have a heavy reliance on handcrafted features (e.g., whether a word is capitalized) and language-specific resources (e.g., gazetteers). Therefore, it could be difficult to apply them to new tasks or shift to new domains. To overcome this drawback, neural networks (NNs) have been proposed to automatically extract features during model learning. Nevertheless, considering the overwhelming number of parameters in NNs and the relatively small size of most sequence labeling corpus, annotations alone may not be sufficient to train complicated models. So, guiding the learning process with extra knowledge could be a wise choice.

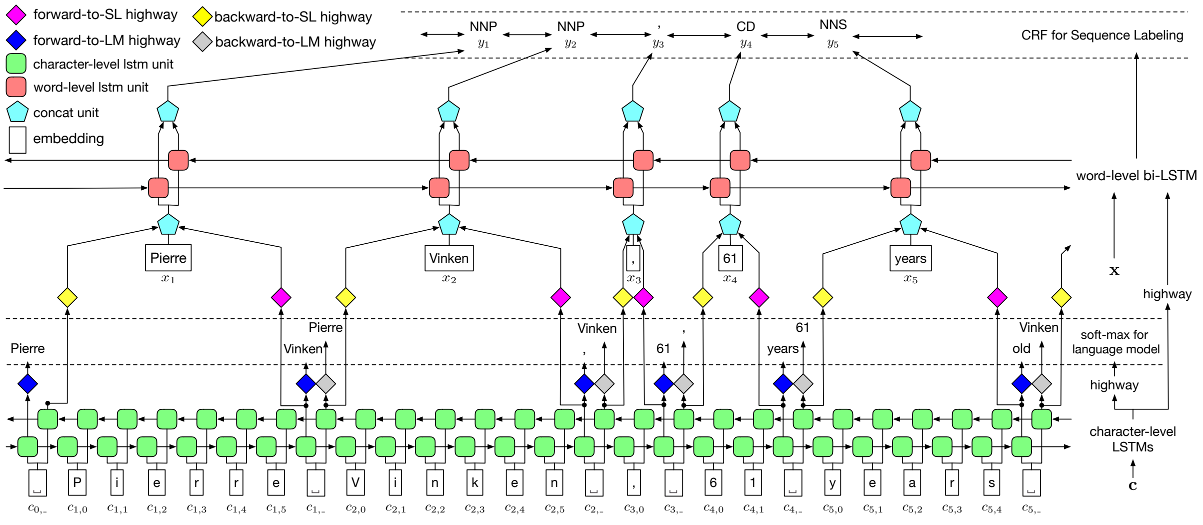


Figure 2–1 LM-LSTM-CRF Neural Architecture

Accordingly, transfer learning and multi-task learning have been proposed to incorporate such knowledge. For example, NER can be improved by jointly conducting other related tasks like entity linking or chunking[19, 20]. After all, these approaches would require additional supervision on related tasks, which might be hard to get, or not even existent for low-resource languages or special domains.

Alternatively, abundant knowledge can be extracted from raw texts, and enhance a variety of tasks. Word embedding techniques represent words in a continuous space[21, 22] and retain the semantic relations among words. Consequently, integrating these embeddings could be beneficial to many tasks[18, 23]. Nonetheless, most embedding methods take a word as a basic unit, thus only obtaining word-level knowledge, while character awareness is also crucial and highly valued in most state-of-the-art NN models.

Only recently, character-level knowledge has been leveraged and empirically verified to be helpful in numerous sequence labeling tasks[24, 25]. Directly adopting pre-trained language models, character-level knowledge can be integrated as context embeddings and demonstrate its potential to achieve the state-of-the-art[24]. However, the knowledge extracted through pre-training is not task-specific, thus containing a large irrelevant portion. So, this approach would require a bigger model, external corpus and longer training. For example, one of its language models was trained on 32 GPUs for more than half a month, which is unrealistic in many situations.

In this chapter, we propose an effective sequence labeling framework, LM-LSTM-CRF , which leverages both word-level and character-level knowledge in an efficient way. For character-level knowledge, we incorporate a neural language model with the sequence labeling task and conduct multi-task learning to guide the language model towards task-specific key knowledge. Besides the potential of training a better model, this strategy also poses a new challenge. Based on our experiments, when the tasks are discrepant, language models could be harmful to sequence labeling in a naïve co-training setting. For this reason, we employ highway networks[26] to transform the output of character-level layers into different semantic spaces, thus mediating and unifying these two tasks. For word-level knowledge, we choose to fine-tune pre-trained word embeddings instead of co-training or pre-training the whole word-level layers, because the majority of parameters in word-level layers come from the embedding layer and such co-training or pre-training cost lots of time and resources.

We conduct experiments on the CoNLL 2003 NER task, the CoNLL 2000 chunking task, as well as the WSJ portion of the Penn Treebank POS tagging task. LM-LSTM-CRF achieves

$\mathbf{x}$	word-level input	x_i	i-th word
$\mathbf{c}$	character-level input	$c_{i,j}$	j-th char in x_i
$c_{i,\_}$	space after x_i	$c_{0,\_}$	space before x_1
$\mathbf{y}$	label sequence	y_i	label of x_i
$\mathbf{f}_i$	output of forward character-level LSTM at $c_{i,\_}$		
$\mathbf{r}_i$	output of backward character-level LSTM at $c_{i,\_}$		
$\mathbf{f^L}_i$	output of forward-to-LM highway unit		
$\mathbf{r^L}_i$	output of backward-to-LM highway unit		
$\mathbf{f^N}_i$	output of forward-to-SL highway unit		
$\mathbf{r^N}_i$	output of backward-to-SL highway unit		
$\mathbf{v}_i$	input of word-level bi-LSTM at x_i		
$\mathbf{z}_i$	output of word-level bi-LSTM at x_i		

Table 2–1 Notation Table.

a significant improvement over the state-of-the-art. Also, our co-training strategy allows us to capture more useful knowledge with a smaller network, thus yielding much better efficiency without loss of effectiveness.

2.2 LM-LSTM-CRF Framework with Inherent Supervision

The neural architecture of our proposed framework, LM-LSTM-CRF , is visualized in Fig. 2–1. For a sentence with annotations $\mathbf{y} = (y_1, \ldots, y_n)$, its word-level input is marked as $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, where x_i is the i-th word; its character-level input is recorded as $\mathbf{c} = (c_{0,\_}, c_{1,1}, c_{1,2}, \ldots, c_{1,\_}, c_{2,1}, \ldots, c_{n,\_})$, where $c_{i,j}$ is the j-th character for word w_i and $c_{i,\_}$ is the space character after w_i. These notations are also summarized in Table 2–1.

Now, we first discuss the multi-task learning strategy and then introduce the architecture in a bottom-up fashion.

2.2.1 Multi-task Learning Strategy

As shown in Fig. 2–1, our language model and sequence labeling model share the same character-level layer, which fits the setting of multi-task learning and transfer learning. However, different from typical models of this setting, our two tasks are not strongly related. This discordance makes our problem more challenging. E.g., although a naive co-training setting, which directly uses the output from character-level layers, could be effective in several scenarios[27], for our two tasks, it would hurt the performance. This phenomenon would be further discussed in the experiment section.

To mediate these two tasks, we transform the output of character-level layers into different semantic spaces for different objectives. This strategy allows character-level layers to focus on general feature extraction and lets the transform layers select task-specific features. Hence, our language model can provide related knowledge to the sequence labeling, without forcing it to share the whole feature space.

2.2.2 Character-level Layer

Character-level neural language models are trained purely on unannotated sequence data but can capture the underlying style and structure. For example, it can mimic Shakespeare's writing and generate sentences of similar styles, or even master the grammar of programming languages (e.g., XML, LaTeX, and C) and generate syntactically correct codes[28]. Accordingly, we adopted the character-level Long Short Term Memory (LSTM) networks to process character-level input. Aiming to capture lexical features instead of remembering words' spelling, we adjust the prediction from the next character to the next word. As in Fig. 2–1, the character-level LSTM would only make predictions for the next word at word boundaries (i.e., space characters or $c_{i,\_}$).

Furthermore, we coupled two LSTM units to capture information in both forward and backward directions. Although it seems similar to the bi-LSTM unit, the outputs of these two units are processed and aligned differently. Specifically, we record the output of forward LSTM at $c_{i,\_}$ as $\mathbf{f}_i$, and the output of backward LSTM at $c_{i,\_}$ as $\mathbf{r}_i$.

2.2.3 Highway Layer

In computer vision, Convolutional Neural Networks (CNN) has been proved to be an effective feature extractor, but its output needs to be further transformed by fully-connected layers to achieve the state-of-the-art. Bearing this in mind, it becomes natural to stack additional layers upon the flat character-level LSTMs. More specifically, we employ highway units[26], which allow unimpeded information flowing across several layers. Typically, highway layers conduct nonlinear transformation as $\mathbf{m} = H(\mathbf{n}) = \mathbf{t} \odot g(W_H \mathbf{n} + b_H) + (1 - \mathbf{t}) \odot \mathbf{n}$, where $\odot$ is element-wise product, $g(\cdot)$ is a nonlinear transformation such as ReLU in our experiments, $\mathbf{t} = \sigma(W_T \mathbf{n} + b_T)$ is called transform gate and $(1 - \mathbf{t})$ is called carry gate.

In our final architecture, there are four highway units, named `forward-to-LM`, `forward-to-SL`, `backward-to-LM`, and `backward-to-SL`. The first two transfer $\mathbf{f}_i$ into $\mathbf{f^L}_i$ and $\mathbf{f^N}_i$, and the last two transfer $\mathbf{r}_i$ into $\mathbf{r^L}_i$ and $\mathbf{r^N}_i$. $\mathbf{f^L}_i$ and $\mathbf{r^L}_i$ are used in the language model, while $\mathbf{f^N}_i$ and $\mathbf{r^N}_i$ are used in the sequence labeling.

2.2.4 Word-level Layer

Bi-LSTM is adopted as the word-level structure to capture information in both directions. As shown in Fig. 2–1, we concatenate $\mathbf{f^N}_i$ and $\mathbf{r^N}_{i-1}$ with word embeddings and then feed them into the bi-LSTM. Note that, in the backward character-level LSTM, $\mathbf{c}_{i-1,\_}$ is the space character before word x_i, therefore, $\mathbf{f^N}_i$ would be aligned and concatenated with $\mathbf{r^N}_{i-1}$ instead of $\mathbf{r^N}_i$. For example, in Fig. 2–1, the word embeddings of 'Pierre' will be concatenated with the output of the `forward-to-SL` over '...Pierre_' and the output of the `backward-to-SL` over '...erreiP_'.

As to word-level knowledge, we chose to fine-tune pre-trained word embeddings, instead of co-training the whole word-level layer. This is because most parameters of our word-level model come from word embeddings, and fine-tuning pre-trained word embeddings have been verified to be effective in leveraging word-level knowledge[16]. Besides, current word embedding methods can easily scale to the large corpus; pre-trained word embeddings are available in many languages and domains[29]. However, this strategy cannot be applied to character-level layers, since the embedding layer of character-level layers contains very few parameters. Based on these considerations, we applied different strategies to leverage word-level knowledge from character-level.

2.2.5 CRF for Sequence Labeling

Label dependencies are crucial for sequence labeling tasks. For example, in NER task with BIOES annotation, it is not only meaningless but illegal to annotate `I-PER` after `B-ORG` (i.e., mixing the person and the organization). Therefore, jointly decoding a chain of labels can ensure the resulting label sequence to be meaningful. Conditional random field (CRF) has been included in most state-of-the-art models to capture such information and further avoid generating illegal annotations. Consequently, we build a CRF layer upon the word-level LSTM.

For training instance $(\mathbf{x}_i, \mathbf{c}_i, \mathbf{y}_i)$, we suppose the output of word-level LSTM is $\mathbf{Z}_i = (\mathbf{z}_{i,1}, \mathbf{z}_{i,2}, \ldots, \mathbf{z}_{i,n})$. CRF models describe the probability of generating the whole label sequence with regard to $(\mathbf{x}_i, \mathbf{c}_i)$ or $\mathbf{Z}$. That is, $p(\hat{\mathbf{y}}|\mathbf{x}_i, \mathbf{c}_i)$ or $p(\hat{\mathbf{y}}|\mathbf{Z})$, where $\hat{\mathbf{y}} = (\hat{y}_1, \ldots, \hat{y}_n)$ is a generic label sequence. Similar to[16], we define this probability as follows.

$$p(\hat{\mathbf{y}}|\mathbf{x}_i, \mathbf{c}_i) = \frac{\prod_{j=1}^{n} \phi(\hat{y}_{j-1}, \hat{y}_j, \mathbf{z}_j)}{\sum_{\mathbf{y}' \in \mathbf{Y}(\mathbf{Z})} \prod_{j=1}^{n} \phi(y'_{j-1}, y'_j, \mathbf{z}_j)} \tag{2–1}$$

Here, $\mathbf{Y}(\mathbf{Z})$ is the set of all generic label sequences, $\phi(y_{j-1}, y_j, \mathbf{z}_j) = \exp(W_{y_{j-1},y_j}\mathbf{z}_i + b_{y_{j-1},y_j})$, where W_{y_{j-1},y_j} and b_{y_{j-1},y_j} are the weight and bias parameters corresponding to the label pair (y_{j-1}, y_j).

For training, we minimize the following negative log-likelihood.

$$\mathcal{J}_{CRF} = -\sum_{i} \log p(\mathbf{y}_i|\mathbf{Z}_i) \tag{2–2}$$

And for testing or decoding, we want to find the optimal sequence $\mathbf{y}^*$ that maximizes the likelihood.

$$\mathbf{y}^* = \arg\max_{\mathbf{y}\in\mathbf{Y}(\mathbf{Z})} p(\mathbf{y}|\mathbf{Z}) \tag{2–3}$$

Although the denominator of Eq. 2–1 is complicated, we can calculate Eqs. 2–2 and 2–3 efficiently by the Viterbi algorithm.

2.2.6 Neural Language Model

The language model is a family of models describing the generation of sequences. In a neural language model, the generation probability of the sequence $\mathbf{x} = (x_1, ..., x_n)$ in the forward direction (i.e., from left to right) is defined as

$$p_f(x_1, ..., x_n) = \prod_{i=1}^{N} p_f(x_i|x_1, \ldots, x_{i-1})$$

where $p_f(x_i|x_1, \ldots, x_{i-1})$ is computed by NN.

In this chapter, our neural language model makes predictions for words but takes the character sequence as input. Specifically, we would calculate $p_f(x_i|c_{0,\_}, \ldots, c_{i-1,1}, \ldots, c_{i-1,\_})$ instead of $p_f(x_i|x_1, \ldots, x_{i-1})$. This probability is assumed as

$$p_f(x_i|c_{0,\_}, \ldots, c_{i-1,\_}) = \frac{\exp(\mathbf{w}_{x_i}^T \mathbf{f^N}_{i-1})}{\sum_{\hat{x}_j} \exp(\mathbf{w}_{\hat{x}_j}^T \mathbf{f^N}_{i-1})}$$

where $\mathbf{w}_{x_i}$ is the weight vector for predicting word x_i. In order to extract knowledge in both directions, we also adopted a reversed-order language model, which calculates the generation

Dataset	# of Sentences		
	Train	Dev	Test
CoNLL03 NER	14,987	3,466	3,684
CoNLL00 chunking	7,936	1,000	2,012
WSJ	38,219	5,527	5,426

Table 2–2 Dataset summary.

probability from right to left as

$$p_r(x_1, ..., x_n) = \prod_{i=1}^{N} p_r(x_i | c_{i+1,\_}, \ldots, c_{n,\_})$$

$$\text{where } p_r(x_i | c_{i+1,\_}, \ldots, c_{n,\_}) = \frac{\exp(\mathbf{w}_{x_i}^T \mathbf{r}^{\mathbf{N}}{}_i)}{\sum_{\hat{x}_j} \exp(\mathbf{w}_{\hat{x}_j}^T \mathbf{r}^{\mathbf{N}}{}_i)}$$

The following negative log likelihood is applied as the objective function of our language model.

$$\mathcal{J}_{LM} = -\sum_i \log p_f(\mathbf{x}_i) - \sum_i \log p_r(\mathbf{x}_i) \tag{2–4}$$

2.2.7 Joint Model Learning

By combining Eqs. 2–2 and 2–4, we can write the joint objective function as

$$\mathcal{J} = -\sum_i \Big(p(\mathbf{y}_i | \mathbf{Z}_i) + \lambda \big(\log p_f(\mathbf{x}_i) + \log p_r(\mathbf{x}_i) \big) \Big) \tag{2–5}$$

where λ is a weight parameter. In our experiments, λ is always set to 1 without any tuning.

In order to train the neural network efficiently, stochastic optimization has been adopted. And at each iteration, we sample a batch of training instances and perform an update according to the summand function of Eq. 2–5: $p(\mathbf{y}_i | \mathbf{Z}_i) + \lambda \big(\log p_f(\mathbf{x}_i) + \log p_r(\mathbf{x}_i) \big)$

2.3 Experiments

Here, we evaluate LM-LSTM-CRF on three benchmark datasets: the CoNLL 2003 NER dataset[30], the CoNLL 2000 chunking dataset[31], and the Wall Street Journal portion of Penn Treebank dataset (WSJ)[32].

- **CoNLL03 NER** contains annotations for four entity types: `PER`, `LOC`, `ORG`, and `MISC`. It has been separated into training, development and test sets.

- **CoNLL00 chunking** defines eleven syntactic chunk types (e.g., NP, VP) in addition to Other. It only includes training and test sets. Following previous works[24], we sampled 1000 sentences from training set as a held-out development set.
- **WSJ** contains 25 sections and categorizes each word into 45 POS tags. We adopt the standard split and use sections 0-18 as training data, sections 19-21 as development data, and sections 22-24 as test data[33].

The corpus statistics are summarized in Table 2–2. We report the accuracy for the WSJ dataset. And in the first two datasets, we adopt the official evaluation metric (micro-averaged F_1), and use the BIOES scheme[34]. Also, in all three datasets, rare words (i.e., frequency less than 5) are replaced by a special token (<UNK>).

2.3.1 Network Training

For a fair comparison, we didn't spend much time on tuning parameters but borrow the initialization, optimization method, and all related hyper-parameter values (except the state size of LSTM) from the previous work[16]. For the hidden state size of LSTM, we expand it from 200 to 300, because introducing additional knowledge allows us to train a larger network. We will further discuss this change later. Since the CoNLL00 is similar to the CoNLL03 NER dataset, we conduct experiments with the same parameters on both tasks.

Initialization. We use GloVe 100-dimension pre-trained word embeddings released by Stanford[1] and randomly initialize the other parameters[35, 36].

Optimization. We employ mini-batch stochastic gradient descent with momentum. The batch size, the momentum and the learning rate are set to 10, 0.9 and $\eta_t = \frac{\eta_0}{1+\rho t}$, where η_0 is the initial learning rate and $\rho = 0.05$ is the decay ratio. Dropout is applied in our model, and its ratio is fixed to 0.5. To increase stability, we use gradient clipping of 5.0.

Network Structure. The hyper-parameters of character-level LSTM are set to the same value of word-level bi-LSTM. We fix the depth of highway layers as 1 to avoid an over-complicated model.

Note that some baseline methods (e.g.,[24, 37]) incorporate the development set as a part of training. However, because we are using early stopping based on the evaluation on the development set, our model is trained purely on the training set.

[1] http://nlp.stanford.edu/projects/glove/

2.3.1.1 Compared Methods

We consider three classes of baseline sequence labeling methods in our experiments.

- **Sequence Labeling Only.** Without any additional supervision or extra resources, LSTM-CRF[23] and LSTM-CNN-CRF[16] are the current state-of-art methods. We also list some top reported performance on each dataset[19, 24, 27, 33, 37–40].
- **Joint Model with Other Supervised Tasks.** There are several attempts[19, 27] to enhance sequence labeling tasks by introducing additional annotations from other related tasks (e.g., enhance NER with entity linking labels).
- **Joint Model with Language Model**: Language models have been employed by some recent works to extract knowledge from raw text and thus enhancing sequence labeling task. TagLM[24] leverages pre-trained language models and shows the effectiveness with the large external corpus, but the large model scale and long training time make it hard to re-run this model. Another work[25] also incorporates the sequence labeling task with the language model.

For comparison, we tune the parameters of three most related baselines[16, 23, 25][1]. , and report the statics of the best working parameter setting. Besides, we index these models by number, and summarize the results in Tables 2–3, 2–4 and 2–6.

2.3.2 Performance Comparison

In this section, we focus on the comparisons between LM-LSTM-CRF and previous state-of-the-arts, including both effectiveness and efficiency. As demonstrated in Tables 2–3, 2–4 and 2–6, LM-LSTM-CRF significantly outperforms all baselines without additional resources. Moreover, even for those baselines with extra resources, LM-LSTM-CRF beats most of them and is only slightly worse than TagLM (index 4)[24].

TagLM (index 4) is equipped with both extra corpoa (about 4000X larger than the CoNLL03 NER dataset) and a tremendous pre-trained forward language model (`4096-8192-1024`[2])[41]. Due to the expensive resources and time required by `4096-8192-1024`, even the authors of TagLM failed to train a backward language model of the same size, instead, chose a much

[1] Implementations: `https://github.com/xuezhemax/lasagnenlp` (Ma et al. 2016), `https://github.com/glample/tagger` (Lample et al. 2016) and `https://github.com/marekrei/sequence-labeler` (Rei 2017)

[2] `4096-8192-1024` is composed of character-level CNN with 4096 filters, 2 layers of stacked LSTMs with 8192 hidden units each and a 1024-dimension projection unit.

Extra Resource	Index & Model	F_1 score	
		Type	Value (±std)
gazetteers	0) Collobert et al. 2011†	reported	89.59
	1) Chiu et al. 2016†	reported	91.62±0.33
AIDA dataset	2) Luo et al. 2015	reported	91.20
CoNLL 2000 / PTB-POS dataset	3) Yang et al. 2017†	reported	91.26
1B Word dataset & `4096-8192-1024`	4) Peters et al. 2017†‡	reported	91.93±0.19
1B Word dataset	5) Peters et al. 2017†‡	reported	91.62±0.23
None	6) Collobert et al. 2011†	reported	88.67
	7) Luo et al. 2015	reported	89.90
	8) Chiu et al. 2016†	reported	90.91±0.20
	9) Yang et al. 2017†	reported	91.20
	10) Peters et al. 2017†	reported	90.87±0.13
	11) Peters et al. 2017†‡	reported	90.79±0.15
	12) Rei 2017 †‡	mean	87.38±0.36
		max	87.94
		reported	86.26
	13) Lample et al. 2016†	mean	90.76±0.08
		max	91.14
		reported	90.94
	14) Ma et al. 2016†	mean	91.37±0.17
		max	91.67
		reported	91.21
	15) LM-LSTM-CRF †‡	mean	91.71±0.10
		max	91.85

Table 2–3 F_1 score on the CoNLL03 NER dataset. We mark models adopting pre-trained word embedding as †, and record models which leverage language models as ‡.

smaller one (`LSTM-2048-512`[1]). It is worth noting that, when either extra corpus or `4096-`

[1] `LSTM-2048-512` is composed of a single-layer LSTM with 2048 hidden units and a 512-dimension projection unit.

`8192-1024` is absent, LM-LSTM-CRF shows significant improvements over TagLM (index 5, 10 and 11).

Also, LSTM-CNN-CRF outperforms LSTM-CRF in our experiments, which is different from[42]. During our experiments, we discover that, when trained on CPU, LSTM-CNN-CRF only reaches 90.83 F_1 score on the NER dataset, but gets 91.37 F_1 score when trained on GPU. We conjecture that this performance gap is due to the difference of runtime environments. Therefore, we conduct all of our experiments on GPU. Additionally, we can observe that, although co-trained with language model, results of index 12 fails to outperform LSTM-CNN-CRF or LSTM-CRF. The reason of this phenomenon could be complicated and beyond the scope of this chapter. However, it verified the effectiveness of our method, and demonstrated the contribution of outperforming these baselines.

2.3.2.1 NER

First of all, we have to point out that the results of index 1, 4, 8, 10 and 11 are not directly comparable with others since their final models are trained on both training and development set, while others are trained purely on the training set. As mentioned before, LM-LSTM-CRF outperforms all baselines except TagLM (index 4). For a thorough comparison, we also compare to its variants, TagLM (index 5), TagLM (index 10) and TagLM (index 11). Both index 10 and 11 are trained on the CoNLL03 dataset alone, while index 11 utilizes language model and index 10 doesn't. Comparing F_1 scores of these two settings, we can find that TagLM (index 11) even performs worse than TagLM (index 10) , which reveals that directly applying co-training might hurt the sequence labeling performance. We will also discuss this challenge later in the Highway Layers & Co-training section.

Besides, changing the forward language model from `4096-8192-1024` to `LSTM-2048-512`, TagLM (index 5) gets a lower F_1 score of 91.62±0.23. Comparing this score to ours (91.71±0.10), one can verify that pre-trained language model usually extracts a large portion of unrelated knowledge. Relieving such redundancy by guiding the language model with task-specific information, our model is able to conduct both effective and efficient learning.

2.3.2.2 POS Tagging

Similar to the NER task, LM-LSTM-CRF outperforms all baselines on the WSJ portion of the PTB POS tagging task. Although the improvements over LSTM-CRF and CNN-LSTM-CRF are less obvious than those on the CoNLL03 NER dataset, considering the fact that the POS

Ind & Model	Accuracy	
	Type	Value (±std)
0) Collobert et al. 2011†	reported	97.29
16) Manning 2011	reported	97.28
17) Søgaard 2011	reported	97.50
18) Sun 2014	reported	97.36
12) Rei 2017†‡	mean	96.97±0.22
	max	97.14
	reported	97.43
13) Lample et al. 2016†	mean±std	97.35±0.09
	maximum	97.51
14) Ma et al. 2016†	mean±std	97.42±0.04
	maximum	97.46
	reported	97.55
15) LM-LSTM-CRF †‡	mean±std	97.53±0.03
	maximum	97.59

Table 2–4 Accuracy on the WSJ dataset. We mark models adopting pre-trained word embedding as †, and record models which leverage language models as ‡.

Model	CoNLL03 NER		WSJ POS		CoNLL00 Chunking	
	h	F_1Score	h	Accuracy	h	F_1Score
LSTM-CRF	46	90.76	37	97.35	26	94.37
LSTM-CNN-CRF	7	91.22	21	97.42	6	95.80
LM-LSTM-CRF	6	91.71	16	97.53	5	95.96
LSTM-CRF*	4	91.19	8	97.44	2	95.82
LSTM-CNN-CRF*	3	90.98	7	96.98	2	95.51

Table 2–5 Training statistics of TagLM (index 4 and 5) and LM-LSTM-CRF on the CoNLL03 NER dataset.

tagging task is believed to be easier than the NER task and current methods have achieved relatively high performance, this improvement could still be viewed as significant. Moreover, it is worth noting that for both NER and POS tagging tasks, LM-LSTM-CRF achieves not only higher F_1 scores, but also with smaller variances, which further verifies the superiority of our framework.

2.3.2.3 Chunking

In the chunking task, LM-LSTM-CRF also achieves relatively high F_1 scores, but with slightly higher variances. Considering the fact that this corpus is much smaller than the other two (only about 1/5 of WSJ or 1/2 of CoNLL03 NER), we can expect more variance due to the lack of training data. Still, LM-LSTM-CRF outperforms all baselines without extra resources, and most of the baselines trained with extra resources.

2.3.2.4 Efficiency

We implement LM-LSTM-CRF[1] based on the PyTorch library[2]. Models has been trained on one GeForce GTX 1080 GPU, with training time recorded in Table 2–7.

In terms of efficiency, the language model component in LM-LSTM-CRF only introduces a small number of parameters in two highway units and a soft-max layer, which may not have a very large impact on the efficiency. To control variables like infrastructures, we further re-implemented both baselines, and report their performance together with original implementations. From the results, these re-implementations achieve better efficiency comparing to the original ones, but yield relative worse performance. Also, LM-LSTM-CRF achieves the best performance, and takes twice the training time of the most efficient model, LSTM-CNN-CRF*. Empirically, considering the difference among the implementations of these models, we think these methods have roughly the same efficiency.

Besides, we list the required time and resources for pre-training model index 4 and 5 on the NER task in Table 2–5[41]. Comparing to these language models pre-trained on external corpus, our model has no such reliance on extensive corpus, and can achieve similar performance with much more concise model and efficient training. It verifies that our LM-LSTM-CRF model can effectively leverage the language model to extract task-specific knowledge to empower sequence labeling.

[1] https://github.com/LiyuanLucasLiu/LM-LSTM-CRF

[2] http://pytorch.org/

Extra Resource	Ind & Model	F$_1$ score	
		Type	Value (±std)
PTB-POS	19) Hashimoto et al. 2016†	reported	95.77
	20) Søgaard et al. 2016†	reported	95.56
CoNLL 2000 / PTB-POS dataset	3)Yang et al. 2017†	reported	95.41
1B Word dataset	4) Peters et al. 2017†‡	reported	96.37±0.05
None	21) Hashimoto et al. 2016†	reported	95.02
	22) Søgaard et al. 2016†	reported	95.28
	9) Yang et al. 2017†	reported	94.66
	12) Rei 2017†‡	mean	94.24±0.11
		max	94.33
		reported	93.88
	13) Lample et al. 2016†	mean	94.37±0.07
		maximum	94.49
	14) Ma et al. 2016†	mean	95.80±0.13
		maximum	95.93
	15) LM-LSTM-CRF †‡	mean	95.96±0.08
		maximum	96.13

Table 2–6 F$_1$ score on the CoNLL00 chunking dataset. We mark models adopting pre-trained word embedding as †, and record models which leverage language models as ‡.

Ind & Model	F$_1$score	Module	Time · Device	
15) LM-LSTM-CRF	91.71	total	6	h·GTX 1080
5) Peters et al. 2017	91.62	LSTM-2048-512	320	h·Telsa K40
		LSTM-2048-512	320	h·Telsa K40
4) Peters et al. 2017	91.93	4096-8192-1024	14112	h·Telsa K40
		LSTM-2048-512	320	h·Telsa K40

Table 2–7 Training time and performance of LSTM-CRF, LSTM-CNN-CRF and LM-LSTM-CRF on three datasets. Our re-implementations are marked with ⋆

Model	State Size	F_1score±std	Recall±std	Precision±std
LM-LSTM-CRF	300	91.71±0.10	92.14±0.12	91.30±0.13
	200	91.63±0.23	92.07±0.22	91.19±0.30
	100	91.13±0.32	91.60±0.37	90.67±0.32
LSTM-CRF	300	90.76±0.08	90.82±0.08	90.69±0.08
	200	90.41±0.07	90.63±0.07	90.20±0.07
	100	90.74±0.22	91.08±0.50	90.42±0.17
LSTM-CNN-CRF	300	91.22±0.19	91.70±0.16	90.74±0.27
	200	91.37±0.17	91.08±0.53	90.58±0.11
	100	91.18±0.10	91.56±0.16	90.81±0.15

Table 2–8 Effect of hidden state size of LSTM

2.3.3 Analysis

To analyze the performance of LM-LSTM-CRF , we conduct additional experiments on the CoNLL03 NER dataset.

2.3.3.1 Hidden State Size

To explore the effect of model size, we train our model with different hidden state sizes. For comparison, we also apply the same hidden state sizes to LSTM-CRF and LSTM-CNN-CRF. From Table 2–8, one can easily observe that the F_1 score of LM-LSTM-CRF keeps increasing when the hidden state size grows, while LSTM-CNN-CRF has a peak at state size 200 and LSTM-CRF has a drop at state size 200. This phenomenon further verified our intuition of employing the language model to extract knowledge and prevent overfitting.

2.3.3.2 Highway Layers & Co-training

To elucidate the effect of language model[1] and highway units, we compare LM-LSTM-CRF with its two variants, LM-LSTM-CRF_NL and LM-LSTM-CRF_NH . The first keeps highway units, but optimizes $\mathcal{J}_{CRF}$ alone; the second jointly optimizes $\mathcal{J}_{CRF}$ and $\mathcal{J}_{LM}$, but without highway units. As shown in Table 2–9, LM-LSTM-CRF_NH yields worse performance than LM-LSTM-CRF_NL . This observation accords with previous comparison between TagLM (index 10) and

[1] the perplexities of the forward language model on CoNLL03 NER's training / development / test sets are 52.87 / 55.03 / 50.22.

State Size	Model	F_1score±std	Recall±std	Precision±std
300	LM-LSTM-CRF	91.71±0.10	92.14±0.12	91.30±0.13
	LM-LSTM-CRF_NL	91.43±0.09	91.85±0.18	91.01±0.19
	LM-LSTM-CRF_NH	91.16±0.22	91.67±0.28	90.66±0.23
200	LM-LSTM-CRF	91.63±0.23	92.07±0.22	91.19±0.30
	LM-LSTM-CRF_NL	91.44±0.10	91.95±0.16	90.94±0.16
	LM-LSTM-CRF_NH	91.34±0.28	91.79±0.18	90.89±0.30
100	LM-LSTM-CRF	91.13±0.32	91.60±0.37	90.67±0.32
	LM-LSTM-CRF_NL	91.17±0.11	91.72±0.14	90.61±0.21
	LM-LSTM-CRF_NH	91.01±0.19	91.50±0.21	90.53±0.30

Table 2–9 Effect of language model and highway

TagLM (index 11) on the CoNLL03 NER dataset. We conjecture that it is because the NER task and the language model is not strongly related to each other. In summary, our proposed co-training strategy is effective and introducing the highway layers is necessary.

2.4 Related Work

There exist two threads of related work regarding the topics in this chapter, which are sequence labeling and how to improve it with additional information.

Sequence Labeling. As one of the fundamental tasks in NLP, linguistic sequence labeling, including POS tagging, chunking, and NER, has been studied for years. Handcrafted features were widely used in traditional methods like CRFs, HMMs, and maximum entropy classifiers[45–48], but also make it hard to apply them to new tasks or domains. Recently, getting rid of handcrafted features, there are attempts to build end-to-end systems for sequence labeling tasks, such as BiLSTM-CNN[37], LSTM-CRF[23], and the current state-of-the-art method in NER and POS tagging tasks, LSTM-CNN-CRF[16]. These models all incorporate character-level structure, and report meaningful improvement over pure word-level model. Also, CRF layer has also been demonstrated to be effective in capturing the dependency among labels. Our model is based on the success of LSTM-CRF model and is further modified to better capture the char-level information in a language model manner.

Leveraging Additional Information. Integrating word-level and character-level knowledge has been proved to be helpful to sequence labeling tasks. For example, word embeddings[21, 22] can be utilized by co-training or pre-training strategies[18, 23]. However, none of these models utilizes the character-level knowledge. Although directly adopting character-level pre-trained language models could be helpful[24]. Such pre-trained knowledge is not task-specific and requires a larger neural network, external corpus, and longer training. Our model leverages both word-level and character-level knowledge through a co-training strategy, which leads to a concise, effective, and efficient neural network. Besides, unlike other multi-task learning methods, our model has no reliance on any extra annotation[24] or any knowledge base[49]. Instead, it extracts knowledge from the self-contained order information.

Chapter 3 Named Entities Recognition in Social Media

3.1 Introduction

Named entity recognition (NER) is one of the first and most important steps in Information Extraction pipelines. Generally, it is to identify mentions of entities (persons, locations, organizations, etc.) within unstructured text. However, the diverse and noisy nature of user-generated content as well as the emerging entities with novel surface forms make NER in social media messages more challenging.

The first challenge brought by user-generated content is its unique characteristics: short, noisy and informal. For instance, tweets are typically short since the number of characters is restricted to 140 and people indeed tend to pose short messages even in social media without such restrictions, such as YouTube comments and Reddit. [1] Hence, the contextual information in a sentence is very limited. Apart from that, the use of colloquial language makes it more difficult for existing NER approaches to be reused, which mainly focus on a general domain and formal text[50, 51]. State-of-the-art NER softwares (e.g. Standford Corenlp) are less effective on such social media messages[51]. Due to the informal and contemporary nature of these micro-posts, performance still lags far behind that on formal text genres such as newswire.

Another challenge of NER in noisy text is the fact that there are large amounts of emerging named entities and rare surface forms among the user-generated text, which tend to be tougher to detect[52] and recall thus is a significant problem[51]. By way of example, the surface form "*kktny*", in the tweet "so.. *kktny* in 30 mins?", actually refers to a new TV series called "*Kourtney and Kim Take New York*", which even human experts found hard to recognize. Additionally, it is quite often that netizens mention entities using rare morphs as surface forms. For example, "*black mamba*", the name for a venomous snake, is actually a morph that Kobe Bryant created for himself for his aggressiveness in playing basketball games[53]. Such morphs and rare surface forms are also very difficult to detect and classify.

This task will evaluate the ability to detect and classify novel, emerging, singleton named entities in noisy text. Detecting commonly-mentioned entities tends to be easier than the rarer, more unusual surface forms. Similarly, entities with unusual surface forms, or that are simply rare, tend to be tougher to detect[52], with recall being a significant problem in rapidly-

[1] The average length of the sentences in this shared task is about 20 tokens per sentence.

changing text types[51]. However, the entities that are common in newly-emerging texts such as newswire or social media are often new, not having been mentioned in prior datasets. This poses a challenge to NER systems, where in many deployments, unusual, previously-unseen entities need to be detected reliably and with high recall. In the shared task, we are provided with turbulent data containing few repeated entities, drawn from rapidly-changing text types or sources of non-mainstream entities.

The goal of this chapter is to present our system participating in the *Novel and Emerging Named Entity Recognition* shared task at the EMNLP 2017 Workshop on Noisy User-generated Text (W-NUT 2017), which aims for NER in such noisy user-generated text. We investigate a multi-channel BiLSTM-CRF neural network model in our participating system, which is described in Section 3.3. The details of our implementation are in presented in Section 3.4, where we also present some conclusion from our experiments.

3.2 Problem Definition

The NER is a classic sequence labeling problem, in which we are given a sentence, in the form of a sequence of tokens $\mathbf{w} = (w_1, w_2, ..., w_n)$, and we are required to output a sequence of token labels $\mathbf{y} = (y_1, y_2, ..., y_n)$. In this specific task, we use the standard BIO2 annotation, and each named entity chunk are classified into 6 categories, namely Person, Location (including GPE, facility), Corporation, Consumer good (tangible goods, or well-defined services), Creative work (song, movie, book, and so on) and Group (subsuming music band, sports team, and non-corporate organizations).

3.3 Combining linguistic structures as indirect supervision

In this section, we will first introduce the overview of our proposed model and then present each part of the model in detail.

3.3.1 Overview

Figure 3–1 shows the overall structure of our proposed model, instead of solely using the original pretrained word embeddings as the final word representations, we construct a comprehensive word representation for each word in the input sentence. This comprehensive word representations contain the character-level sub-word information, the original pretrained word embeddings and multiple syntactical features. Then, we feed them into a Bidirectional LSTM layer, and thus we have a hidden state for each word. The hidden states are considered as the feature vectors of the words by the final CRF layer, from which we can decode the final predicted tag sequence for the input sentence.

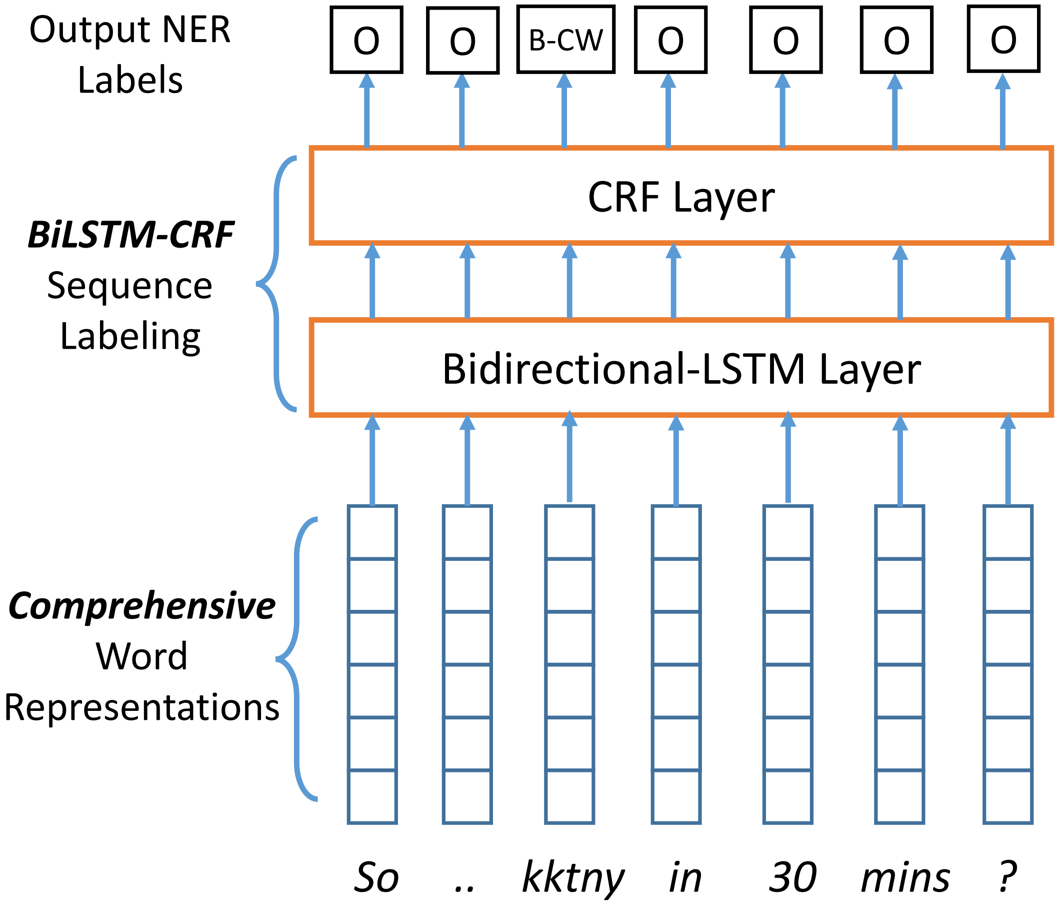


Figure 3–1 Overview of our approach.

3.3.2 Comprehensive Word Representations

In this subsection, we present our proposed comprehensive word representations. We first build character-level word representations from the embeddings of every character in each word using a bidirectional LSTM. Then we further incorporate the final word representation with the embedding of the syntactical information of each token, such as the part-of-speech tag, the dependency role, the word position in the sentence and the head position. Finally, we combine the original word embeddings with the above two parts to obtain the final comprehensive word representations.

3.3.2.1 Character-level Word Representations

In noisy user-generated text analysis, sub-word (character-level) information is much more important than that in normal text analysis for two main reasons: 1) People are more likely to use novel abbreviations and morphs to mention entities, which are often out of vocabulary and only occur a few times. Thus, solely using the original word-level word embedding as features to represent words is not adequate to capture the characteristics of such mentions. 2) Another reason why we have to pay more attention to character-level word representation for noisy text is that it is can capture the orthographic or morphological information of both formal words and Internet slang.

There are two main network structures to make use of character embeddings: one is CNN[54] and the other is BiLSTM[55]. BiLSTM turns to be better in our experiment on development dataset. Thus, we follow Lample et al. ([55]) to build a BiLSTM network to encode the characters in each token as Figure 3–2 shows. We finally concatenate the forward embedding and backward embedding to the final character-level word representation.

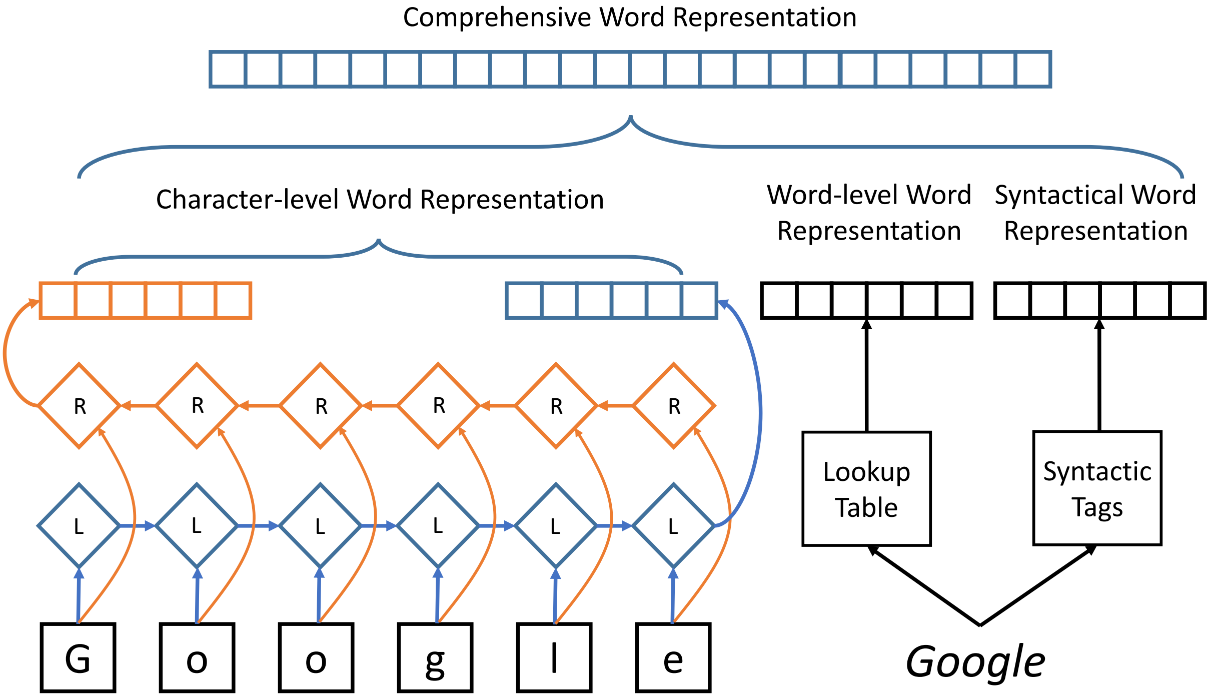

Comprehensive Word Representation
Character-level Word Representation
Word-level Word Representation
Syntactical Word Representation
R
L
G
o
o
g
l
e
Lookup Table
Syntactic Tags
Google

Figure 3–2 Illustration of comprehensive word representations.

3.3.2.2 Syntactical Word Representations

We argue that the syntactical information, such as POS tags and dependency roles, should also be explicitly considered as contextual features of each token in the sentence.

TweetNLP and TweeboParser[56, 57] are two popular software to generate such syntactical tags for each token given a tweet. Given the nature of the noisy tweet text, a new set of POS tags and dependency trees are used in the tool, called Tweebank[58]. See Table 3–1 for an example POS tagging. Since a tweet often contains more than one utterance, the output of TweeboParser will often be a multi-rooted graph over the tweet.

Word position embedding are included as well as it is widely used in other similar tasks, like relation classification[59]. Also, head position embeddings are taken into account while calculating these embedding vectors to further enrich the dependency information. It tries to exclude these tokens from the parse tree, resulting a head index of -1.

After calculating all 4 types of embedding vectors (POS tags, dependency roles, word positions, head positions) for every tokens, we concatenate them to form a syntactical word representation.

Table 3–1 Example of POS tagging for tweets.

Token	so	..	kktny	in	30	mins	?
POS	R	,	N	P	$	N	,
Position	1	2	3	4	5	6	7
Head	0	-1	0	3	6	4	-1

3.3.2.3 Combination with Word-level Word Representations

After obtaining the above two additional word representations, we combine them with the original word-level word representations, which are just traditional word embeddings.

To sum up, our comprehensive word representations are the concatenation of three parts: 1) character-level word representations, 2) syntactical word representation and 3) original pre-trained word embeddings.

3.3.3 BiLSTM Layer

LSTM based networks are proven to be effective in sequence labeling problem for they have access to both past and the future contexts. Whereas, hidden states in unidirectional LSTMs only takes information from the past, which may be adequate to classify the sentiment is a shortcoming for labeling each token. Bidirectional LSTMs enable the hidden states to capture both historical and future context information and then to label a token.

Mathematically, the input of this BiLSTM layer is a sequence of comprehensive word representations (vectors) for the tokens of the input sentence, denoted as $(\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n})$. The output of this BiLSTM layer is a sequence of the hidden states for each input word vectors, denoted as $(\mathbf{h_1}, \mathbf{h_2}, ..., \mathbf{h_n})$. Each final hidden state is the concatenation of the forward $\overleftarrow{\mathbf{h_i}}$ and backward $\overrightarrow{\mathbf{h_i}}$ hidden states. We know that

$$\overleftarrow{\mathbf{h_i}} = \text{lstm}(\mathbf{x_i}, \overleftarrow{\mathbf{h_{i-1}}}) \text{ , } \overrightarrow{\mathbf{h_i}} = \text{lstm}(\mathbf{x_i}, \overrightarrow{\mathbf{h_{i+1}}})$$

$$\mathbf{h_i} = \left[\overleftarrow{\mathbf{h_i}} \; ; \; \overrightarrow{\mathbf{h_i}} \right]$$

3.3.4 CRF Layer

It is almost always beneficial to consider the correlations between the current label and neighboring labels since there are many syntactical constrains in natural language sentences. For example, I-PERSON will never follow a B-GROUP. If we simply feed the above mentioned hidden states independently to a Softmax layer to predict the labels, then such constrains will not be more likely to be broken. Linear-chain Conditional Random Field is the most popular way to control the structure prediction and its basic idea is to use a series of potential function to approximate the conditional probability of the output label sequence given the input word sequence.

Formally, we take the above sequence of hidden states $\mathbf{h} = (\mathbf{h_1}, \mathbf{h_2}, ..., \mathbf{h_n})$ as our input to the CRF layer, and its output is our final prediction label sequence $\mathbf{y} = (y_1, y_2, ..., y_n)$, where y_i is in the set of all possible labels. We denote $\mathcal{Y}(\mathbf{h})$ as the set of all possible label sequences. Then we derive the conditional probability of the output sequence given the input hidden state sequence is

$$p(\mathbf{y}|\mathbf{h}; \mathbf{W}, \mathbf{b}) = \frac{\prod_{i=1}^{n} \exp(\mathbf{W}_{y_{i-1},y_i}^{T} \mathbf{h} + \mathbf{b}_{y_{i-1},y_i})}{\sum_{\mathbf{y}' \in \mathcal{Y}(\mathbf{h})} \prod_{i=1}^{n} \exp(\mathbf{W}_{y'_{i-1},y'_i}^{T} \mathbf{h} + \mathbf{b}_{y'_{i-1},y'_i})}$$

, where $\mathbf{W}$ and $\mathbf{b}$ are the two weight matrices and the subscription indicates that we extract the weight vector for the given label pair (y_{i-1}, y_i).

To train the CRF layer, we use the classic maximum conditional likelihood estimation to train our model. The final log-likelihood with respect to the weight matrices is

$$L(\mathbf{W}, \mathbf{b}) = \sum_{(\mathbf{h_i}, \mathbf{y_i})} \log p(\mathbf{y_i} | \mathbf{h_i}; \mathbf{W}, \mathbf{b})$$

Finally, we adopt the Viterbi algorithm for training the CRF layer and the decoding the optimal output sequence $\mathbf{y}^*$.

3.4 Experiments

In this section, we discuss the implementation details of our system such as hyper parameter tuning and the initialization of our model parameters. [1]

3.4.1 Parameter Initialization

For word-level word representation (i.e. the lookup table), we utilize the pretrained word embeddings[2] from GloVe[60]. For all out-of-vocabulary words, we assign their embeddings by randomly sampling from range $\left[-\sqrt{\frac{3}{\text{dim}}}, +\sqrt{\frac{3}{\text{dim}}}\right]$, where *dim* is the dimension of word embeddings, suggested by He et al.([61]). The random initialization of character embeddings are in the same way. We randomly initialize the weight matrices **W** and **b** with uniform samples from $\left[-\sqrt{\frac{6}{r+c}}, +\sqrt{\frac{6}{r+c}}\right]$, r and c are the number of the rows and columns, following Glorot and Bengio([62]). The weight matrices in LSTM are initialized in the same work while all LSTM hidden states are initialized to be zero except for the bias for the forget gate is initialized to be 1.0 , following Jozefowicz et al.([63]).

3.4.2 Hyper Parameter Tuning

We tuned the dimension of word-level embeddings from {50, **100**, 200}, character embeddings from {10, **25**, 50}, character BiLSTM hidden states (i.e. the character level word representation) from {20, **50**, 100}. We finally choose the bold ones. The dimension of part-of-speech tags, dependecny roles, word positions and head positions are all 5.

[1] The detailed description of the evaluation metric and the dataset are shown in `http://noisy-text.github.io/2017/emerging-rare-entities.html`

[2] `http://nlp.stanford.edu/data/glove.twitter.27B.zip`

As for learning method, we compare the traditional SGD and Adam[64]. We found that Adam performs always better than SGD, and we tune the learning rate form {1e-2,**1e-3**,1e-4}.

3.4.3 Results

To evaluate the effectiveness of each feature in our model, we do the feature ablation experiments and the results are shown in Table 3–2.

Table 3–2 Feature Ablation

Features	**F1 (entity)**	**F1 (surface form)**
Word	37.16	34.15
Char(LSTM)+Word	38.24	37.21
POS+Char(LSTM)+Word	40.01	37.57
Syntactical+Char(CNN)+Word	40.12	37.52
Syntactical+Char(LSTM)+Word	**40.42**	**37.62**

In comparison with other participants, the results are shown in Table 3–3.

Table 3–3 Result comparison

Team	**F1 (entity)**	**F1 (surface form)**
Drexel-CCI	26.30	25.26
MIC-CIS	37.06	34.25
FLYTXT	38.35	36.31
Arcada	39.98	37.77
Ours	**40.42**	**37.62**
SpinningBytes	40.78	39.33
UH-RiTUAL	**41.86**	**40.24**

3.5 Related Work

Conditional random field (CRF) is a most effective approaches[65, 66] for NER and other sequence labeling tasks and it achieved the state-of-the-art performance previously in Twitter NER[50]. Whereas, it often needs lots of hand-craft features. More recently, Huang et al. ([67]) introduced a similar but more complex model based on BiLSTM, which also considers hand-crafted features. Lample et al. ([55]) further introduced using BiLSTM to incorporate character-level word representation. Whereas, Ma and Hovy ([54]) replace the BiLSTM to CNN to build the character-level word representation. Limsopatham and Collier ([68]), used similar model and achieved the best performance in the last shared task[69]. Based on the previous work, our system take more syntactical information into account, such as part-of-speech tags, dependency roles, token positions and head positions, which are proven to be effective.

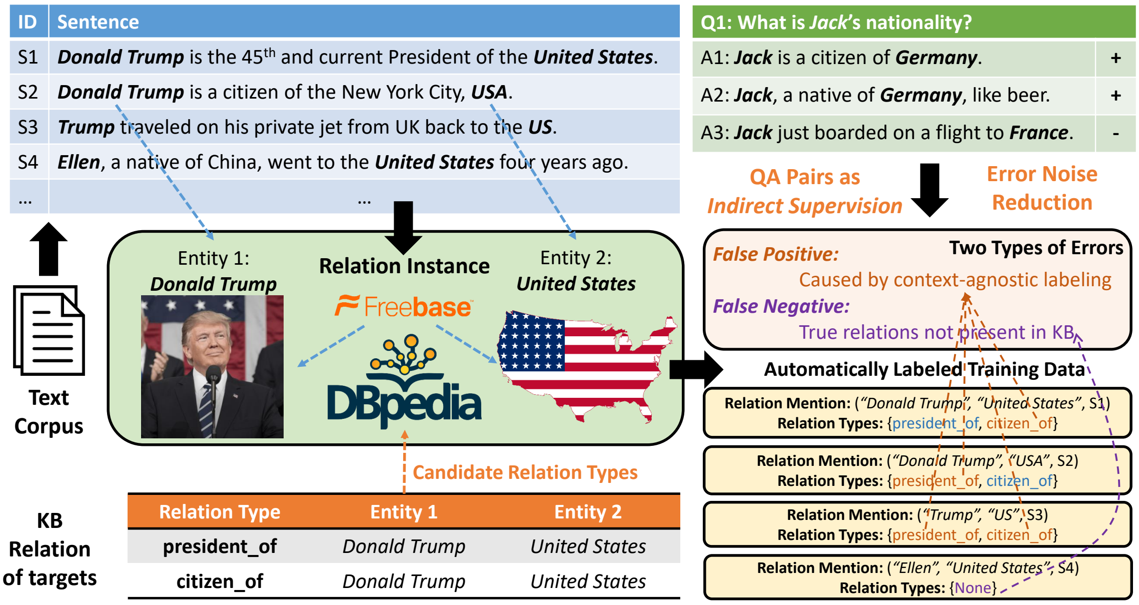
Chapter 4 Relation Extraction for Named Entities

4.1 Introduction

Relation extraction is an important task for understanding massive text corpora by turning unstructured text data into relation triples for further analysis. For example, it detects the relationship "`president_of`" between entities "*Donald Trump*" and "*United States*" in a sentence. Such extracted information can be used for more downstream text analysis tasks (e.g. serving as primitives for information extraction and knowledge base (KB) completion, and assisting question answering systems).

Typically, RE systems rely on training data, primarily acquired via human annotation, to achieve satisfactory performance. However, such manual labeling process can be costly and non-scalable when adapting to other domains (e.g. biomedical domain). In addition, when the number of types of interest becomes large, the generation of handcrafted training data can be error-prone. To alleviate such an exhaustive process, the recent trend has deviated towards the adoption of distant supervision (DS). DS replaces the manual training data generation with a pipeline that automatically links texts to a knowledge base (KB). The pipeline has the following steps: (1) detect entity mentions in text; (2) map detected entity mentions to entities in KB; (3) assign, to the candidate type set of each entity mention pair, all KB relation types between their KB-mapped entities. However, the noise introduced to the automatically generated training data is not negligible. There are two major causes of error: incomplete KB and context-agnostic labeling process. If we treat unlinkable entity pairs as the pool of negative examples, false negatives can be commonly encountered as a result of the insufficiency of facts in KBs, where many true entity or relation mentions fail to be linked to KBs (see example in Figure 4–1). In this way, models counting on extensive negative instances may suffer from such misleading training data. On the other hand, context-agnostic labeling can engender false positive examples, due to the inaccuracy of the DS assumption that if a sentence contains any two entities holding a relation in the KB, the sentence must be expressing such relation between them. For example, entities "*Donald Trump*" and "*United States*" in the sentence "*Donald Trump flew back to United States*" can be labeled as "`president_of`" as well as "`born_in`", although only an out-of-interest relation type "`travel_to`" is expressed explicitly (as shown in Figure 4–1).

To alleviate such exhaustive process, two main lines of work have emerged: weak supervi-



ID	Sentence
S1	***Donald Trump*** is the 45th and current President of the ***United States***.
S2	***Donald Trump*** is a citizen of the New York City, ***USA***.
S3	***Trump*** traveled on his private jet from UK back to the ***US***.
S4	***Ellen***, a native of China, went to the ***United States*** four years ago.
...	...

Q1: What is *Jack*'s nationality?	
A1: ***Jack*** is a citizen of ***Germany***.	+
A2: ***Jack***, a native of ***Germany***, like beer.	+
A3: ***Jack*** just boarded on a flight to ***France***.	-

Relation Type	Entity 1	Entity 2
president_of	*Donald Trump*	*United States*
citizen_of	*Donald Trump*	*United States*

Relation Mention: (*"Donald Trump", "United States"*, S1) Relation Types: {president_of, citizen_of}

Relation Mention: (*"Donald Trump", "USA"*, S2) Relation Types: {president_of, citizen_of}

Relation Mention: (*"Trump", "US"*, S3) Relation Types: {president_of, citizen_of}

Relation Mention: (*"Ellen", "United States"*, S4) Relation Types: {None}

Figure 4–1 Distant supervision generates training data by linking relation mentions in sentences S1-S4 to KB and assigning the linkable relation types to all relation mentions. Those unlinkable entity mention pairs are treated as negative examples. This automatic labeling process may cause errors of false positives (highlighted in red) and false negatives (highlighted in purple). QA pairs provide indirect supervision for correcting such errors.

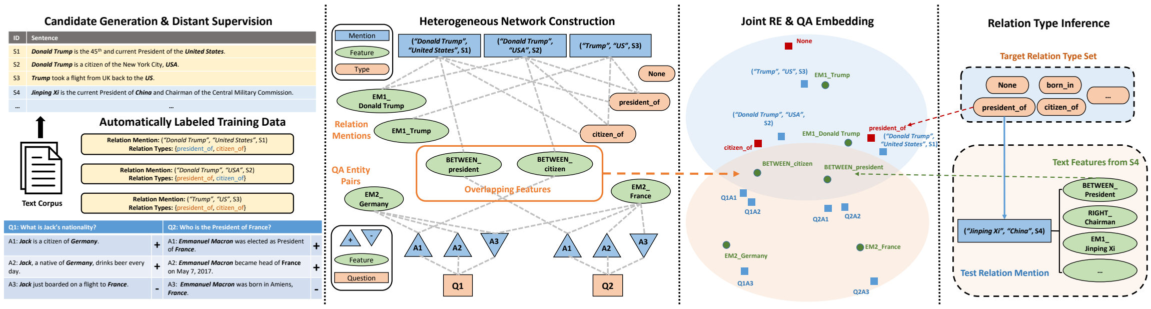


Figure 4–2 Overall Framework.

sion and distant supervision (DS). Weak supervision relies on a small set of manually-specified seed instances (or patterns) that are applied in bootstrapping learning to identify more instances of each type. This assumes seeds are unambiguous and sufficiently frequent in the corpus, which requires careful seed selection by human. The recent trend has deviated towards the adoption of distance supervision (DS). DS generates training data automatically by aligning texts and a knowledge base(KB). The typical workflow is : (1) detect entity mentions in text; (2) map detected entity mentions to entities in KB; (3) assign, to the candidate type set of each entity mention pair, all KB relation types between their KB-mapped entities. The automatically labeled training corpus is then used to infer types of the remaining candidate entity mentions and relation mentions (i.e., unlinkable candidate mentions).

Towards the goal of diminishing the negative effects by noisy DS training data, distantly supervised RE models that deal with training noise, as well as methods that directly improve the automatic training data generation process have been proposed. These methods mostly involve designing distinct assumptions to remove redundant training information[9–12]. For example, method applied in[10, 11] assumes that for each relation triple in the KB, at least one sentence might express the relation instead of all sentences. Moreover, these noise reduction systems usually only address one type of error, either false positives or false negatives. Hence, current methods handling DS noises still have the following challenges:

1. Lack of trustworthy sources: Current de-noising methods mainly focus on recognizing labeling mistakes from the labeled data itself, assisted by pre-defined assumptions or patterns. They do not have external trustworthy sources as guidance to uncover incorrectly labeled data, while not at the expense of excessive human efforts. Without other separate information sources, the reliability of false label identification can be limited.
2. Incomplete noise handling: Although both false negative and false positive errors are observed to be significant, most existing works only address one of them.

In this chapter, to overcome the above two issues derived from relation extraction with distant supervision, we study the problem of relation extraction with indirect supervision from external sources. Recently, the rapid emergence of QA systems promotes the availability of user feedback or datasets of various QA tasks. We investigate to leverage QA, a downstream application of relation extraction, to provide additional signals for learning RE models. Specifically, we use datasets for the task of answer sentence selection to facilitate relation typing. Given a domain-specific corpus and a set of target relation types from a KB, we aim to detect relation mentions from text and categorize each in context by target types or Non-Target-Type (None) by leveraging an independent dataset of QA pairs in the same domain. We address the above two challenges as follows: (1) We integrate indirect supervision from another same-domain data source in the format of QA sentence pairs, that is, each question sentence maps to several positive (where a true answer can be found) and negative (where no answer exists) answer sentences. We adopt the principle that for the same question, positive pairs of (question, answer) should be semantically similar while they should be dissimilar from negative pairs. (2) Instead of differentiating types of labeling errors at the instance level, we concentrate on how to better learn semantic representation of features. Wrongly labeled training examples essentially misguide the understanding of features. It increases the risk of having a non-representative feature learned to be close to a relation type and vice versa. Therefore, if the feature learning process is improved, potentially both types of error can be reduced. (See how QA pairs improve the feature embedding learning process in Figure 4–3).

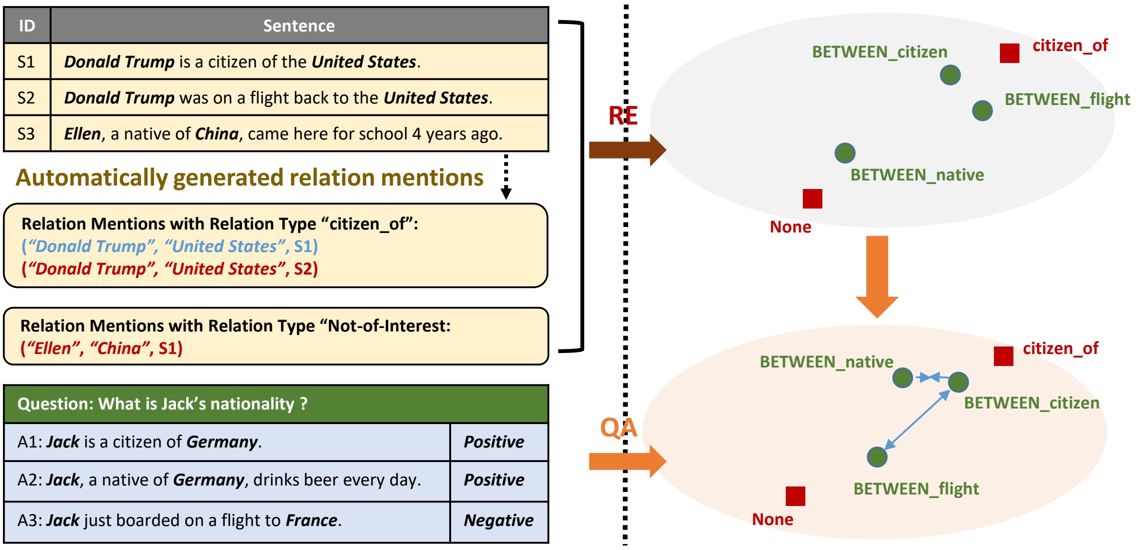
To integrate all the above elements, a novel framework, ReQuest, is proposed. First, ReQuest constructs a heterogeneous graph to represent three kinds of objects: relation mentions, text features and relation types for RE training data labeled by KB linking. Then, ReQuest constructs a second heterogeneous graph to represent entity mention pairs (include question, answer entity mention pairs) and features for QA dataset. These two graphs are combined into a single graph by overlapped features. We formulate a global objective to jointly embed the graph into a low-dimensional space where, in that space, RE objects whose types are semantically close also have similar representations and QA objects linked by positive (question, answer) entity mention pairs of a same question should have close representations. In particular, we design a novel margin-based loss to model the semantic similarity between QA pairs and transmit such information into feature and relation type representations via shared features. With the learned embeddings, we can efficiently estimate the types for test relation mentions. In summary, this chapter makes the following contributions:

1. We propose the novel idea of applying indirect supervision from question answering datasets to help eliminate noise from distant supervision for the task of relation extraction.
2. We design a novel joint optimization framework, ReQuest, to extract typed relations in domain-specific corpora.
3. Experiments with two public RE datasets combined with TREC QA demonstrate that ReQuest improves the performance of state-of-the-art RE systems significantly.

4.2 Definitions and Problem

Our proposed ReQuest framework takes the following input: an automatically labeled training corpus $\mathcal{D}_L$ obtained by linking a text corpus $\mathcal{D}$ to a KB (e.g. Freebase) Ψ, a target relation type set $\mathcal{R}$ and a set of QA sentence pairs $\mathcal{D}_{QAS}$ with extract answers labeled.

Entity and Relation Mention. An *entity mention* (denoted by m) is a token span in text which represents an entity e. A *relation instance* $r(e_1, e_2, \ldots, e_n)$ denotes some type of relation $r \in \mathcal{R}$ between multiple entities. In this chapter, we focus on binary relations, *i.e.*, $r(e_1, e_2)$. We define a *relation mention* (denoted by z) for some relation instance $r(e_1, e_2)$ as a (ordered) pair



ID	Sentence
S1	***Donald Trump*** is a citizen of the ***United States***.
S2	***Donald Trump*** was on a flight back to the ***United States***.
S3	***Ellen***, a native of ***China***, came here for school 4 years ago.

Automatically generated relation mentions

Relation Mentions with Relation Type "citizen_of":
("Donald Trump", "United States", S1)
("Donald Trump", "United States", S2)

Relation Mentions with Relation Type "Not-of-Interest:
("Ellen", "China", S1)

Question: What is Jack's nationality ?	
A1: ***Jack*** is a citizen of ***Germany***.	***Positive***
A2: ***Jack***, a native of ***Germany***, drinks beer every day.	***Positive***
A3: ***Jack*** just boarded on a flight to ***France***.	***Negative***

Figure 4–3 Due to the noise in the automatically generated RE training corpus, the associations between learned feature embeddings and relation types can be affected by the wrongly labeled training examples. However, the idea of QA pairwise interactions has the potential to correct such embedding deviations by bringing extra semantic clues from overlapped features in QA corpus.

of entities mentions of e_1 and e_2 in a sentence s, and represent a relation mention with entity mentions m_1 and m_2 in sentence s as $z = (m_1, m_2, s)$.

Knowledge Bases and Target Types. A KB contains a set of entities $\mathcal{E}_\Psi$, entity types $\mathbf{Y}$ and relation types $\mathcal{R}$, as well as human-curated facts on both relation instances $\mathbf{I}_\Psi = \{r(e_1, e_2)\} \subset \mathcal{R}_\Psi \times \mathcal{E}_\Psi \times \mathcal{E}_\Psi$, and entity-type facts $\mathcal{T}_\Psi = \{(e, y)\} \subset \mathcal{E}_\Psi \times \mathbf{Y}_\Psi$. *Target relation type set* $\mathcal{R}$ covers a subset of relation types that the users are interested in from Ψ, *i.e.*, $\mathcal{R} \subset \mathcal{R}_\Psi$.

Automatically Labeled Training Corpora. Distant supervision maps the set of entity mentions extracted from the text corpus to KB entities $\mathcal{E}_\Psi$ with an entity disambiguation system[70, 71]. Between any two linkable entity mentions m_1 and m_2 in a sentence, a relation mention z_i is formed if there exists one or more KB relations between their KB-mapped entities e_1 and e_2. Relations between e_1 and e_2 in KB are then associated to z_i to form its candidate relation type set $\mathcal{R}_i$, *i.e.*, $\mathcal{R}_i = \{r \mid r(e_1, e_2) \in \mathcal{R}_\Psi\}$.

Let $\mathbf{Z} = \{z_i\}_{i=1}^{Nz}$ denote the set of extracted relation mentions that can be mapped to KB. Formally, we represent the automatically labeled training corpus $\mathcal{D}_L$ for relation extraction, using a set of tuples $\mathcal{D}_L = \{(z_i, \mathcal{R}_i)\}_{i=1}^{Nz}$. There exists publicly available automatically labeled corpora such as the NYT dataset[10] where relation mentions have already been extracted and mapped to KB.

QA Entity Mention Pairs. The set of QA sentence pairs $\mathcal{D}_{QAS}$ consists of questions Q in the same domain as the training text corpus. For each question q_i, there will be a number of positive sentences $\mathbf{A}_i^+$, each of which contains a correct answer to the question and another set of negative sentences $\mathbf{A}_i^-$ where no answer can be found. And the tokens spans of the exact answer in each positive is marked as well. For each question, we extract positive QA (ordered) entity mention pairs $\mathbf{P}_i^+$ from $\mathbf{A}_i^+$ and negative entity mention pairs $\mathbf{P}_i^-$ from $\mathbf{A}_i^-$. A positive QA entity mention pair p_k contains an entity mention being asked about (question entity mention m_1) and an entity mention serving as the answer (answer entity mention m_2) to a question. That being said, we can get one positive QA entity mention pair from each positive answer sentence if both entity mentions can be found. In contrast, A negative QA entity mention pair does not follow such pattern for the corresponding question.

Let $Q = \{q_i\}_{i=1}^{N_q}$ denote the set of questions; $\mathbf{P} = \{p_k\}_{k=1}^{N_p}$ denote all QA entity mention pairs; $\mathbf{P}_i^+ = \{p_{k^+}\}_{k^+=1}^{N_i^+}$ denote the set of positive QA entity mention pairs for q_i; $\mathbf{P}_i^- = \{p_{k^-}\}_{k^-=1}^{N_i^-}$ denote the set of negative QA entity mention pairs for q_i. Formally, the QA entity mention pairs corpus is represented as $\mathcal{D}_{QA} = \{(q_i, \mathbf{P}_i^+, \mathbf{P}_i^-)\}_{i=1}^{N_q}$.

Definition 1 (Problem Definition)**.**
Given *an automatically generated training corpus $\mathcal{D}_L$, a target relation type set $\mathcal{R} \subset \mathcal{R}_\Psi$ and a set of QA sentence pairs $\mathcal{D}_{QAS}$ in the same domain, the relation extraction task* ***aims to*** *(1) extract QA entity mention pairs to generate $\mathcal{D}_{QA}$; (2) estimate a relation type $r^* \in \mathcal{R} \cup \{\mathsf{None}\}$ for each test relation mention, using both the training corpus and the extracted QA pairs with their contexts.*

4.3 Indirect Supervised Approach with Question Answering

Framework Overview. We propose an *embedding-based* framework with indirect supervision (illustrated in Figure 4–2) as follows:

1. Generate text features for each relation mention or QA entity mention pair, and construct a heterogeneous graph using four kinds of objects in combined corpus, namely relation mentions from RE corpus, entity mention pairs from QA corpus, target relation types and text features to encode aforementioned signals in a unified form (Section 4.3.1).
2. Jointly embed relation mentions, QA pairs, text features, and type labels into two low-dimensional spaces connected by shared features, where close objects tend to share the same types or questions (Section 4.3.2).

3. Estimate type labels r^* for each test relation mention z from learned embeddings, by searching the target type set $\mathcal{R}$ (Section 4.3.3).

4.3.1 Heterogeneous Network Construction

Relation Mentions and Types Generation. We get the relation mentions along with their heuristically obtained relation types from the automatically labeled training corpus $\mathcal{D}_L$. And we randomly sample a set of unlinkable entity mention pairs as the negative relation mentions (*i.e.*, relation mentions assigned with type "None").

QA Entity Mention Pairs Generation. We apply Stanford NER[72] to extract entity mentions in each question or answer sentence. First, we detect the target entity being asked about in each question sentence. For example, in the question "*Who is the president of United States*", the question entity is "*United States*". In most cases, a question only contains one entity mention and for those containing multiple entity mentions, we notice the question entity is mostly mentioned at the very last. Thus, we follow this heuristic rule to assign the lastly occurred entity mention to be the question entity mention m_0 in each question sentence q_i. Then, in each positive answer sentence of q_i, we extract the entity mention with matched head token and smallest edit string distance to be the question entity mention m_1, and the entity mention matching the exact answer string to be the answer entity mention m_2. Then we form a positive QA entity mention pair with its context s, $p_k = (m_1, m_2, s) \in \mathbf{P}_i^+$ for q_i. If either m_1 or m_2 can not be found, this positive answer sentence is dropped. We randomly select pairs of entity mentions in each negative answer sentence to be negative QA entity mention pairs for q_i (*e.g.*, if a negative sentence includes 3 entity mentions, we randomly select negative examples from the $3 \cdot 2 \cdot 1 = 6$ different pairs of entity mentions in total, if we ignore the order), with each negative example marked as $p_k\prime = (m_1\prime, m_2\prime, s\prime) \in \mathbf{P}_i^-$ for q_i.

Text Feature Extraction. We extract lexical features of various types from not only the mention itself (*e.g.*, head token), as well as the context s (*e.g.*, bigram) in a POS-tagged corpus. It is to capture the syntactic and semantic information for any given relation mentions or entity mention pairs. See Table 4–1 for all types of text features used, following those in[9, 73] (excluding the dependency parse-based features and entity type features).

We denote the set of M_z unique features extracted from relation mentions $\mathbf{Z}$ as $\mathcal{F}_z = \{f_j\}_{j=1}^{M_z}$ and the set of M_{QA} unique features extracted of QA entity mention pairs $\mathbf{P}$ as $\mathcal{F}_{QA} = \{f_j\}_{j=1}^{M_{QA}}$. As our embedding learning process will combine these two sets of features and their shared ones will act as the bridge of two embedding spaces, we denote the overall feature set as $\mathcal{F} = \{f_j\}_{j=1}^{M}$.

Table 4–1 Text features for relation mentions used in this work[10, 74] (excluding dependency parse-based features and entity type features, EM = Entity Mention). ("*Donald Trump*", "*United States*") is used as an example relation mention from the sentence "*NYC native* ***Donald Trump*** *is the current President of the* ***United States***."

Feature	**Description**	**Example**
EM head	Syntactic head token of each entity mention	"*HEAD_EM1_Trump*"
EM Token	Tokens in each entity mention	"*TKN_EM1_Donald*"
Tokens	Each token between two EMs	"*is*", "*current*", "*President*", "*of*"
POS tag	POS tags of tokens between two EMs	"*VBZ*", "*DT*", "*JJ*", "*NN*", "*IN*", "*DT*"
Collocations	Bigrams in 3-word window of each EM	"*NYC native*", "*native Donald*", ...
EM order	Whether EM 1 is before EM 2	"*EM1_BEFORE_EM2*"
EM distance	Number of tokens between the two EMs	"*EM_DISTANCE_6*"
EM context	Unigrams before and after each EM	"*native*", "*is*", "*the*", "."
Special pattern	Occurrence of pattern "em1_in_em2"	"*PATTERN_NULL*"
Brown cluster	Brown cluster ID for each token	"*8_1101111*", "*12_111011111111*"

Heterogeneous Network Construction. After the nodes generation process, we construct a heterogeneous network connected by text features, relation mentions, relation types, questions, QA entity mention pairs, as shown in the second column of Figure 4–2.

4.3.2 Joint RE and QA Embedding

This section first introduces how we model different types of interactions between linkable relation mentions $\mathbf{Z}$, QA entity mention pairs $\mathbf{P}$, relation type labels $\mathcal{R}$ and text features $\mathcal{F}$ into a d-dimensional *relation vector space* and a d-dimensional *QA pair vector space*. In the relation vector space, objects whose types are close to each other should have similar representation and in the QA pair vector space, positive QA mention pairs who share the same question are close to each other. (*e.g.*, see the 3rd col. in Figure 4–2). We then combine multiple objectives and formulate a joint optimization problem.

We propose a novel global objective, which employs a margin-based rank loss[75] to model *noisy mention-type associations* and utilizes the second-order proximity idea[76] to model *mention-feature (QA pair-feature) co-occurrences*. In particular, we adopt a pairwise margin loss, following the intuition of pairwise rank[77] to capture the *interactions between QA pairs*, and the shared features $\mathcal{F}_z \cap \mathcal{F}_{QA}$ between relation mentions $\mathbf{Z}$ and QA pairs $\mathbf{P}$ connect the two vector spaces.

Modeling Types of Relation Mentions. We introduce the concepts of both *mention-feature co-occurrences* and *mention-type associations* in the modeling of relation types for relation mentions in set Z.

The first hypothesis involved in modeling types of relation mentions is as follows.

Hypothesis 1 (Mention-Feature Co-occurrence)**.**
If two relation mentions share many text features, they tend to share similar types (close to each other in the embedding space). If two features co-occur with a similar set of relation mentions, they tend to have similar embedding vectors.

This is based on the intuition that if two relation mentions share many text features, they have high distributional similarity over the set of text features $\mathcal{F}_z$ and likely they have similar relation types. On the other hand, if text features co-occur with many relation mentions in the corpus, such features tend to represent close type semantics. For example, in sentences s_1 and s_4 in the first column of Figure 4–2, the two relation mentions ("*Donald Trump*", "*United States*", s_1) and ("*Jinping Xi*", "*China*", s_4) share many text features including "*BETWEEN_President*" and they indeed have the same relation type "`president_of`"

Formally, let vectors $\mathbf{z}_i$, $\mathbf{c}_j \in \mathbb{R}^d$ represent relation mention $z_i \in \mathbf{Z}$ and text feature $f_j \in \mathcal{F}_z$ in the d-dimensional *relation embedding space*. Similar to the distributional hypothesis[21] in text corpora, we apply second-order proximity[76] to model the idea in Hypothesis 1 as follows.

$$\mathbf{L}_{ZF} = -\sum_{z_i \in \mathbf{Z}} \sum_{f_j \in \mathcal{F}_z} w_{ij} \cdot \log p(f_j | z_i), \tag{4–1}$$

where $p(f_j|z_i) = \mathbf{e}(\mathbf{z}_i^T \mathbf{c}_j) / \sum_{f' \in \mathcal{F}_z} \mathbf{e}(\mathbf{z}_i^T \mathbf{c}_{j'})$ denotes the probability of f_j generated by z_i, and w_{ij} is the co-occurrence frequency between (z_i, f_j) in corpus $\mathcal{D}$.

For the goal of efficient optimization, we apply negative sampling strategy[21] to sample multiple *false* features for each (z_i, f_j) based on some *noise distribution* $P_n(f) \propto D_f^{3/4}$[21] (with D_f denotes the number of relation mentions co-occurring with f). Term $\log p(f_j|z_i)$ in Eq. (4–1) is replaced with the term as follows.

$$\log \sigma(\mathbf{z}_i^T \mathbf{c}_j) + \sum_{v=1}^{V} \mathbb{E}_{f_{j'} \sim P_n(f)} \left[\log \sigma(-\mathbf{z}_i^T \mathbf{c}_{j'})\right], \tag{4–2}$$

where $\sigma(x) = 1/\big(1 + \exp(-x)\big)$ is the sigmoid function. The first term in Eq. (4–2) models the observed co-occurrence, and the second term models the V negative feature samples.

In D_L, each relation mention z_i is associated with a set of candidate types $\mathcal{R}_i$ in a context-agnostic setting, which leads to some false associations between z_i and $r \in \mathcal{R}_i$ (*i.e.*, false positives). For example, in the first column of Figure 4–2, the two relation mentions ("*Donald Trump*", "*United States*", s_1) and ("*Donald Trump*", "*USA*", s_2) are assigned to the same relation types while each mention actually only has one true type. To handle such conflicts, we use the following hypothesis to model the associations between each linkable relation mention z_i (in set $\mathbf{Z}$) and its noisy candidate relation type set $\mathcal{R}_i$.

Hypothesis 2 (Partial-Label Association)**.**
A relation mention's embedding vector should be more similar (closer in the low-dimensional space) to its "most relevant" candidate type, than to any other non-candidate type.

Let vector $\mathbf{r}_k \in \mathbb{R}^d$ denote relation type $r_k \in \mathcal{R}$ in the embedding space, the similarity between (z_i, r_k) is defined as the dot product of their embedding vectors, *i.e.*, $\phi(z_i, r_k) = \mathbf{z}_i^T \mathbf{r}_k$. $\overline{\mathcal{R}}_i = \mathcal{R} \setminus \mathcal{R}_i$ denotes the set of *non-candidate types*. We extend the margin-based loss in[75] to define a partial-label loss ℓ_i for each linkable relation mention $z_i \in \mathcal{M}_L$.

To comprehensively model the types of relation mentions, we integrate the modeling of mention-feature co-occurrences and mention-type associations by the following objective, so that feature embeddings also participate in modeling the relation type embeddings.

$$O_Z = \mathbf{L}_{ZF} + \sum_{i=1}^{N_Z} \ell_i + \frac{\lambda}{2} \sum_{i=1}^{N_Z} \|\mathbf{z}_i\|_2^2 + \frac{\lambda}{2} \sum_{k=1}^{K_r} \|\mathbf{r}_k\|_2^2, \tag{4–3}$$

where tuning parameter $\lambda > 0$ on the regularization terms is used to control the scale of the embedding vectors.

Modeling Associations between QA Entity Mention Pairs. We follow Hypothesis 1 to model the QA pair-feature co-occurrence in a similar way. Formally, let vectors $\mathbf{p}_i, \mathbf{c}'_j \in \mathbb{R}^d$ represent QA entity mention pair $p_i \in \mathbf{P}$ and text features (for entity mentions) $f_j \in \mathcal{F}_{QA}$in a d-dimensional *QA entity pair embedding space*, respectively. We model the corpus-level co-occurrences between QA entity mention pairs and text features by second-order proximity as follows.

$$\mathbf{L}_{PF} = -\sum_{p_i \in \mathbf{P}} \sum_{f_j \in \mathcal{F}_{QA}} w_{ij} \cdot \log p(f_j | p_i), \tag{4–4}$$

where the term $\log p(f_j|p_i)$ is defined as $\log p(f_j|p_i) = \log\sigma(\mathbf{p}_i^T\mathbf{c}'_j)+\sum_{v=1}^{V}\mathbb{E}_{f_{j'}\sim P_n(f)}\left[\log\sigma(-\mathbf{p}_i^T\mathbf{c}'_{j'})\right]$.

For each QA entity mention pair, if we consider it as a relation mention with an unknown type, intuitively, positive pairs sharing a same question are relation mentions with the same relation type or more specifically, are semantically similar relation mentions. In contrast, a positive pair and a negative pair for a question should be semantically far away from each other. For example, in Figure 4–3, the embeddings of the entity mention pair in answer sentence A_1 should be close to the pair in A_2 while far away from the pair in A_3. To impose such idea, we model the interactions between QA entity mention pairs based on the following hypothesis.

Hypothesis 3 (QA Pairwise Interaction)**.**
A positive QA entity mention pair's embedding vector should be more similar (closer in the low-dimensional space) to any other positive QA entity mention pair, than to any negative QA entity mention pair of the same question.

Specifically, we use vector $\mathbf{p}_k \in \mathbb{R}^d$ to represent a positive QA entity mention pair p_k in the embedding space. The similarity between two QA entity mention pairs p_{k1} and p_{k2} is defined as the dot product of their embedding vectors. For a positive QA entity mention pair p_k of a question q_i (e.g. $p_k \in \mathbf{P}_i^+$), we define the pairwise margin-based loss as follows.

$$\ell_{i,k} = \sum_{p_{k_1}\in\mathbf{P}_i^+, p_{k_2}\in\mathbf{P}_i^-, k_1\neq k} \max\left\{0, 1-\left[\phi(p_k,p_{k_1})-\phi(p_k,p_{k_2})\right]\right\}. \quad (4\text{–}5)$$

To integrate both the modeling of QA pair-feature co-occurrence and QA pairs interaction, we formulate the following objective.

$$O_{QA} = \mathbf{L}_{PF} + \sum_{i=1}^{N_Q}\sum_{k=1}^{N_i^+}\ell_{i,k} + \frac{\lambda}{2}\sum_{k=1}^{N_P}\|\mathbf{p}_k\|_2^2. \quad (4\text{–}6)$$

By doing so, we can extend the semantic relationships between QA pairs to feature embeddings, such that features of close QA pairs also have similar embeddings. Thus, the learned embeddings of text features from QA corpus carry semantic information inferred from QA pairs. The shared features can propagate such extra semantic knowledge into relation vector space and help better learn the semantic embeddings of both text features and relation types. While feature embeddings of both false positive or false negative examples in the training corpus can deviate towards unrepresentative relation types, the transmitted knowledge from QA space has the potential to adjust such semantic inconsistency. For example, as illustrated in Figure 4–3, the false labeled examples in s_2 and s_3 lead the features "*BETWEEN_flight*" and "*BETWEEN_native*" to

Algorithm 4–1 Model Learning of ReQuest

```
Input: labeled training corpus $\mathcal{D}_L$, text features $\{\mathcal{F}\}$, regularization parameter $\lambda$, learning rate $\alpha$, number of negative samples $V$, dim. $d$
Output: relation mention/QA entity mention pair embeddings $\{\mathbf{z}_i\}$/$\{\mathbf{p}_k\}$, feature embeddings $\{\mathbf{c}_j\}$, $\{\mathbf{c}'_j\}$, relation type embedding $\{\mathbf{r}_k\}$
  Initialize: vectors $\{\mathbf{z}_i\}$,$\{\mathbf{p}_k\}$,$\{\mathbf{c}_j\}$,$\{\mathbf{c}'_j\}$,$\{\mathbf{r}_k\}$ as random vectors
  while $O$ in Eq. (4–7) not converge do
      Sample one component $O_{cur}$ from $\{O_Z,\ O_{QA}\}$
      if $O_{cur}$ is $O_Z$ then
          Sample a mention-feature co-occurrence $w_{ij}$; draw $V$ negative samples; update $\{\mathbf{z}, \mathbf{c}\}$ based on $\mathbf{L}_{ZF}$
          Sample a relation mention $z_i$; get its candidate types $\mathcal{R}_i$; update $\mathbf{z}$ and $\{\mathbf{r}\}$ based on $O_Z - \mathbf{L}_{ZF}$
      end if
      if $O_{cur}$ is $O_{QA}$ then
          Sample a pair-feature co-occurrence $w_{ij}$; draw $V$ negative samples; update $\{\mathbf{p}, \mathbf{c}'\}$ based on $\mathbf{L}_{PF}$
          Sample an positive QA entity mention pair $p_k$ of question $q_i$; sample one more positive pair and one negative pair of question $q_i$; update $\mathbf{p}$ based on $O_{QA} - \mathbf{L}_{PF}$
      end if
  end while
```

be close to “`citizen_of`” and “`None`” type respectively. After injecting the QA pairwise interactions from the example question, these wrongly placed features are brought back towards the relation types they actually indicate. Minimizing the objective O_{QA} yields an QA pair embedding space where, in that space, positive QA mention pairs who share the same question are close to each other.

A Joint Optimization Problem. Our goal is to embed all the available information for relation mentions and relation types, QA entity mention pairs and text features into a single d-dimensional embedding space. An intuitive solution is to collectively minimize the two objectives O_Z and O_{QA} as the embedding vectors of overlapped text features are shared across relation vector space and QA pair vector space. To achieve the goal, we formulate a joint optimization problem as follows.

$$\min_{\{\mathbf{z}_i\},\{\mathbf{c}_j\},\{\mathbf{r}_k\},\{\mathbf{p}_k\},\{\mathbf{c}'_j\}} O = O_Z + O_{QA}. \tag{4–7}$$

When optimizing the global objective O, the learning of RE and QA embeddings can be mutually influenced as errors in each component can be constrained and corrected by the other. This mutual enhancement also helps better learn the semantic relations between features and relation types. We apply edge sampling strategy[76] with a stochastic sub-gradient descent algorithm[78] to efficiently solve Eq. (4–7). In each iteration, we alternatively sample from each of the two objectives $\{O_Z, O_M\}$ a batch of edges (*e.g.*, (z_i, f_j)) and their negative samples, and update each embedding vector based on the derivatives. The detailed learning process of ReQuest can be seen in Algorithm 4–1. To prove convergence of this algorithm (to the local minimum), we can adopt the proof procedure in[78].

4.3.3 Type Inference

To predict the type for each test relation mention z, we search for nearest neighbor in the target relation type set $\mathcal{R}$, with the learned embeddings of features and relation types (*i.e.*, $\{\mathbf{c}_i\}$, $\{\mathbf{c}'_i\}$, $\{\mathbf{r}_k\}$). Specifically, we represent test relation mention z in our learned relation embedding space by $\mathbf{z} = \sum_{f_j \in \mathcal{F}_z(z)} \mathbf{c}_j$ where $\mathcal{F}_z(z)$ is the set of text features extracted from z's local context s. We categorize z to None type if the similarity score is below a pre-defined threshold (e.g. $\eta > 0$).

4.4 Experiments

4.4.1 Data Preparation and Experiment Setting

Our experiments consists of two different type of datasets, one for relation extraction and another answer sentence selection dataset for indirect supervision. Two public datasets are used for relation extraction: **NYT**[10, 11]and **KBP**[79, 80]. The test data are manually annotated with relation types by their respective authors. Statistics of the datasets are shown in Table 4–2. Automatically generated training data by distant supervision on these two training corpora have been used in[10, 81] and is accessible via public links, as well as the test data[1]. The automatic data generation process is the same as described in Section 4.2 by utilizing DBpedia Spotlight[2], a state-of-the-art entity disambiguation tool, and Freebase, a large entity knowledge base. As for

[1] https://github.com/shanzhenren/CoType/tree/master/data/source

[2] http://spotlight.dbpedia.org/

Data sets	NYT	KBP
#Relation types	24	19
#Documents	294,977	780,549
#Sentences	1.18M	1.51M
#Training RMs	353k	148k
#Text features	2.6M	1.3M
#Test Sentences	395	289
#Ground-truth RMs	3,880	2,209

Table 4–2 Statistics of relation extraction datasets.

QA dataset, we use the answer sentence selection dataset extracted from **TREC-QA** dataset[82] used by many researchers[83–85]. We obtain the compiled version of the dataset from[86, 87], which can be accessed via publicly available link[1]. Then, we parse this QA dataset to generate QA entity mention pairs following the steps described in Section 4.3.1. During this procedure, we drop the question or answer sentences where no valid QA entity mention pairs can be found. The statistics of this dataset is presented in Table 4–3.

Feature Generation. This step is run on both relation extraction dataset and preprocessed QA entity mention pairs and sentences. Table 4–1 lists the set of text features of both relation mentions and QA entity mention pairs used in our experiments. We use a 6-word window to extract context features for each mention (3 words on the left and the right). We apply the Stanford CoreNLP tool[72] to get POS tags. Brown clusters are derived for each corpus using public implementation[2]. The same kinds of features are used in all the compared methods in our experiments. As the overlapped features in both RE and QA datasets play an important role in the optimization process, we put the statistics of the shared features in Table 4–4.

Evaluation Sets. The provided train/test split are used in NYT and KBP relation extraction datasets. The relation mentions in test data have been manually annotated with relation types in the released dataset (see Table 4–2 for the data statistics). A *validation set* is created through randomly sampling 10% of relation mentions from test data, and the rest are used as *evaluation set*.

Compared Methods. We compare ReQuest with its variants which model parts of the proposed hypotheses. Several state-of-the-art relation extraction methods (*e.g.*, supervised,

[1] https://github.com/xuchen/jacana/tree/master/tree-edit-data

[2] https://github.com/percyliang/brown-cluster

Versions of QA dataset	**COMPLETE**	**FILTERED**
#Questions	1.4K	186
#Positive Answer Sentences	6.9K	969
#Negative Answer Sentences	49K	5.5K
#Positive entity mention pairs	-	969
#Negative entity mention pairs	-	28K

Table 4–3 Statistics of the answer sentence selection datasets. The complete version is the raw corpus we obtain from the public link. The filtered version is the input to ReQuest after dropping sentences where no valid QA entity mention pair can be found.

Data sets	**NYT**	**KBP**
% distinct shared features with TREC QA	10.0%	11.6%
% occurrences of shared features with TREC QA	90.1%	85.6%

Table 4–4 Statistics of overlapped features. For example, if we have the following observations in NYT and TREC QA respectively: $(f_1, f_1, f_1, f_2, f_3)$ and (f_1, f_2, f_4), then % distinct shared features with TREC QA of NYT is 66.7% (f_1, f_2) and % occurrences of shared features with TREC QA of NYT is 80.0%.

embedding, neural network) are also implemented (or tested using their published codes): (1) **DS+Perceptron**[79]: adopts multi-label learning on automatically labeled training data $\mathcal{D}_L$. (2) **DS+Kernel**[88]: applies bag-of-feature kernel[88] to train a SVM classifier using $\mathcal{D}_L$; (3) **DS+Logistic**[9]: trains a multi-class logistic classifier[1] on $\mathcal{D}_L$; (4) **DeepWalk**[89]: embeds mention-feature co-occurrences and mention-type associations as a homogeneous network (with binary edges); (5) **LINE**[76]: uses second-order proximity model with edge sampling on a feature-type bipartite graph (where edge weight w_{jk} is the number of relation mentions having feature f_j and type r_k); (6) **MultiR**[11]: is a state-of-the-art distant supervision method, which models noisy label in $\mathcal{D}_L$ by multi-instance multi-label learning; (7) **FCM**[90]: adopts neural language model to perform compositional embedding; (8) **DS+SDP-LSTM**[59, 91]: current state-of-the-art in SemEval 2010 Task 8 relation classification task[92], leverages a multi-channel input along the shortest dependency path between two entities into stacked deep recurrent neural network model. We use $\mathcal{D}_L$ to train the model. (9) **DS+LSTM-ER**[93]: current state-of-the-art model on ACE2005 and ACE2004 relation classification task[94, 95]. It is a multi-layer

[1] We use liblinear package from `https://github.com/cjlin1/liblinear`

Relation Mention	**ReQuest**	**CoType-RM**
.. traveling to *Amman* **,** *Jordan* ..	/location/location/contains	None
The photograph showed **Gov.** *Ernie Fletcher* **of** *Kentucky* ..	/people/person/place_lived	None
.. **as chairman of** the *Securities and Exchange Commission* , *Christopher Cox* ..	/business/person/company	None

Table 4–5 Case Study.

LSTM-RNN based model that captures both word sequence and dependency tree substructure information. We use $\mathcal{D}_L$ to train the model. (10) **CoType-RM**[81]: A distant supervised model which adopts the partial-label loss to handle label noise and train the relation extractor.

Besides the proposed joint optimization model, **ReQuest-Joint**, we conduct experiments on two other variations to compare the performance (1) **ReQuest-QA_RE**: This variation optimizes objective O_{QA} first and then uses the learned feature embeddings as the initial state to optimize O_Z; and (2) **ReQuest-RE_QA**: It first optimizes O_Z, then optimizes O_{QA} to finely tune the learned feature embeddings.

Parameter Settings. In the testing of ReQuest and its variants, we set $\eta = 0.35$ and $\lambda = 10^{-4}$ and $V = 3$ based on validation sets. We stop further optimization if the relative change of O in Eq. (4–7) is smaller than 10^{-4}. The dimensionality of embeddings d is set to 50 for all embedding methods. For other parameters, we tune them on validation sets and picked the values which lead to the best performance.

Evaluation Metrics. We adopt standard Precision, Recall and F1 score[88, 96] for measuring the performance of relation extraction task. Note that all our evaluations are *sentence-level* or *mention-level* (*i.e.*, context-dependent), as discussed in[11].

4.4.2 Experiments and Performance Study

Performance Comparison with Baselines. To test the effectiveness of our proposed framework ReQuest, we compare with other methods on the relation extraction task. The precision, recall, F1 scores as well as the model learning time measured on two datasets are reported in Table 4–6. As shown in the table, ReQuest achieves superior F1 score on both datasets compared with other models. Among all these baselines, MultiR and CoType-RM handle noisy training data

	NYT[10, 11]				**KBP**[79, 80]			
Method	**Prec**	**Rec**	**F1**	**Time**	**Prec**	**Rec**	**F1**	**Time**
DS+Perceptron[79]	0.068	**0.641**	0.123	15min	0.233	0.457	0.308	7.7min
DS+Kernel[88]	0.095	0.490	0.158	56hr	0.108	0.239	0.149	9.8hr
DS+Logistic[9]	0.258	0.393	0.311	25min	0.296	0.387	0.335	14min
DeepWalk[89]	0.176	0.224	0.197	1.1hr	0.101	0.296	0.150	27min
LINE[76]	0.335	0.329	0.332	2.3min	0.360	0.257	0.299	1.5min
MultiR[11]	0.338	0.327	0.333	5.8min	0.325	0.278	0.301	4.1min
FCM[90]	**0.553**	0.154	0.240	1.3hr	0.151	**0.500**	0.301	25min
DS+SDP-LSTM[91]	0.307	0.532	0.389	21hr	0.249	0.300	0.272	10hr
DS+LSTM-ER[93]	0.373	0.171	0.234	49hr	0.338	0.106	0.161	30hr
CoType-RM[81]	0.467	0.380	0.419	2.6min	0.342	0.339	0.340	1.5min
ReQuest-QA_RE	0.407	0.437	0.422	10.2min	**0.459**	0.300	0.363	5.3min
ReQuest-RE_QA	0.435	0.419	0.427	8.0min	0.356	0.352	0.354	13.2min
ReQuest-Joint	0.404	0.480	**0.439**	4.0min	0.386	0.410	**0.397**	5.9min

Table 4–6 Performance comparison on end-to-end relation extraction (at the highest F1 point) on the two datasets.

while the remaining ones assume the training corpus is perfectly labeled. Due to their nature of being cautious towards the noisy training data, both MultiR and CoType-RM reach relatively high results confronting with other models that blindly exploit all heuristically obtained training examples. However, as external reliable information sources are absent and only the noise from multi-label relation mentions (while none or only one assigned label is correct) is tackled in these models, MultiR and CoType-RM underperform ReQuest. Especially from the comparison with CoType-RM, which is also an embedding learning based relation extraction model with the idea of partial-label loss incorporated, we can conclude that the extra semantic inklings provided by the QA corpus do help boost the performance of relation extraction.

Performance Comparison with Ablations. We experiment with two variations of ReQuest, ReQuest-QA_RE and ReQuest-RE_QA, in order to validate the idea of joint optimization. As presented in Table 4–6, both ReQuest-QA_RE and ReQuest-RE_QA outperform most of the baselines, with the indirect supervision from QA corpus. However, their results still fall behind ReQuest's. Thus, separately training the two components may not capture as much information as jointly optimizing the combined objective. The idea of constraining each component in the joint optimization process proves to be effective in learning embeddings to present semantic meanings of objects (e.g. features, types and mentions).

4.4.3 Case Study

Example Outputs. We have done some interesting investigations regarding the type of prediction errors that can be corrected by the indirection supervision from QA corpus. We have analyzed the prediction results on NYT dataset from CoType-RM and ReQuest and find out the top three target relation types that can be corrected by ReQuest are "`contains_location`", "`work_for`", "`place_lived`". Both the issues of KB incompleteness and context-agnostic labeling are severe for these relation types. For example, there can be lots of not that well-known suburban areas belonging to a city, a state or a country while not marked in KB. And a person can has lived in tens or even hundreds places for various lengths of period. These are hard to be fully annotated into a KB. Thus, the automatically obtained training corpus may end up containing a large percentage of false negative examples for such relation types. On the other hand, there are abundant entity pairs having both "`contains_location`" and "`capital_of`", or both "`place_lived`" and "`born_in`" relation types in KB. Naturally, training examples of such entity pairs can be greatly polluted by false positives. In this case, it becomes tough to learn semantic embeddings for relevant features of these relation types. However, we notice there are quite a few answer sentences for relevant questions like "Where is *XXX* located", "Where did *XXX* live", "What company is *XXX* with" in the QA corpus, which plays an important role in adjusting vectors for features that are supposed to be the indicators for these relation types. Table 4–5 shows some prediction errors from CoType-RM that are fixed in ReQuest.

Study the effect of QA dataset processing on F1 scores. As stated in Section 4.3.1, ReQuest uses Stanford NER to extract entity mentions in QA dataset and all QA pairs consist of two entity mentions and if either question or answer entity mention is not found, it drops the sentence. Beyond that, we have conducted experiments with four other ways to construct QA pairs from the raw QA sentences. As shown in Table 4–3, we lose many positive QA pairs if we only remain answer (or question) targets that are detected as named entities. Thus, we have tried to keep more positive pairs by relaxing the restriction from named entities to noun phrases. In addition, we have tried to evaluate the performance by 1) keeping negative pairs as named entity pairs or 2) changing them to noun phrase pairs. Besides that, inspired by[59, 91], the third processing variation we have tried is to parse the QA sentences into dependency paths and to extract features from these paths instead of the full sentences. The last one is that, we sample negative QA

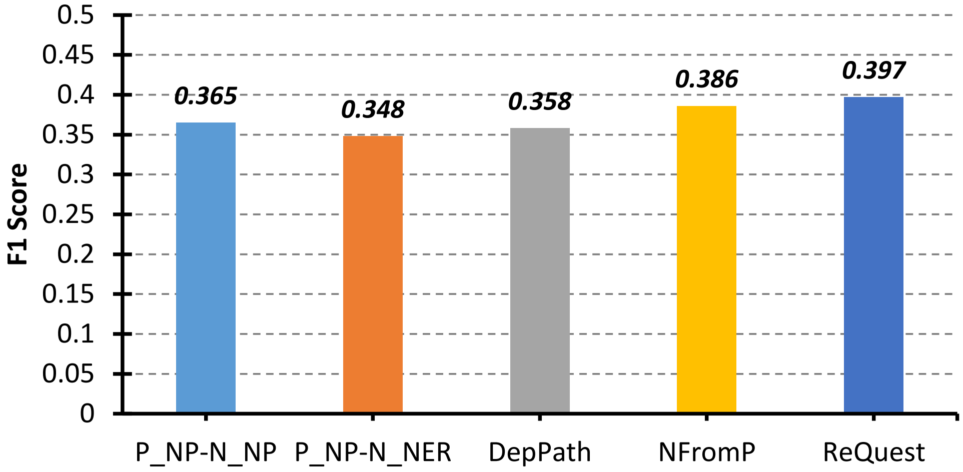


Figure 4–4 Effect of QA dataset processing on F1 scores. P_NP-N_NP: positive QA noun phrase pairs + negative QA noun phrase pairs, P_NP-N_NER: positive QA noun phrase pairs + negative QA named entity pairs, DepPath: convert QA sentences to dep paths, NFromP: sample negative QA pairs from both positive and negative answer sentences.

pairs not only from negative answer sentences, but also from positive sentence when extracting QA pairs. However, ReQuest achieves highest F1 score compared with these four processing variations (as shown in Figure 4–4) by filtering out all non entity mention answers, keeping full sentences and extracting only positive QA pairs from positive answer sentences.

Although by doing so, ReQuest filters out a large number of question/answer sentences and fewer QA pairs are constructed to provide semantic knowledge for RE, the remaining QA pairs provide cleaner and more consistent information with RE dataset. Thus, it still outperforms the other variations. Another interesting highlight is the comparison between using negative named entity pairs and using negative noun phrase pairs when positive QA pairs are formed by noun phrases. Although enforcing named entities is more consistent with RE datasets, a trade-off exists when the data format of positive and negative QA pairs are inconsistent. As we can see from the bar chart, the performance by using negative noun phrase pairs is better than negative named entity pairs.

4.5 Related Work

Classifying relation types between entities in a certain sentence and automatically extracting them from large corpora plays a key role in information extraction and natural language processing applications and thus has been a hot research topic recently. Even though many existing knowledge bases are very large, they are still far from complete. A lot of information is hidden in unstructured data, such as natural language text. Most tasks focus on knowledge base completion (KBP)[97] as a goal of relation extraction from corpora like New York Times (NYT)[10]. Others extract valuable relation information from community question-answer texts, which may be unique to other sources[98].

For supervised relation extraction, feature-based methods[92] and neural network techniques[99, 100] are most common. Most of them jointly leverage both semantic and syntactic features[93], while some use multi-channel input information as well as shortest dependency path to narrow down the attention[59, 91]. Two of the aforementioned papers perform the best on the SemEval-2010 Task 8 and constitutes our neural baseline methods.

However, most of these methods require large amount of annotated data, which is time consuming and labor intensive. To address this issue, most researchers align plain text with knowledge base by *distant supervision*[9] for relation extraction. However, distant supervision inevitably accompanies with the wrong labeling problem. To alleviate the wrong labeling problem, multi-instance and multi-label learning are used[10, 11]. Others[81, 95] propose joint extraction of typed entities and relations as joint optimization problem and posing cross-constraints of entities and relations on each other. Neural models with selective attention[12] are also proposed to automatically reduce labeling noise.

The distant supervision provides one solution to the cost of massive training data. However, traditional DS methods mostly only exploit one specific kind of indirect supervision knowledge - the relations/facts in a given knowledge base, thus often suffer from the problem of lack of supervision. There exist other *indirect supervision* methods for relation extraction, where some utilize globally and cross sentence boundary supervision[101, 102], some leverage the power of passage retrieval model for providing relevance feedback on sentences[103], and others[104–106]. Recently, with the prevalence of reinforcement learning applications, many information extraction and relation extraction tasks have adopted such techniques to boost existing approaches[107, 108]. Our methodology follows the success of indirect supervision, by adding question-answering pairs as another source of supervision for relation extraction task along with knowledge base auto-labeled distant supervision as well as partial supervision.

Another indirect supervision source we use in the chapter, passage retrieval, as described here, is the task of retrieving only the portions of a document that are relevant to a particular information need. It could be useful for limiting the amount of non-relevant material presented to a searcher, or for helping the searcher locate the relevant portions of documents more quickly. Passage retrieval is also often an intermediate step in other information retrieval tasks, like question answering[109–112] and combining with summarization. Some passage retrieval approaches[113] include calculating query-likelihood and relevance modeling[114], others show that language model approaches used for document retrieval can be applied to answer passage retrieval[115]. Following the success of passage retrieval usage in question-answering pipelines, to the best of our knowledge, we are the first to utilize passage retrieval, or specifically, answer sentence selection from question-answer pairs to provide additional indirect feedback and supervision for relation extraction task.

Chapter 5 Relation Extraction for Commonsense Knowledege

5.1 Introduction

Commonsense knowledge is an important ingredient in machine comprehension and inference. Artificial intelligence systems can benefit from incorporating commonsense knowledge as background, such as *ice is cold* (HasProperty), *chewing is a sub-event of eating* (HasSubevent), *chair and table are typically found near each other* (LocatedNear), etc. These kinds of commonsense facts have been used in many downstream tasks, such as textual entailment[5, 6] and visual recognition tasks[7]. The commonsense knowledge is often represented as relation triples in commonsense knowledge bases, such as *ConceptNet*[8], one of the largest commonsense knowledge graphs available today. However, most commonsense knowledge bases are manually curated or crowd-sourced by community efforts and thus do not scale well. For example, ConceptNet contains only 49 LocatedNear relation triples. Many commonly co-located objects such as (house, garden) and (fork, knife) are not included in this knowledge base. Another problem is that such commonsense knowledge bases are typically contributed by just a very limited number of people due to the cost of manual labor. Thus no meaningful statistical scores can be associated with the triples, making rank-based computation difficult. For instance, although ConceptNet gives a confidence score (from 0 to infinity) to each triple, most of the triples have the default score of 1, simply because the human contributor did not or could not provide a score. If such commonsense knowledge is harnessed automatically from open-domain text corpora, both of the above problems can be effectively addressed. Open information extraction not only provides the much needed scale, but also valuable statistics that can turn into confidence scores.

This chapter aims to automatically extract the commonsense LocatedNear relation between physical objects from textual corpora. LocatedNear is defined as the relationship between two objects typically found near each other in real life. Because some physical objects can be a location itself, this relation may include some instances of the atLocation relation, e.g., *room* and *door*.

We focus on LocatedNear relation for these reasons:

1. LocatedNear facts provide helpful prior knowledge to object detection tasks in com-

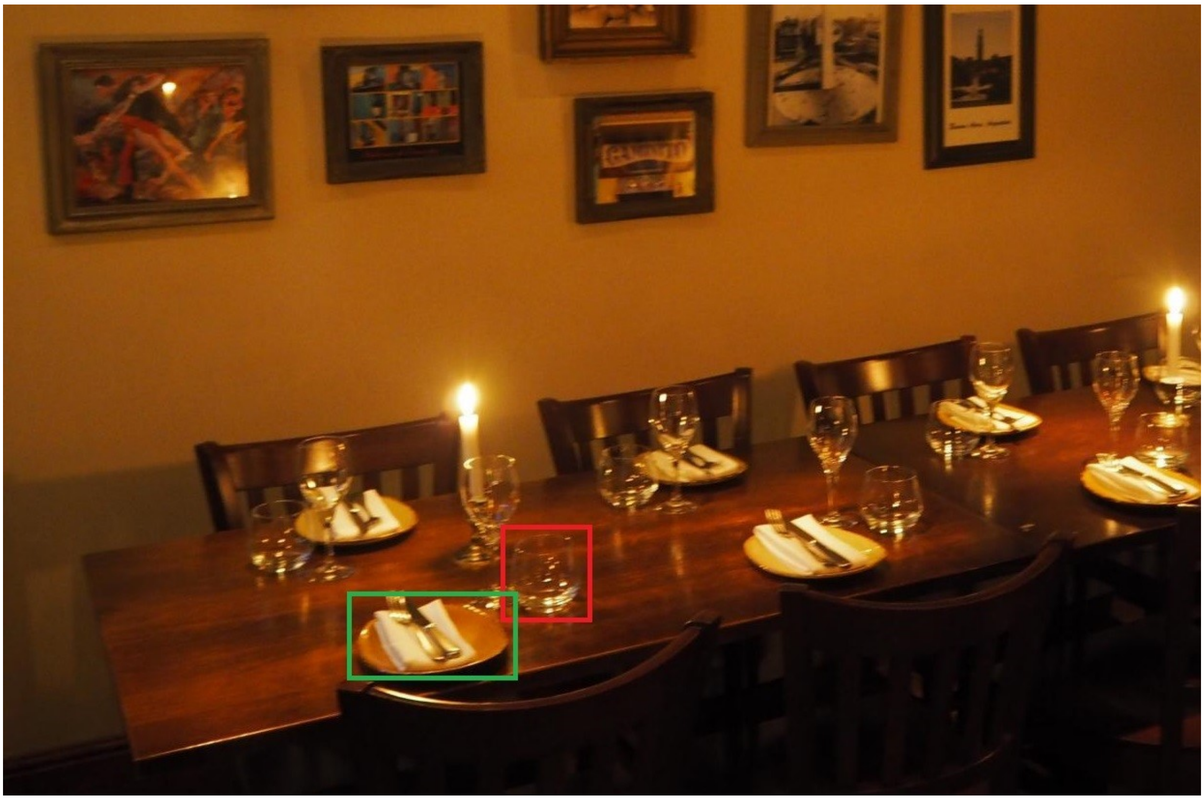


Figure 5–1 LocatedNear facts assist the detection of vague objects: if a set of knife, fork and plate is on the table, one may believe there is a glass beside based on the commonsense, even though these objects are hardly visible due to low light.

plex image scenes[116]. See Figure 5–1 for an example.

2. This commonsense knowledge can benefit reasoning related to spatial facts and physical scenes in reading comprehension, question answering, etc.[117]
3. Existing knowledge bases have very few facts for this relation (*ConceptNet 5.5* has only 49 triples of LocatedNear relation).

We propose two novel tasks in extracting LocatedNear relation from textual corpora. One is a sentence-level relation classification problem which judges whether or not a sentence describes two objects (mentioned in the sentence) being physically close by. The other task is to produce a ranked list of LocatedNear facts with the given classified results of large number of sentences. We believe both two tasks can be used to automatically populate and complete existing commonsense knowledge bases.

Additionally, we create two benchmark datasets for evaluating LocatedNear relation extraction systems on the two tasks: one is 5,000 sentences each describing a scene of two physical objects and with a label indicating if the two objects are co-located in the scene; the other consists of 500 pairs of objects with human-annotated scores indicating confidences that a certain pair of objects are commonly located near in real life.[1]

We propose several methods to solve the tasks including feature-based models and LSTM-based neural architectures. The proposed neural architecture compares favorably with the current state-of-the-art method for general-purpose relation classification problem. From our relatively smaller proposed datasets, we extract in total 2,067 new LocatedNear triples that are not in *ConceptNet*.

5.2 Sentence-level LocatedNear Relation Classification

Problem Statement Given a sentence s mentioning a pair of physical objects $<e_i, e_j>$, we call $<s, e_i, e_j>$ an *instance*. For each instance, the problem is to determine whether e_i and e_j are located near each other in the physical scene described in the sentence s. For example, suppose e_i is “dog", e_j is “cat”, and s = “*The King puts his dog and cat on the table.*”. As it is true that the two objects are located near in this sentence, a successful classification model is expected to label this instance as *True*. However, if s_2 = “*My dog is older than her cat.*”, then the label of the instance $<s_2, e_i, e_j>$ is *False*, because s_2 just talks about a comparison in age. In the following subsections, we present two different kinds of baseline methods for this binary classification task: feature-based methods and LSTM-based neural architectures.

5.2.1 Feature-based Methods

Our first baseline method is an SVM classifier based on following features commonly used in many relation extraction models[91]:

1. *Bag of Words (BW)*: the set of words that ever appeared in the sentence.
2. *Bag of Path Words (BPW)*: the set of words that appeared on the shortest dependency path between objects e_i and e_j in the dependency tree of the sentence s, plus the words in the two subtrees rooted at e_i and e_j in the tree.
3. *Bag of Adverbs and Prepositions (BAP)*: the existence of adverbs and prepositions in the sentence as binary features.
4. *Global Features (GF)*: the length of the sentence, the number of nouns, verbs, adverbs, adjectives, determiners, prepositions and punctuations in the whole sentence.

[1] https://github.com/adapt-sjtu/commonsense-locatednear

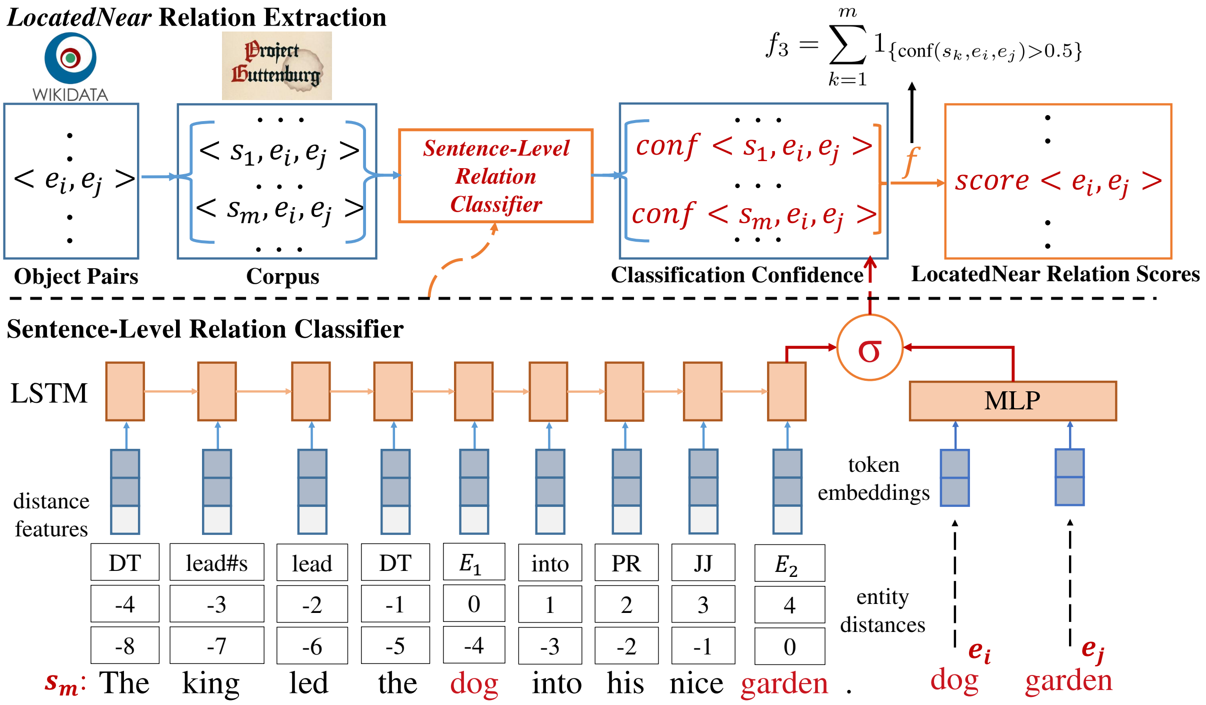

LocatedNear Relation Extraction
WIKIDATA
Project Gutenberg
$f_3 = \sum_{k=1}^{m} 1_{\{\mathrm{conf}(s_k,e_i,e_j)>0.5\}}$
$< e_i, e_j >$
$< s_1, e_i, e_j >$
$< s_m, e_i, e_j >$
Sentence-Level Relation Classifier
$conf < s_1, e_i, e_j >$
$conf < s_m, e_i, e_j >$
f
$score < e_i, e_j >$
Object Pairs
Corpus
Classification Confidence
LocatedNear Relation Scores
Sentence-Level Relation Classifier
LSTM
σ
MLP
token embeddings
distance features
DT lead#s lead DT E_1 into PR JJ E_2
-4 -3 -2 -1 0 1 2 3 4
-8 -7 -6 -5 -4 -3 -2 -1 0
entity distances
e_i
e_j
s_m: The king led the dog into his nice garden .
dog
garden

Figure 5–2 Framework with a LSTM-based classifier

5. *Shortest Dependency Path features (SDP)*: the same features as with GF but in dependency parse trees of the sentence and the shortest path between e_i and e_j, respectively.
6. *Semantic Similarity features (SS)*: the cosine similarities between the pre-trained *GloVe* word embeddings[22] of the two object words.

We evaluate *linear* and *RBF* kernels with different parameter settings, and find the *RBF* kernel with $\{C = 100, \gamma = 10^{-3}\}$ performs the best overall.

5.2.2 LSTM-based Neural Architectures

Although above features are both informative and easy to implement, they involve little sequential information such as the word order. LSTMs[118] are widely used in relation classification[59, 91, 99, 100]. capturing not only the input to output but also the sequential relationships. We observe that the existence of LocatedNear relation in an instance $<s,e_1,e_2>$ depends on two major information sources: one is from the semantic and syntactical features of sentence s and the other is from the object pair $<e_1,e_2>$. By this intuition, we design our LSTM-based model with two parts, shown in lower part of Figure 5–2. The left part is for encoding the syntactical and semantic information of the sentence s, while the right part is encoding the

Level	Examples
Objects	E_1, E_2
Lemma	open, lead, into, ...
Dependency Role	open#s, open#o, into#o, ...
POS Tag	DT, PR, CC, JJ, ...

Table 5–1 Examples of four types of tokens during sentence normalization. (#s stands for subjects and #o for objects)

semantic similarity between the pre-trained word embeddings of e_1 and e_2.

Solely relying on the original word sequence of a sentence s has two problems: (i) the irrelevant words in the sentence can introduce noise into the model; (ii) the large vocabulary of original sentences induce too many parameters, which may cause over-fitting. For example, given two sentences "*The king led the dog into his nice garden.*" and "*A criminal led the dog into a poor garden.*". The object pair is <*dog, garden*> in both sentences. The two words "*lead*" and "*into*" are essential for determining whether the object pair is located near, but they are not attached with due importance. Also, the semantic differences between irrelevant words, such as "king" and "criminal", "beautiful" and "poor", are not useful to the co-location relation between the "dog" and "garden", and thus tend to act as noise.

To address above issues, we propose utilizing POS (Part-of-Speech) tags instead to capture more syntactical information and reduce the vocabulary size. However, solely doing this loses too much semantic dependency between the words.

Sentence Normalization. To address the above issues, we propose a normalized sentence representation method merging the three most important and relevant kinds of information about each instance: lemmatized forms, POS (Part-of-Speech) tags and dependency roles. We first replace the two nouns in the object pair as "E_1" and "E_2", and keep the lemmatized form of the original words for all the *verbs, adverbs and prepositions*, which are highly relevant to describing physical scenes. Then, we replace the *subjects and direct objects* of the *verbs and prepositions* (`nsubj`, `dobj` for verbs and `case` for prepositions in dependency parse trees) with special tokens indicating their dependency roles. For the remaining words, we simply use their POS tags to replace the originals. The four kinds of tokens are illustrated in Table 5–1. Figure 5–2 shows a real example of our normalized sentence representation, where the object pair of interest is <*dog, garden*>.

Apart from the normalized tokens of the original sequence, to capture more structural

The	*king*	*opened*	*the*	*door*	*and*	*led*	*the*	***dog***	*into*	*his*	*nice*	***garden**.*
DT	open#s	open	DT	open#o	CC	lead	DT	E_1	into	PR	JJ	E_2.

Table 5–2 Sentence Normalization Example

information, we also encode the distances from each token to E_1 and E_2 respectively. Such *position embeddings* (position/distance features) are proposed by[119] with the intuition that information needed to determine the relation between two target nouns normally comes from the words which are close to the target nouns.

The original sentence is first transformed to normalized sequence described above. We adopt this feature because it can help LSTM keep track of the position of E_1 and E_2, better knowing *where* the two object words are. Then, we leverage LSTM to encode the whole sequence of the tokens of normalized representation plus position embedding. In the meantime, two pretrained *GloVe* word embeddings[22] of the original two physical object words are fed into a hidden dense layer.

Finally, we concatenate both outputs and then use `sigmoid` activation function to obtain the final prediction. We choose to use the popular binary cross-entropy as our loss function, and RMSProp as the optimizer. We apply a dropout rate[120] of 0.5 in the LSTM and embedding layer to prevent overfitting.

5.3 LocatedNear Relation Extraction

The upper part of Figure 5–2 shows the overall workflow of our automatic framework to mine LocatedNear relations from raw text. We first construct a vocabulary of physical objects and generate all candidate instances. For each sentence in the corpus, if a pair of physical objects e_i and e_j appear as nouns in a sentence s, then we apply our sentence-level relation classifier on this instance. The relation classifier yields a probabilistic score s indicating the confidence of the instance in the existence of LocatedNear relation. Finally, all scores of the instances from the corpus are grouped by the object pairs and aggregated, where each object pair is associated with a final score. These mined physical pairs with scores can easily be integrated into existing commonsense knowledge base.

More specifically, for each object pair $<e_i, e_j>$, we find all the m sentences in our corpus mentioning both objects. We classify the m instances with the sentence-level relation classifier and obtain confidence scores for each instance, then feed them into a heuristic scoring function

f to obtain the final aggregated score for the given object pair. We propose the following 5 choices of f considering accumulation and threshold:

$$f_0 = m \tag{5–1}$$

$$f_1 = \sum_{k=1}^{m} \mathrm{conf}(s_k, e_i, e_j) \tag{5–2}$$

$$f_2 = \frac{1}{m} \sum_{k=1}^{m} \mathrm{conf}(s_k, e_i, e_j) \tag{5–3}$$

$$f_3 = \sum_{k=1}^{m} 1_{\{\mathrm{conf}(s_k, e_i, e_j) > 0.5\}} \tag{5–4}$$

$$f_4 = \frac{1}{m} \sum_{k=1}^{m} 1_{\{\mathrm{conf}(s_k, e_i, e_j) > 0.5\}} \tag{5–5}$$

5.4 Evaluation

	Random	Majority	SVM	SVM(-BW)	SVM(-BPW)	SVM(-BAP)	SVM(-GF)	SVM(-SDP)	SVM(-SS)	DRNN	LSTM+Word	LSTM+POS	LSTM+Norm
Acc.	0.500	0.551	0.584	0.577	0.556	0.563	**0.605**	0.579	0.584	0.635	0.637	0.641	**0.653**
P	0.551	0.551	0.606	0.579	0.567	0.573	**0.616**	0.597	0.605	**0.658**	0.635	0.650	0.654
R	0.500	1.000	0.702	0.675	0.681	**0.811**	0.751	0.728	0.708	0.702	**0.800**	0.751	0.784
F1	0.524	0.710	0.650	0.623	0.619	0.672	**0.677**	0.656	0.652	0.679	0.708	0.697	**0.713**

Table 5–3 Performance of baselines on co-location classification task with ablation. (Acc.=Accuracy, P=Precision, R=Recall, "-" means without certain feature)

In this section, we first present our evaluation of our proposed methods and the state-of-the-art general relation classification model on the first task. Then, we evaluate the quality of the new LocatedNear triples we extracted.

5.4.1 Sentence-level LocatedNear Relation Classification

We evaluate the proposed methods against the state-of-the-art general domain relation classification model (DRNN)[121]. The results are shown in Table 5–3. For feature-based SVM, we do feature ablation on each of the 6 feature types. For LSTM-based model, we experiment on variants of input sequence of original sentence: "LSTM+Word" uses the original words as the input tokens; "LSTM+POS" uses only POS tags as the input tokens; "LSTM+Norm" uses the tokens of sequence after sentence normalization. Besides, we add two naive baselines: "Random" baseline method classifies the instances into two classes with equal probability. "Majority" baseline method considers all the instances to be positive.

From the results, we find that the SVM model without the Global Features performs best, which indicates that bag-of-word features benefit more in shortest dependency paths than on the whole sentence. Also, we notice that DRNN performs best (0.658) on precision but not significantly higher than LSTM+Norm (0.654). The experiment shows that LSTM+Word enjoys the highest recall score, while LSTM+Norm is the best one in terms of the overall performance. One reason is that the normalization representation reduces the vocabulary of input sequences, while also preserving important syntactical and semantic information. Another reason is that the LocatedNear relation are described in sentences decorated with prepositions/adverbs. These words are usually descendants of the object word in the dependency tree, outside of the shortest dependency paths. Thus, DRNN cannot capture the information from the words belonging to the descendants of the two object words in the tree, but this information is well captured by LSTM+Norm.

5.4.2 LocatedNear Relation Extraction

Once we have obtained the probability score for each instance using LSTM+Norm, we can extract LocatedNear relation using the scoring function f. We compare the performance of 5 different heuristic choices of f, by quantitative results. We rank 500 commonsense LocatedNear object pairs described in Section 5.3. Table 5–4 shows the ranking results using *Mean Average Precision* (MAP) and *Precision* at K as the metrics. Accumulative scores (f_1 and f_3) generally do better. Thus, we choose $f = f_3$ with a MAP score of 0.59 as the scoring function.

Qualitatively, we show 15 object pairs with some of the highest f_3 scores in Table 5–5. Setting a threshold of 40.0 for f_3, which is the minimum non-zero f_3 score for all true object pairs in the LocatedNear object pairs data set (500 pairs), we obtain a total of 2,067 LocatedNear

f	MAP	P@50	P@100	P@200	P@300
f_0	0.42	0.40	0.44	0.42	0.38
f_1	0.58	**0.70**	0.60	0.53	**0.44**
f_2	0.48	0.56	0.52	0.49	0.42
f_3	**0.59**	0.68	**0.63**	**0.55**	**0.44**
f_4	0.56	0.40	0.48	0.50	0.42

Table 5–4 Ranking results of scoring functions.

(door, room)	(boy, girl)	(cup, tea)
(ship, sea)	(house, garden)	(arm, leg)
(fire, wood)	(house, fire)	(horse, saddle)
(fire, smoke)	(door, hall)	(door, street)
(book, table)	(fruit, tree)	(table, chair)

Table 5–5 Top object pairs returned by best performing scoring function f_3

relations, with a precision of 68% by human inspection.

5.5 Related Work

Classifying relations between entities in a certain sentence plays a key role in NLP applications and thus has been a hot research topic recently. Feature-based methods[92] and neural network techniques[99, 100] are most common. Xu, Mou, Li, et al. (2015) introduce multi-channel SDP-based LSTM model to classify relations incooperating several different kinds of information of a sentence improved by Xu, Jia, Mou, et al. (2016), which performed best on SemEval-2010 Task 8 and is one of our baseline methods.

The most related work to ours is the extraction of visual commonsense knowledge by Yatskar, Ordonez, Farhadi (2016). This work learns the textual representation of seven types of fine-grained visual relations using textual caption for the image in MS-COCO dataset[122]. Another important related work is from Li, Taheri, Tu, et al. (2016), which enriches several popular relations in *ConceptNet* with little textual information from real large corpora. However, LocatedNear relation was not studied in this work, while this relation is extremely scarce in *ConceptNet* and has its own distinctiveness.

Chapter 6 Extracting Cross-Cultural Differences and Similarities in Social Media

6.1 Introduction

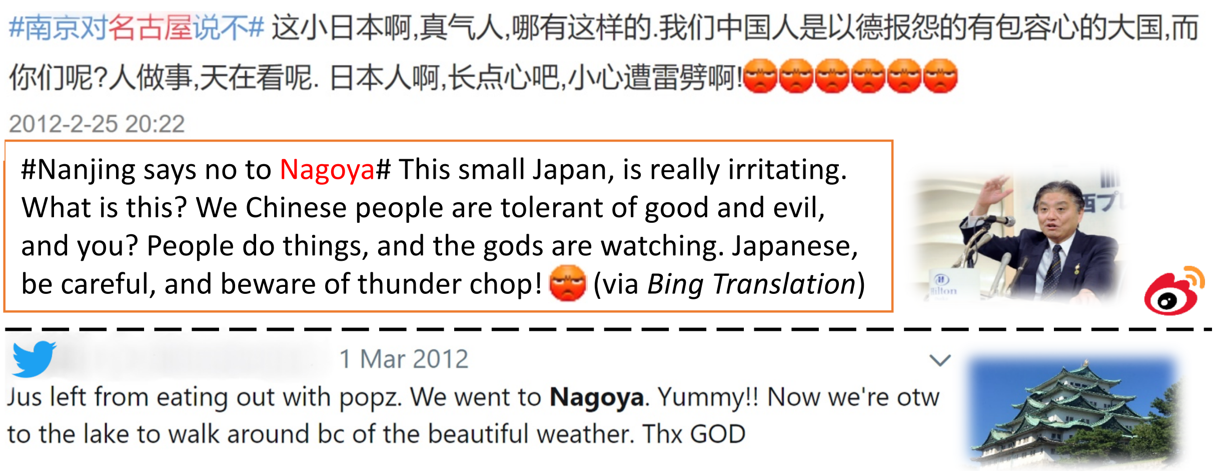
Computing similarities between terms is one of the most fundamental computational tasks in natural language understanding. Much work has been done in this area, most notably using the distributional properties drawn from large monolingual textual corpora to train vector representations of words or other linguistic units[22, 123]. However, computing cross-cultural similarities of terms between different cultures is still an open research question, which is important in cross-lingual natural language understanding. In this chapter, we address cross-cultural research questions such as these:

1. *Were there any cross-cultural differences between Nagoya (a city in Japan) for native English speakers and 名古屋 (Nagoya in Chinese) for Chinese people in 2012?*
2. *What English terms can be used to explain "浮云" (a Chinese slang term)?*

These kinds of questions about cross-cultural differences and similarities are important in cross-cultural social studies, multi-lingual sentiment analysis, culturally sensitive machine translation, and many other NLP tasks, especially in social media. We propose two novel tasks in mining them from social media.

The first task (Section 6.4) is to mine cross-cultural differences in the perception of named entities (e.g., persons, places and organizations). Back in 2012, in the case of "Nagoya", many native English speakers posted their pleasant travel experiences in Nagoya on Twitter. However, Chinese people overwhelmingly greeted the city with anger and condemnation on *Weibo* (a Chinese version of Twitter), because the city mayor denied the truthfulness of the Nanjing Massacre. Figure 6–1 illustrates two example microblog messages about Nagoya in Twitter and Weibo respectively.

The second task (Section 6.5) is to find similar terms for slang across cultures and languages. Social media is always a rich soil where slang terms emerge in many cultures. For example, "浮云" literally means "floating clouds", but now almost equals to "nothingness" on the Chinese web. Our experiments show that well-known online machine translators such as *Google Translate* are only able to translate such slang terms to their literal meanings, even under



#南京对名古屋说不# 这小日本啊,真气人,哪有这样的.我们中国人是以德报怨的有包容心的大国,而你们呢?人做事,天在看呢. 日本人啊,长点心吧,小心遭雷劈啊!😡😡😡😡😡😡

2012-2-25 20:22

#Nanjing says no to Nagoya# This small Japan, is really irritating. What is this? We Chinese people are tolerant of good and evil, and you? People do things, and the gods are watching. Japanese, be careful, and beware of thunder chop! 😡 (via *Bing Translation*)

[redacted] 1 Mar 2012

Jus left from eating out with popz. We went to **Nagoya**. Yummy!! Now we're otw to the lake to walk around bc of the beautiful weather. Thx GOD

Figure 6–1 Two social media messages about Nagoya from different cultures in 2012

clear contexts where slang meanings are much more appropriate.

Enabling intelligent agents to understand such cross-cultural knowledge can benefit their performances in various cross-lingual language processing tasks. Both tasks share the same core problem, which is **how to compute cross-cultural differences (or similarities) between two terms from different cultures.** A term here can be either an ordinary word, an entity name, or a slang term. We focus on names and slang in this chapter for they convey more social and cultural connotations.

There are many works on cross-lingual word representation[124] to compute general cross-lingual similarities[125]. Most existing models require bilingual supervision such as aligned parallel corpora, bilingual lexicons, or comparable documents[126–128]. However, they do not purposely preserve social or cultural characteristics of named entities or slang terms, and the required parallel corpora are rare and expensive.

In this chapter, we propose a lightweight yet effective approach to project two incompatible monolingual word vector spaces into a single bilingual word vector space, known as social vector space (*SocVec*). A key element of SocVec is the idea of "bilingual social lexicon", which contains bilingual mappings of selected words reflecting psychological processes, which we believe are central to capturing the socio-linguistic characteristics. Our contribution in this chapter is two-fold:

1. We present an effective approach (SocVec) to mine cross-cultural similarities and differences of terms, which could benefit research in machine translation, cross-cultural social media analysis, and other cross-lingual research in natural language processing and computational social science.

2. We propose two novel and important tasks in cross-cultural social studies and social media analysis. Experimental results on our annotated datasets show that the proposed method outperforms many strong baseline methods.

6.2 The *SocVec* Framework

In this section, we first discuss the intuition behind our model, the concept of "social words" and our notations. Then, we present the overall workflow of our approach. We finally describe the *SocVec* framework in detail.

6.2.1 Problem Statement

We choose (English, Chinese) to be the target language pair throughout this chapter for the salient cross-cultural differences between the east and the west[1]. Given an English term W and a Chinese term U, the core research question is how to compute a similarity score, $ccsim(W, U)$, to represent the *cross-cultural similarities* between them.

We cannot directly calculate the similarity between the monolingual word vectors of W and U, because they are trained separately and the semantics of dimension are not aligned. Thus, the challenge is to devise a way to compute similarities across two different vector spaces while retaining their respective cultural characteristics.

A very intuitive solution is to firstly translate the Chinese term U to its English counterpart U' through a Chinese-English bilingual lexicon, and then regard $ccsim(W, U)$ as the (cosine) similarity between W and U' with their monolingual word embeddings. However, this solution is not promising in some common cases for three reasons:

1. if U is an OOV (Out of Vocabulary) term, e.g., a novel slang term, then there is probably no translation U' in bilingual lexicons.
2. if W and U are names referring to the same named entity, then we have $U' = W$. Therefore, $ccsim(W, U)$ is just the similarity between W and itself, and we cannot capture any cross-cultural differences with this method.
3. this approach does not explicitly preserve the cultural and social contexts of the terms.

To overcome the above problems, our intuition is to project both English and Chinese word vectors into a single third space, known as *SocVec*, and the projection is supposed to purposely carry cultural features of terms.

[1] Nevertheless, the techniques are language independent and thus can be utilized for any language pairs so long as the necessary resources outlined in Section 6.2.3 are available.

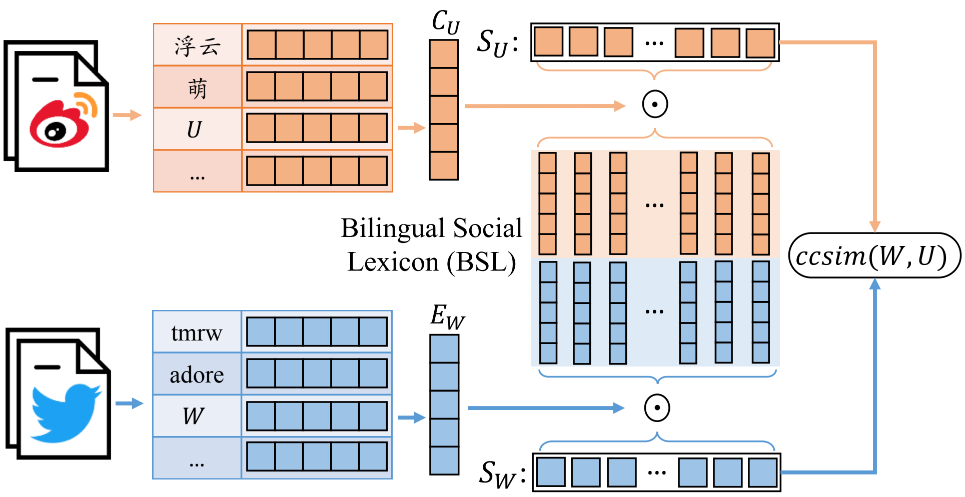


Figure 6–2 Workflow for computing the cross-cultural similarity between an English word W and a Chinese word U, denoted by $ccsim(W,U)$

6.2.2 Social Words and Our Notations

Some research in psychology and sociology[129, 130] show that culture can be highly related to emotions and opinions people express in their discussions. As suggested by[131], we thus define the concept of "**social word**" as the words directly reflecting opinion, sentiment, cognition and other human psychological processes[1], which are important to capturing cultural and social characteristics. Both[132] and[133] find such *social words* are most effective culture/socio-linguistic features in identifying cross-cultural differences.

We use these notations throughout the chapter: *CnVec* and *EnVec* denote the Chinese and English word vector space, respectively; *CSV* and *ESV* denote the Chinese and English social word vocab; *BL* means Bilingual Lexicon, and *BSL* is short for Bilingual Social Lexicon; finally, we use $\mathbf{E_x}$, $\mathbf{C_x}$ and $\mathbf{S_x}$ to denote the word vectors of the word x in *EnVec*, *CnVec* and *SocVec* spaces respectively.

6.2.3 Overall Workflow

Figure 6–2 shows the workflow of our framework to construct the *SocVec* and compute $ccsim(W,U)$. Our proposed *SocVec* model attacks the problem with the help of three low-cost external resources: (i) an English corpus and a Chinese corpus from social media; (ii) an English-to-Chinese bilingual lexicon (*BL*); (iii) an English social word vocabulary (*ESV*) and a

[1] Example social words in English include *fawn, inept, tremendous, gratitude, terror, terrific, loving, traumatic*, etc. We discuss the sources of such social words in Section 6.3.

Chinese one (*CSV*).

We train English and Chinese word embeddings (*EnVec* and *CnVec*) on the English and Chinese social media corpus respectively. Then, we build a *BSL* from the *CSV*, *ESV* and *BL* (see Section 6.2.4). The *BSL* further maps the previously incompatible *EnVec* and *CnVec* into a single common vector space *SocVec*, where two new vectors, S_W for W and S_U for U, are finally comparable.

6.2.4 Building the BSL

The process of building the *BSL* is illustrated in Figure 6–3. We first extract our bilingual lexicon (*BL*), where confidence score w_i represents the probability distribution on the multiple translations for each word. Afterwards, we use BL to translate each social word in the *ESV* to a set of Chinese words and then filter out all the words that are not in the *CSV*. Now, we have a set of Chinese social words for each English social word, which is denoted by a "translation set". The final step is to generate a Chinese "pseudo-word" for each English social word using their corresponding translation sets. A "pseudo-word" can be either a real word that is the most representative word in the translation set, or an imaginary word whose vector is a certain combination of the vectors of the words in the translation set.

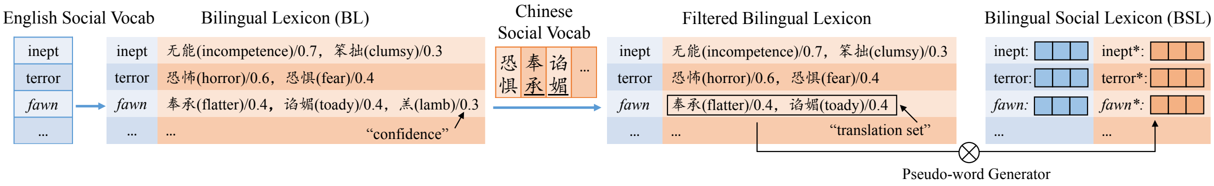

English Social Vocab
inept
terror
fawn
...
Bilingual Lexicon (BL)
inept　无能(incompetence)/0.7，笨拙(clumsy)/0.3
terror　恐怖(horror)/0.6，恐惧(fear)/0.4
fawn　奉承(flatter)/0.4，谄媚(toady)/0.4，羔(lamb)/0.3
...　...
"confidence"
Chinese Social Vocab
恐惧
奉承
谄媚
...
Filtered Bilingual Lexicon
inept　无能(incompetence)/0.7，笨拙(clumsy)/0.3
terror　恐怖(horror)/0.6，恐惧(fear)/0.4
fawn　奉承(flatter)/0.4，谄媚(toady)/0.4
...　...
"translation set"
Bilingual Social Lexicon (BSL)
inept:
terror:
fawn:
...
inept*:
terror*:
fawn*:
...
Pseudo-word Generator

Figure 6–3 Generating an entry in the BSL for "*fawn*" and its pseudo-word "*fawn**"

For example, in Figure 6–3, the English social word "*fawn*" has three Chinese translations in the bilingual lexicon, but only two of them (underlined) are in the CSV. Thus, we only keep these two in the translation set in the filtered bilingual lexicon. The pseudo-word generator takes the word vectors of the two words (in the black box), namely 奉承 (flatter) and 谄媚 (toady), as input, and generates the pseudo-word vector denoted by "*fawn**". Note that the direction of building *BSL* can also be from Chinese to English, in the same manner. However, we find that the current direction gives better results due to the better translation quality of our *BL* in this direction.

Given an English social word, we denote $\mathbf{t_i}$ as the i^{th} Chinese word of its translation set consisting of N social words. We design four intuitive types of pseudo-word generator as follows, which are tested in the experiments:

(1) Max. Maximum of the values in each dimension, assuming dimensionality is K:

$$\text{Pseudo}(\mathbf{C_{t_1}}, ..., \mathbf{C_{t_N}}) = \begin{bmatrix} max(C^{(1)}_{t_1}, ..., C^{(1)}_{t_N}) \\ \vdots \\ max(C^{(K)}_{t_1}, ..., C^{(K)}_{t_N}) \end{bmatrix}^{\mathrm{T}}$$

(2) Avg. Average of the values in every dimension:

$$\text{Pseudo}(\mathbf{C_{t_1}}, ..., \mathbf{C_{t_N}}) = \frac{1}{N} \sum_{i}^{N} \mathbf{C_{t_i}}$$

(3) WAvg. Weighted average value of every dimension with respect to the translation confidence:

$$\text{Pseudo}(\mathbf{C_{t_1}}, ..., \mathbf{C_{t_N}}) = \frac{1}{N} \sum_{i}^{N} w_i \mathbf{C_{t_i}}$$

(4) Top. The most confident translation:

$$\text{Pseudo}(\mathbf{C_{t_1}}, ..., \mathbf{C_{t_N}}) = \mathbf{C_{t_k}}, k = \underset{i}{\text{argmax}}\, w_i$$

Finally, the *BSL* contains a set of English-Chinese word vector pairs, where each entry represents an English social word and its Chinese pseudo-word based on its “translation set”.

6.2.5 Constructing the SocVec Space

Let B_i denote the English word of the i^{th} entry of the *BSL*, and its corresponding Chinese pseudo-word is denoted by B_i^*. We can project the English word vector $\mathbf{E_W}$ into the *SocVec* space by computing the cosine similarities between $\mathbf{E_W}$ and each English word vector in *BSL* as values on SocVec dimensions, effectively constructing a new vector $\mathbf{S_W}$ of size L. Similarly, we map a Chinese word vector $\mathbf{C_U}$ to be a new vector $\mathbf{S_U}$. $\mathbf{S_W}$ and $\mathbf{S_U}$ belong to the same vector space *SocVec* and are comparable. The following equation illustrates the projection, and how to compute *ccsim*[1].

$$\begin{aligned} ccsim(W, U) &:= f(\mathbf{E}_{\mathrm{W}}, \mathbf{C}_{\mathrm{U}}) \\ &= sim\left(\begin{bmatrix} cos(\mathbf{E}_{\mathrm{W}}, \mathbf{E_{B_1}}) \\ \vdots \\ cos(\mathbf{E}_{\mathrm{W}}, \mathbf{E_{B_L}}) \end{bmatrix}^{\mathrm{T}}, \begin{bmatrix} cos(\mathbf{C}_{\mathrm{U}}, \mathbf{C_{B_1^*}}) \\ \vdots \\ cos(\mathbf{C}_{\mathrm{U}}, \mathbf{C_{B_L^*}}) \end{bmatrix}^{\mathrm{T}} \right) \\ &= sim(\mathbf{S}_{\mathrm{W}}, \mathbf{S}_{\mathrm{U}}) \end{aligned}$$

[1] The function *sim* is a generic similarity function, for which several metrics are considered in experiments.

For example, if W is “Nagoya” and U is ‘‘名古屋”, we compute the cosine similarities between “Nagoya” and each English social word in the *BSL* with their monolingual word embeddings in English. Such similarities compose $\mathbf{S}_{\text{nagoya}}$. Similarly, we compute the cosine similarities between ‘‘名古屋” and each Chinese pseudo-word, and compose the social word vector $\mathbf{S}_{\text{名古屋}}$.

In other words, for each culture/language, the new word vectors like S_{W} are constructed based on the monolingual similarities of each word to the vectors of a set of task-related words (“social words” in our case). This is also a significant part of the novelty of our transformation method.

6.3 Experimental Setup

Prior to evaluating *SocVec* with our two proposed tasks in Section 6.4 and Section 6.5, we present our preparation steps as follows.

Social Media Corpora Our English Twitter corpus is obtained from Archive Team’s Twitter stream grab[1]. The Chinese Weibo corpus comes from Open Weiboscope Data Access[2][134]. Both corpora cover the whole year of 2012. We then randomly down-sample each corpus to 100 million messages where each message contains at least 10 characters, normalize the text[135], lemmatize the text[72] and use LTP[136] to perform word segmentation for the Chinese corpus.

Entity Linking and Word Embedding Entity linking is a preprocessing step which links various entity mentions (surface forms) to the identity of corresponding entities. For the Twitter corpus, we use Wikifier[137, 138], a widely used entity linker in English. Because no sophisticated tool for Chinese short text is available, we implement our own tool that is greedy for high precision. We train English and Chinese monolingual word embedding respectively using *word2vec*’s skip-gram method with a window size of 5[139].

Bilingual Lexicon Our bilingual lexicon is collected from *Microsoft Translator*[3], which translates English words to multiple Chinese words with confidence scores. Note that all named entities and slang terms used in the following experiments are excluded from this bilingual lexicon.

Social Word Vocabulary Our social word vocabularies come from *Empath*[140] and *OpinionFinder*[141] for English, and *TextMind*[142] for Chinese. Empath is similar to LIWC[131],

[1] https://archive.org/details/twitterstream

[2] http://weiboscope.jmsc.hku.hk/datazip/

[3] http://www.bing.com/translator/api/Dictionary/Lookup?from=en&to=zh-CHS&text=<input_word>

but has more words and more categories and is publicly available. We manually select 91 categories of words that are relevant to human perception and psychological processes following[133]. OpinionFinder consists of words relevant to opinions and sentiments, and TextMind is a Chinese counterpart for Empath. In summary, we obtain 3,343 words from Empath, 3,861 words from OpinionFinder, and 5,574 unique social words in total.

6.4 Task 1: Mining cross-cultural differences of named entities

Task definition: This task is to discover and quantify cross-cultural differences of concerns towards named entities. Specifically, the input in this task is a list of 700 named entities of interest and two monolingual social media corpora; the output is the scores for the 700 entities indicating the cross-cultural differences of the concerns towards them between two corpora. The ground truth is from the labels collected from human annotators.

6.4.1 Ground Truth Scores

[143] states that the meaning of words is evidenced by the contexts they occur with. Likewise, we assume that the cultural properties of an entity can be captured by the terms they always co-occur within a large social media corpus. Thus, for each of randomly selected 700 named entities, we present human annotators with two lists of 20 most co-occurred terms within Twitter and Weibo corpus respectively.

Our annotators are instructed to rate the topic-relatedness between the two word lists using one of following labels: “very different”, “different”, “hard to say”, “similar” and “very similar”. We do this for efficiency and avoiding subjectivity. As the word lists presented come from social media messages, the social and cultural elements are already embedded in their chances of occurrence. All four annotators are native Chinese speakers but have excellent command of English and lived in the US extensively, and they are trained with many selected examples to form shared understanding of the labels. The inter-annotator agreement is 0.67 by Cohen’s kappa coefficient, suggesting substantial correlation[144].

6.4.2 Baseline and Our Methods

We propose eight baseline methods for this novel task: **distribution-based** methods (BL-JS, E-BL-JS, and WN-WUP) compute cross-lingual relatedness between two lists of the words surrounding the input English and Chinese terms respectively ($\mathcal{L}_E$ and $\mathcal{L}_C$); **transformation-based**

Table 6–1 Selected culturally different entities with summarized Twitter and Weibo's trending topics

Entity	Twitter topics	Weibo topics
Maldives	coup, president Nasheed quit, political crisis	holiday, travel, honeymoon, paradise, beach
Nagoya	tour, concert, travel, attractive, Osaka	Mayor Takashi Kawamura, Nanjing Massacre, denial of history
Quebec	Conservative Party, Liberal Party, politicians, prime minister, power failure	travel, autumn, maples, study abroad, immigration, independence
Philippines	gunman attack, police, quake, tsunami	South China Sea, sovereignty dispute, confrontation, protest
Yao Ming	NBA, Chinese, good player, Asian	patriotism, collective values, Jeremy Lin, Liu Xiang, Chinese Law maker, gold medal superstar
USC	college football, baseball, Stanford, Alabama, win, lose	top destination for overseas education, Chinese student murdered, scholars, economics, Sino American politics

methods (LTrans and BLex) compute the vector representation in English and Chinese corpus respectively, and then train a transformation; MCCA, MCluster and Duong are three typical **bilingual word representation models** for computing general cross-lingual word similarities.

The $\mathcal{L}_E$ and $\mathcal{L}_C$ in the BL-JS and WN-WUP methods are the same as the lists that annotators judge. **BL-JS** (*Bilingual Lexicon Jaccard Similarity*) uses the bilingual lexicon to translate $\mathcal{L}_E$ to a Chinese word list $\mathcal{L}_E^*$ as a medium, and then calculates the Jaccard Similarity between $\mathcal{L}_E^*$ and $\mathcal{L}_C$ as J_{EC}. Similarly, we compute J_{CE}. Finally, we regard $(J_{EC} + J_{CE})/2$ as the score of this named entity. **E-BL-JS** (*Embedding-based Jaccard Similarity*) differs from BL-JS in that it instead compares the two lists of words gathered from the rankings of word embedding similarities between the name of entities and all English words and Chinese words respectively. **WN-WUP** (*WordNet Wu-Palmer Similarity*) uses Open Multilingual Wordnet[145] to compute the average similarities over all English-Chinese word pairs constructed from the $\mathcal{L}_E$ and $\mathcal{L}_C$.

We follow the steps of[146] to train a linear transformation (**LTrans**) matrix between *EnVec* and *CnVec*, using 3,000 translation pairs with maximum confidences in the bilingual lexicon. Given a named entity, this solution simply calculates the cosine similarity between the vector of its English name and the *transformed* vector of its Chinese name. **BLex** (*Bilingual Lexicon Space*) is similar to our *SocVec* but it does not use any social word vocabularies but uses bilingual lexicon entries as pivots instead.

MCCA[147] takes two trained monolingual word embeddings with a bilingual lexicon as input, and develop a bilingual word embedding space. It is extended from the work of[148], which performs slightly worse in the experiments. **MCluster**[147] requires re-training the bilingual word embeddings from the two mono-lingual corpora with a bilingual lexicon. Similarly, **Duong**[149] retrains the embeddings from monolingual corpora with an EM-like training algorithm. We also use our BSL as the bilingual lexicon in these methods to investigate its effectiveness and generalizability. The dimensionality is tuned from $\{50, 100, 150, 200\}$ in all these bilingual word embedding methods.

With our constructed ***SocVec*** space, given a named entity with its English and Chinese names, we can simply compute the similarity between their *SocVec*s as its cross-cultural difference score. Our method is based on monolingual word embeddings and a BSL, and thus does not need the time-consuming re-training on the corpora.

6.4.3 Experimental Results

For qualitative evaluation, Table 6–1 shows some of the most culturally different entities mined by the SocVec method. The hot and trendy topics on Twitter and Weibo are manually summarized to help explain the cross-cultural differences. The perception of these entities diverges widely between English and Chinese social media, thus suggesting significant cross-cultural differences. Note that some cultural differences are time-specific. We believe such temporal variations of cultural differences can be valuable and beneficial for social studies as well. Investigating temporal factors of cross-cultural differences in social media can be an interesting future research topic in this task.

In Table 6–2, we evaluate the benchmark methods and our approach with three metrics: Spearman and Pearson, where correlation is computed between truth averaged scores (quantifying the labels from 1.0 to 5.0) and computed cultural difference scores from different methods; Mean Average Precision (MAP), which converts averaged scores as binary labels, by setting 3.0 as the threshold. The ***SocVec:opn*** considers only OpinionFinder as the ESV, while ***SocVec:all***

Table 6–2 Comparison of Different Methods

Method	**Spearman**	**Pearson**	**MAP**
BL-JS	0.276	0.265	0.644
WN-WUP	0.335	0.349	0.677
E-BL-JS	0.221	0.210	0.571
LTrans	0.366	0.385	0.644
BLex	0.596	0.595	0.765
MCCA-BL(100d)	0.325	0.343	0.651
MCCA-BSL(150d)	0.357	0.376	0.671
MCluster-BL(100d)	0.365	0.388	0.693
MCluster-BSL(100d)	0.391	0.425	0.713
Duong-BL(100d)	0.618	0.627	0.785
Duong-BSL(100d)	0.625	0.631	0.791
SocVec:opn	0.668	0.662	**0.834**
SocVec:all	**0.676**	**0.671**	**0.834**
SocVec:noun	0.564	0.562	0.756
SocVec:verb	0.615	0.618	0.779
SocVec:adj.	0.636	0.639	0.800

uses the union of Empath and OpinionFinder vocabularies[1].

Lexicon Ablation Test. To show the effectiveness of social words versus other type of words as the bridge between the two cultures, we also compare the results using sets of nouns (***SocVec:noun***), verbs (***SocVec:verb***) and adjectives (***SocVec:adj.***). All vocabularies under comparison are of similar sizes (around 5,000), indicating that the improvement of our method is significant. Results show that our *SocVec* models, and in particular, the *SocVec* model using the social words as cross-lingual media, performs the best.

Similarity Options. We also evaluate the effectiveness of four different similarity options in *SocVec*, namely, Pearson Correlation Coefficient (*PCorr.*), L1-normalized Manhattan distance (*L1+M*), Cosine Similarity (*Cos*) and L2-normalized Euclidean distance (*L2+E*). From Table 6–3, we conclude that among these four options, *Cos* and *L2+E* perform the best.

Pseudo-word Generators. Table 6–4 shows effect of using four pseudo-word generator

[1] The following tuned parameters are used in *SocVec* methods: 5-word context window, 150 dimensions monolingual word vectors, cosine similarity as the *sim* function, and “*Top*” as the pseudo-word generator.

Table 6–3 Different Similarity Functions

Similarity	**Spearman**	**Pearson**	**MAP**
PCorr.	0.631	0.625	0.806
L1 + M	0.666	0.656	0.824
Cos	**0.676**	0.669	**0.834**
L2 + E	**0.676**	**0.671**	**0.834**

Table 6–4 Different Pseudo-word Generators

Generator	**Spearman**	**Pearson**	**MAP**
Max.	0.413	0.401	0.726
Avg.	0.667	0.625	0.831
W.Avg.	0.671	0.660	0.832
Top	**0.676**	**0.671**	**0.834**

functions, from which we can infer that “*Top*” generator function performs best for it reduces some noisy translation pairs.

6.5 Task 2: Finding most similar words for slang across languages

Task Description: This task is to find the most similar English words of a given Chinese slang term in terms of its slang meanings and sentiment, and vice versa. The input is a list of English/Chinese slang terms of interest and two monolingual social media corpora; the output is a list of Chinese/English word sets corresponding to each input slang term. Simply put, for each given slang term, we want to find a set of the words in a different language that are most similar to itself and thus can help people understand it across languages. We propose Average Cosine Similarity (Section 6.5.3) to evaluate a method’s performance with the ground truth (presented below).

6.5.1 Ground Truth

Slang Terms. We collect the Chinese slang terms from an online Chinese slang glossary[1] consisting of 200 popular slang terms with English explanations. For English, we resort to a

[1] https://www.chinasmack.com/glossary

Gg	Bi	Bd	CC	LT
18.24	16.38	17.11	17.38	9.14
TransBL	MCCA	MCluster	Duong	SV
18.13	17.29	17.47	20.92	**23.01**

(a) Chinese Slang to English

Gg	Bi	Bd	LT	TransBL
6.40	15.96	15.44	7.32	11.43
MCCA	MCluster	Duong	SV	
15.29	14.97	15.13	**17.31**	

(b) English Slang to Chinese

Table 6–5 ACS Sum Results of Slang Translation

slang word list from OnlineSlangDictionary[1] with explanations and downsample the list to 200 terms.

Truth Sets. For each Chinese slang term, its truth set is a set of words extracted from its English explanation. For example, we construct the truth set of the Chinese slang term "二百五" by manually extracting significant words about its slang meanings (bold) in the glossary:

二百五: A ***foolish*** person who is lacking in sense but still ***stubborn***, ***rude***, and ***impetuous***. Similarly, for each English slang term, its Chinese word sets are the translation of the words hand picked from its English explanation.

6.5.2 Baseline and Our Methods

We propose two types of baseline methods for this task. The first is based on well-known *online translators*, namely Google (Gg), Bing (Bi) and Baidu (Bd). Note that experiments using them are done in August, 2017. Another baseline method for Chinese is CC-CEDICT[2] (CC), an online public Chinese-English dictionary, which is constantly updated for popular slang terms.

Considering situations where many slang terms have literal meanings, it may be unfair to retrieve target terms from such machine translators by solely inputing slang terms without specific contexts. Thus, we utilize example sentences of their slang meanings from some websites (mainly from Urban Dictionary[3]). The following example shows how we obtain the

[1] http://onlineslangdictionary.com/word-list/

[2] https://cc-cedict.org/wiki/

[3] http://www.urbandictionary.com/

Table 6–6 Bidirectional Slang Translation Examples Produced by SocVec

Slang	**Explanation**	**Google**	**Bing**	**Baidu**	**Ours**
浮云	something as ephemeral and unimportant as "passing clouds"	clouds	nothing	floating clouds	nothingness, illusion
水军	"water army", people paid to slander competitors on the Internet and to help shape public opinion	Water army	Navy	Navy	propaganda, complicit, fraudulent
floozy	a woman with a reputation for promiscuity	N/A	劣根性 (depravity)	荡妇 (slut)	骚货 (slut), 妖精 (promiscuous)
fruitcake	a crazy person, someone who is completely insane	水果蛋糕 (fruit cake)	水果蛋糕 (fruit cake)	水果蛋糕 (fruit cake)	怪诞 (bizarre), 厌烦 (annoying)

target translation terms for the slang word "fruitcake" (an insane person):

Input sentence: *Oh man, you don't want to date that girl. She's always drunk and yelling. She is a total* ***fruitcake****.*[1]

Google Translation: 哦，男人，你不想约会那个女孩。她总是喝醉了，大喊大叫。她是一个**水果蛋糕**。

Another lines of baseline methods is scoring-based. The basic idea is to score all words in our bilingual lexicon and consider the top K words as the target terms. Given a source term to be translated, the Linear Transform (LT), MCCA, MCluster and Duong methods score the candidate target terms by computing cosine similarities in their constructed bilingual vector space (with the tuned best settings in previous evaluation). A more sophisticated baseline (TransBL) leverages the bilingual lexicon: for each candidate target term w in the target language, we first obtain its translations T_w back into the source language and then calculate the average word similarities between the source term and the translations T_w as w's score.

Our *SocVec-based method* (**SV**) is also scoring-based. It simply calculates the cosine similarities between the source term and each candidate target term within *SocVec* space as their

[1] http://www.englishbaby.com/lessons/4349/slang/fruitcake

Table 6–7 Slang-to-Slang Translation Examples

Chinese Slang	English Slang	Explanation
萌	adorbz, adorb, adorbs, tweeny, attractiveee	cute, adorable
二百五	shithead, stupidit, douchbag	A foolish person
鸭梨	antsy, stressy, fidgety, grouchy, badmood	stress, pressure, burden

scores.

6.5.3 Experimental Results

To quantitatively evaluate our methods, we need to measure similarities between a produced word set and the ground truth set. Exact-matching Jaccard similarity is too strict to capture valuable relatedness between two word sets. We argue that average cosine similarity (ACS) between two sets of word vectors is a better metric for evaluating the similarity between two word sets.

$$\mathrm{ACS}(A,B) = \frac{1}{|A||B|}\sum_{i=1}^{|A|}\sum_{j=1}^{|B|}\frac{\mathbf{A_i}\cdot\mathbf{B_j}}{||\mathbf{A_i}||||\mathbf{B_j}||}$$

The above equation illustrates such computation, where A and B are the two word sets: A is the truth set and B is a similar list produced by each method. In the previous case of ‘‘二百五” (Section 6.5.1), A is {foolish, stubborn, rude, impetuous} while B can be {imbecile, brainless, scumbag, imposter}. $\mathbf{A_i}$ and $\mathbf{B_j}$ denote the word vector of the i^{th} word in A and j^{th} word in B respectively. The embeddings used in ACS computations are pre-trained *GloVe* word vectors[1] and thus the computation is fair among different methods.

Experimental results of Chinese and English slang translation in terms of the sum of *ACS* over 200 terms are shown in Table 6–5. The performance of online translators for slang typically depends on human-set rules and supervised learning on well-annotated parallel corpora, which are rare and costly, especially for social media where slang emerges the most. This is probably the reason why they do not perform well. The Linear Transformation (LT) model is trained

[1] `https://nlp.stanford.edu/projects/glove/`

on highly confident translation pairs in the bilingual lexicon, which lacks OOV slang terms and social contexts around them. The TransBL method is competitive because its similarity computations are within monolingual semantic spaces and it makes great use of the bilingual lexicon, but it loses the information from the related words that are not in the bilingual lexicon. Our method (SV) outperforms baselines by directly using the distances in the *SocVec* space, which proves that the *SocVec* well captures the cross-cultural similarities between terms.

To qualitatively evaluate our model, in Table 6–6, we present several examples of our translations for Chinese and English slang terms as well as their explanations from the glossary. Our results are highly correlated with these explanations and capture their significant semantics, whereas most online translators just offer literal translations, even within obviously slang contexts. We take a step further to directly translate Chinese slang terms to English slang terms by filtering out ordinary (non-slang) words in the original target term lists, with examples shown in Table 6–7.

6.6 Related Work

Although social media messages have been essential resources for research in computational social science, most works based on them only focus on a single culture and language[150–155]. Cross-cultural studies have been conducted on the basis of a questionnaire-based approach for many years. There are only a few of such studies using NLP techniques.

[156] present a framework to visualize the cross-cultural differences in concerns in multilingual blogs collected with a topic keyword.[132] show that cross-cultural analysis through language in social media data is effective, especially using emotion terms as culture features, but the work is restricted in monolingual analysis and a single domain (love and relationship).[133] investigate the cross-cultural differences in word usages between Australian and American English through their proposed “socio-linguistic features” (similar to our social words) in a supervised way. With the data of social network structures and user interactions,[157] study how to quantify the controversy of topics within a culture and language.[158] propose an approach to detect differences of word usage in the cross-lingual topics of multilingual topic modeling results. To the best of our knowledge, our work for Task 1 is among the first to mine and quantify the cross-cultural differences in concerns about named entities across different languages.

Existing research on slang mainly focuses on automatic discovering of slang terms[159] and normalization of noisy texts[135] as well as slang formation.[160] are among the first to propose an automatic supervised framework to mono-lingually explain slang terms using external resources.

However, research on automatic translation or cross-lingually explanation for slang terms is missing from the literature. Our work in Task 2 fills the gap by computing cross-cultural similarities with our bilingual word representations (*SocVec*) in an unsupervised way. We believe this application is useful in machine translation for social media[161].

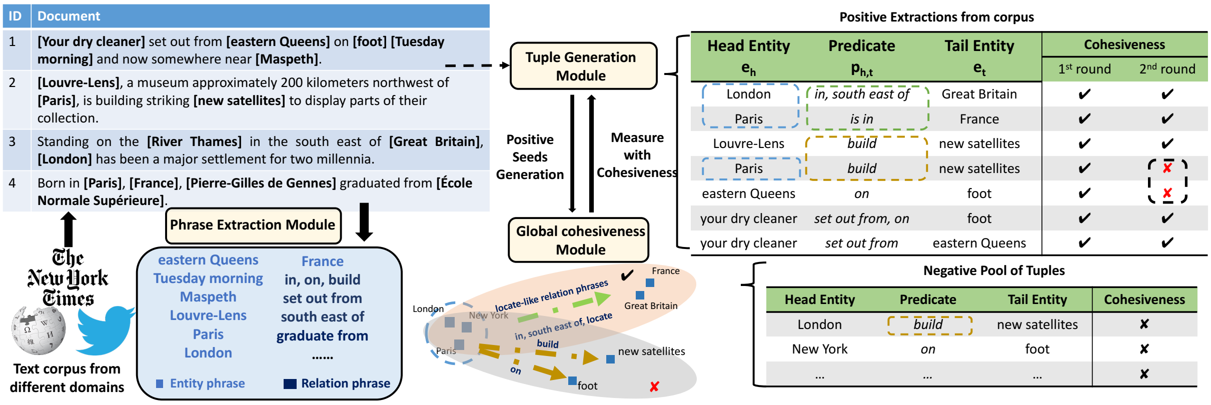
Many existing cross-lingual word embedding models rely on expensive parallel corpora with word or sentence alignments[127, 162]. These works often aim to improve the performance on monolingual tasks and cross-lingual model transfer for document classification, which does not require cross-cultural signals. We position our work in a broader context of "monolingual mapping" based cross-lingual word embedding models in the survey of[124]. The SocVec uses only lexicon resource and maps monolingual vector spaces into a common high-dimensional third space by incorporating social words as pivot, where orthogonality is approximated by setting clear meaning to each dimension of the *SocVec* space.

Chapter 7 Global Structure Cohesiveness for Open Information Extraction

7.1 Introduction

Massive text corpora are emerging worldwide in different domains and languages. The sheer size of such unstructured data and the rapid growth of new data pose grand challenges on making sense of these massive corpora. Information extraction (IE)[1] – extraction of relation tuples in the form of (*head entity*, relation, *tail entity*) – is a key step towards automating knowledge acquisition from text. In Fig. 7–1, for example, the relation tuple (*Louvre-Lens*, build, *new satellites*) can be extracted from sentence S_2 to represent a piece of factual knowledge in text with structured form. Relation tuples so extracted have a variety of downstream applications, including serving as building blocks for knowledge base construction[2] and facilitating question answering systems[3, 4]. While traditional IE systems require people to pre-specify the set of relations of interests, recent studies on *open-domain information extraction* (Open IE)[13–15] rely on *relation phrases* extracted from text to represent the entity relationship, making it possible to adapt to various domains (*i.e.*, open-domain) and different languages (*i.e.*, language-independent).

Prior work on Open IE can be summarized as sharing two common characteristics: (1) conducting extraction based on local context information; and (2) adopting a pre-trained incremental system pipeline, which suffers from domain-shift problem. Current Open IE systems focus on analyzing the local context within individual sentences to extract entity and their relationships, while ignoring the redundant information that can be collectively referenced across different sentences and documents in the corpus. For example, in Fig. 7–1, seeing entity phrases “*London*" and “*Paris*" frequently co-occur with similar relation phrase and tail entities in the corpus, one gets to know that they have close semantics (same for “*Great Britain*" and “*France*"). On one hand, this helps confirm that (*Paris*, is in, *France*) is a quality tuple if knowing (*London*, is in , *Great Britain*) is a good tuple. On the other, this helps rule out the tuple (*Paris*, build, *new satellites*) as “*Louvre-Lens*" is semantically distant from “*Paris*". Therefore, the rich information redundancy in the massive corpus motivates us to design an effective way of measuring whether a candidate relation tuple is consistently used across various context in the corpus (*i.e.*,



ID	Document
1	**[Your dry cleaner]** set out from **[eastern Queens]** on **[foot]** **[Tuesday morning]** and now somewhere near **[Maspeth]**.
2	**[Louvre-Lens]**, a museum approximately 200 kilometers northwest of **[Paris]**, is building striking **[new satellites]** to display parts of their collection.
3	Standing on the **[River Thames]** in the south east of **[Great Britain]**, **[London]** has been a major settlement for two millennia.
4	Born in **[Paris]**, **[France]**, **[Pierre-Gilles de Gennes]** graduated from **[École Normale Supérieure]**.

Positive Extractions from corpus

Head Entity e_h	Predicate $p_{h,t}$	Tail Entity e_t	Cohesiveness 1st round	Cohesiveness 2nd round
London	*in, south east of*	Great Britain	✔	✔
Paris	*is in*	France	✔	✔
Louvre-Lens	*build*	new satellites	✔	✔
Paris	*build*	new satellites	✔	✘
eastern Queens	*on*	foot	✔	✘
your dry cleaner	*set out from, on*	foot	✔	✔
your dry cleaner	*set out from*	eastern Queens	✔	✔

Negative Pool of Tuples

Head Entity	Predicate	Tail Entity	Cohesiveness
London	*build*	new satellites	✘
New York	*on*	foot	✘
...	...	...	✘

Figure 7–1 Overview of the ReMine Framework.

global cohesiveness). To overcome the second issue, we propose a domain-specific framework, called ReMine , to unify two important yet *complementary* signals on target corpus, *i.e.*, the local context information and the global cohesiveness (see also Fig. 7–1). Specifically cohesive semantics is measured by low-dimensional embeddings of entity and relation phrases, where two entity phrases are similar if they share similar relation phrases and entity arguments. The entity and relation embeddings so learned can be used to measure the cohesiveness score of a candidate relation tuple. ReMine *jointly* optimizes both the *extraction of entity and relation phrases* and the *global cohesiveness across the corpus*, each being formalized as an objective function so as to quantify the quality scores. It achieves competitive performance in Open IE task(see Sec. 7.3.2) and shows its extraction clearness in Sec. 7.4. We demonstrate that an End-to-End solution with global information for OpenIE task is a promising direction.

7.2 The ReMine Framework

In general, Open IE systems first identify entity phrases $\mathcal{E}$, relation phrases $\mathcal{R}$, then select and pair up entity phrases and further extract meaningful relation tuples $(e_h, e_t, p_{h,t})$ among them. Formally, we define the task of Open IE as follows.[1].

Given a corpus $\mathcal{D}$, the task of Open IE aims to extract entity phrases $\mathcal{E}$, relation phrases $\mathcal{R}$ and relation tuples $\{e_h, e_t, p\}_{k=1}^{N_t}$, where entity argument pairs $(e_h,\ e_t)$ extracted from one sentence are distinctive to each other.

There are three challenges. First, true label of phrases in target domain is unknown and thus asks for effective measuring of the phrase quality. Second, there exist multiple entity

[1] All the notations used can be found in Table 7–1

Table 7–1 Notation Table

b_i	the start index of the i-th phrase in the sentense
s_i	the word sequence of i-th phrase segment $[b_t, b_{t+1})$
t_i	type of the i-th phrase, $t_i \in \{ent, rel, background\}$
f_i	text feature of phrase segment s_i
w_i	word sequence probability given b_i and b_{i+1}
e_h	head entity in a relation tuple, where $e_h \in \mathcal{E}$
e_t	tail entity in a relation tuple, where $e_t \in \mathcal{E}$
r	relation phrase between any $(e_h, e_t) \in E_p^+$
$p_{h,t}$	predicate between head and tail entity, $p = (r_1, ...r_n)$
v_i	embedding for entity/relation phrase s_i
σ	cohesiveness measure of relation tuple $(e_h, p_{h,t}, e_t)$

phrases in one sentence. Therefore, selecting correct head and tail entity can be problematic. Third, ranking extracted tuples without referring to the entire corpus may favor with good local structures.

7.2.1 Domain-specific phrase extraction

We address entity and relation phrase extraction as a multiple type phrasal segmentation task, traditional Open IE uses NP-chunking to extract entity phrases, yet not all noun phrases can carry rich information and it requires additional training. Given word sequence C and corresponding linguistic features $\mathcal{F}$ in Table 7–3, a segmentation $\S = s_1, s_2, ..., s_n$ is separated by boundary index $B = b_1, b_2, ..., b_{n+1}$. For each segment s_i, there is a type indicator t_i, indicating the most possible type of s_i, the joint probability is factorized as:

$$P(C, \mathcal{F}) = \prod_{i=1}^{n} P(b_{i+1}, s_i | b_i, \mathcal{F}) \tag{7–1}$$

ReMine generates each segment as follows,

1. Given the start index b_i, generate the end index b_{i+1} according to context-free prior Δ, *i.e.*dependency tree pattern prior.

2. Given the start and end index (b_i, b_{i+1}) of segment s_i, generate a word sequence s_i according

Table 7–2 Entity and relation phrase candidates generation with regular expression patterns on part-of-speech tag

Pattern	Examples
Entity Phrase Patterns	
`<DT|PP$>?<JJ>*<NN>+`	the state health department
`<NNP>+<IN>?<NNP>+`	Gov. Tim Pawlenty of Minnesota
Relation Phrase Patterns	
`{V=<VB|VB*>+}`	furnish, work, leave
`{V}{P=<NN|JJ|RP|PRP|DT>}`	provided by, retire from
`{V}{W=<IN|RP>?*}{P}`	die peacefully at home in

to a multinomial distribution over all segments at the same length.

$$P(s_i|b_i, b_{i+1}) = P(s_i|b_{i+1} - b_i) \tag{7–2}$$

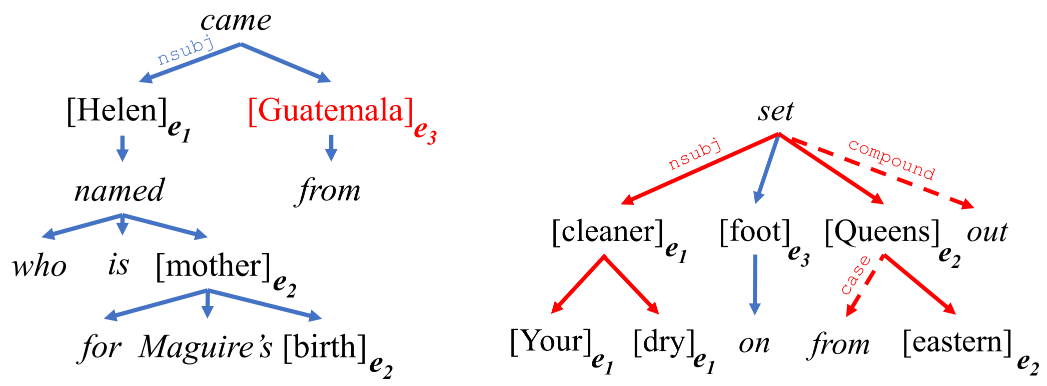
3. Finally, we generate a phrase type t_i indicating that phrase s_i most likely belongs to and a quality score showing how it likely to be a good phrase $\lceil s \rfloor$.

$$P(\lceil s_i \rfloor | s_i) = \max_{t_i} P(t_i|s_i) \tag{7–3}$$

Candidate Generation. Phrase Mining[49] had made an assumption that quality phrases are frequent n-grams in corpus, while it is not the case when sentence-level extractions are important. To overcome phrase sparsity, several NP chunking rules[163], see Table 7–2, are adopted to discover infrequent but informative phrase candidates. In our experiments, frequent n-grams and NP chunking rules contribute comparable amount of phrase candidates.

We denote unique phrase as u among all word sequence s_i and $P(s_i|b_{i+1} - b_i)$ as θ_u and $\max_{t_i} P(t_i|s_i)$ is determined by random forest classifiers robust positive-only distant training[49]. Similar with[164], we use Viterbi Training[165] to update Segmentation S and parameters θ, Δ iteratively. In the E-step, given θ and Δ, dynamic programming is used to find the optimized segmentation. In the M-step, we first fixed parameter θ, and update context-dependent prior δ. Next when Δ is fixed, optimized solution of θ_u is:

$$\theta_u = \frac{\sum_{i=1}^{m} \mathbf{1} \cdot (s_i = u)}{\sum_{i=1}^{m} \mathbf{1} \cdot (b_{i+1} - b_i = |u|)} \tag{7–4}$$



(a) Selecting among candidate subject entities (b) Finding shortest dependency path

Figure 7–2 Dependency parsing tree of example sentence in Fig. 7–1, *"Your dry cleaner set out from eastern Queens on foot Tuesday morning."* Segmented entities are marked as "[entity_token]$_{e_i}$"

7.2.2 Relational Extraction in Local Context

Leveraging information along the dependency path between two given entities has been proved useful for open information extraction[166, 167], as it reduces noise by removing irrelevant semantic phrases or clauses in long sentences with multiple entities. Instead of words, we treat phrases as atoms, each predicate $p_{h,t}$ is composed of one or more relation phrases r. We now present how we generate valuable relation tuples based on those phrases along dependency path, *i.e.*

$$\begin{aligned}\mathbf{P}(r, e_h, e_t) &= \prod_{i=1}^{n} \mathbf{P}(r_i|s_i, e_h, e_t)\mathbf{P}(s_i|b_i, b_{i+1}) \\ \max_{p_{h,t}} \mathbf{P}(r, e_h, e_t) &\Rightarrow \sum_{i\in p_{h,t}} \log\sigma(r_i, e_h, e_t) + \log w_i\end{aligned} \tag{7–5}$$

where $b_1, b_2, ..., b_{n+1}$ are boundary index along dependency path of entity argument pair(e_h, e_t). $\mathbf{P}(s_i|b_i, b_i + 1)$ is inherited from phrase extraction module as word sequence probability w_i, then ReMine judges wether it is a good relation between entity e_h and entity e_t. We will introduce how to obtain positive entity pairs (e_h, e_t) and similarity measure σ in next section. Notice relation phrase boundary $i \in p_{h,t}$ in equation 7–5 can be derived via dynamic programming since w_i and σ is known for every possible segmentation.

Positive Entity Pairs Initialization. For a given sentence s, after phrase segmentation, we have entity arguments $e_1, e_2, ..., e_n$ and relation arguments $r_1, r_2, ..., r_n$. Notice that good background phrases also recognized as arguments. However, it's infeasible to every entity pair and a large portion of tuples are incorrect among $N(N-1)$ pairs. Positive entity pairs E_p^+ are entity arguments pair selected. Here we heuristically initialize E_p^{+0} by attaching *nearest* subject e_i to object e_j and make an approximation that each entity argument phrase can only be object once, which also guarantees entity pairs to be distinctive. Nearest subject of e_j is defined as entity e_i that has the shortest dependency path length to e_j among all other entities. Considering Fig. 7–2a, we would like to find subject of entity e_3 : *Guatemala*, length of the shortest path between e_3 and e_1, e_2 are 2,4 respectively. For those entity candidates with the same distance, see Fig. 7–2b, both e_1: *Your dry cleaner* and e_2: *eastern Queens* are one hop away from e_2: *foot*. We will prefer subject with "nsubj" type *i.e.*e_1 then choose closest entity in original sentence if there are still multiple of them.

7.2.3 Global Measuring of Cohesiveness

In our illustrative example in Fig. 7–1, current methods use textual patterns[15, 168] to identify (Paris, build, new satellites) as a false extraction, while we prune it via global cohesiveness measure σ. To capture the global cohesiveness of relation tuples, we adopt translating measuring of knowledge base completion[169].

$$\sigma(p_{h,t}, h, t) = \|v_h + v_p - v_t\|; v \in \mathbb{R} \tag{7–6}$$

Such objective associates entity and relation with dense feature vectors, where v_h, v_t are embeddings for head and tail entities, p is the predicate. We use L_1 norm in ReMine for efficiency.

Based on initial positive entity pairs constructed E_p^{+0} and relation tuples, we construct a pseudo knowledge graph. Particularly, predicate $p_{h,t} = (r_1, r_2, ..., r_n)$ may contain several relation phrases. We model multiple relation phrases in one predicate as process of knowledge traverse[170] *i.e.*$v_p = \sum_{i=1}^{n} v_{r_i}/n$.

In order to learn global cohesiveness representation $\mathcal{V}$. We construct correlated negative tuples from positive seeds *i.e.*current relation tuples $\mathcal{T}$ accordingly, see Fig. 7–1. False tuples like (Paris, build, new satellites) will be reduced by some similar negative tuples like (Paris, build, River Thames). Cohesiveness measure S is optimized as follows,

$$\max \sum_{p,h,t}^{\mathcal{T}} \sum_{p,h',t'}^{\mathcal{T}^-} \sigma(p, h, t) - \sigma(p, h', t') - \gamma \tag{7–7}$$

where $\mathcal{T}$ denote positive relation tuples generated by local relational extraction, γ is the hyper margin, $(p, h', t') \in \mathcal{T}^-$ is composed of training tuples with either **h** or **t** replaced.

7.2.4 The Joint Optimization Problem

We now show how local context and global cohesiveness above can be organically integrated. Relation tuple generation in sec 7.2.2 incoporates cohesiveness similarity σ and cohesivess measure learning depends on target domain tuples $\mathcal{T}$.

Update Positive Pairs. Given semantic representation for each entity **e** and relation **r** and local segmentation between entity pairs, we can update the *Positive Entity Pairs* by finding most semantically consistent subject e_h for each object $\mathbf{e}_t$. By optimizing $\mathbf{P}(r, e_h, e_t)$ in Eq. 7–5, we also obtain the relation tuples for updated positive pairs E_p^{+n+1}.

$$E_p^+ = \underset{e_h}{\operatorname{argmax}}\,\mathbf{P}(p_{h,t}, e_h, e_t) \tag{7–8}$$

Overall Updating Schema. From an overall point of view, the final objective for update is formulated as the sum of both sub-objectives,

$$O = O_{local} + O_{global} \tag{7–9}$$

where $O_{local} = \max \sum_{E_p^+} \mathbf{P}(p_{h,t}, e_h, e_t)$, $O_{global} = \max \sum_{p,h,t}^{\mathcal{T}} \sum_{p,h',t'}^{\mathcal{T}^-} \sigma(p,h,t) - \sigma(p,h',t') - \gamma$. To maximize above unified open IE objective, see Alg. 7–1, we first initialize positive entity pairs E_p^{+0}. Given entity argument pairs, we perform local optimization, which leads to positive relation tuples $\mathcal{T}$. Note that, at the first round, there is no global representation, so we initialize all $\sigma = 1$ as identical. Then we update global phrase semantic representation via stochastic gradient descent. With both global cohesiveness information and local segmentation result, ReMine updates positive pairs as described in Sec. 7.2.2. Overall ReMine solves the integrated problem in a greedy manner, it iteratively updates local and global objectives until a stable σ and E_p^+ is reached.

7.3 Experiments

In this section, we evaluate the performance of the proposed system on Open Information Extraction. By designing experiment datasets and comparing the output of our system with state-of-the-art Open IE systems to examine our claim: (1) a domain-specific and end-to-end mannered pipeline performs consistently well on different domains; and (2) global structure cohesiveness improves Open Information Extraction.

Algorithm 7–1 Joint Tuple Mining

```
Input: Corpus $\mathcal{D}$, Sentence S, Entities $\mathcal{E}$, Relations $\mathcal{R}$, Word sequence probability$\mathcal{W}$
Output: Relation Tuples $\mathcal{T}$, representation $\mathcal{V}$, similarity measure $\sigma$
  initialize positive $E_p^+$ according Sec. 7.2.2
  initialize similarity measure $\sigma = 1$
  while $E_p^+$ is not stable do
      for each entity argument pair $(e_h, e_t)$ do
          identify semantic path $P(e_h, e_t)$
          $p_{h,t} \Leftarrow$ from relation tuple generation module given $\mathcal{W}$ and $\sigma$
          update relation tuple $(e_h, p_{h,t}, e_t) \in \mathcal{T}$
      end for
      $\mathcal{V}, \sigma \Leftarrow$ update global cohesiveness module
      update $E_p^+$ according to $\mathcal{E}$, $\mathcal{R}$ and $\sigma$
  end while
```

Table 7–3 List of features used in the phrase extraction module (Sec. 7.2.1).

Feature	Descriptions	Example
popularity	raw frequency, occurrence probability	
completeness	sub-phrases within long frequent phrases are also informative	*"relational database system" meets the criteria*
concordance	tokens in quality phrases should co-occurs frequently	*"strong tea" versus "heavy tea"*
punctuations	phrase in parenthesis, quote or has dash after	*(12.pm), "the Zeitlin sidewinder"*
stopwords	first/last token is stopword and stopword ratio	*the, their, therefore*
word shape	first capitalized or all capitalized	*NBA, Defense Secretary Donald H. Rumsfeld*
part-of-speech tags	unigram and bigram POS tags	*the Rev. Ian Paisley: DT,NNP,DT-NNP*

7.3.1 Experimental Setup

Datasets. We use two datasets in our experiments: (1) NYT[171]: The training corpus consists of 23.6k sentences from ~294k 1987-2007 New York Times news articles. 395 sentences are manually annotated with entity and relation mentions by authors[171]. (2) Twitter[172]: The dataset consists of 1.4 million tweets in Los Angeles collected from 2014.08.01 to 2014.11.30.

Distantly Supervised Phrase Seeds. Our proposed method ReMine mainly have several outcomes, including high-quality entity and relation phrases and relation tuples. For each corpus, we first generate some distant supervision seeds via DBpedia Spotlight service[1][173] for entity phrases. With entity phrases, we generate relation phrases between each pair of entity mentions via pattern matching. We then followed the procedure introduced in Sec. 7.2.1, segmenting input corpora into entity phrases, relation phrases, and background phrases.

Phrase Features Generation. To estimate type and quality in step 4 of Sec. 7.2.1, we designed a set of features $\mathcal{F}$ in Table 7–3 that indicates a good phrase and its type. It can be grouped into several different categories, *i.e.*statistic features, token-wise features and POS features. ReMine treats phrases with multiple POS tag sequences into different patterns. For example, "work NN" and "work VBP" are two different semantic patterns. Shang et al.[49] show that considering POS tags in quality predictor yields better performance.

Compared Methods. We consider following approaches for comparison:
(1) OLLIE[15] utilizes open pattern learning and extracts patterns over dependency path and part-of-speech tags. (2) ClausIE[168] adopts clause patterns to handle long-distance relationships. (3) Stanford OpenIE[174] learns a clause splitter via distant training data. (4) MinIE[175] refines tuple extracted by ClausIE by identifying and removing parts that are considered overly specific. (5) ReMine-L is a variant of our approach with only local tuple generation. (6) ReMine-G extend ReMine-L by ranking tuples via global cohesiveness without any further iterations. (7) ReMine is our proposed approach, in which relation tuple generation module collaborates with global cohesiveness module. All Open IE methods, to some extent, requires weak supervision or distant supervision.

Evaluation Setup. We aim to compare performance in both normal and short text, so we choose NYT and Twitter dataset for evaluation. For the relation tuple extraction task, since each tuple obtained by ReMine and other benchmark methods will also be assigned a confidence score. We rank all the tuples according to their confidence scores. Based on the ranking list, we use the following four measures: $P@k$ is the precision at rank k. MAP is mean average precision of the entire ranking list. $NDCG@k$ is the normalized discounted cumulative gain at rank k. Note that we do not use recall in this task because it is infeasible to know all the "correct" tuples.

Annotations Setup. We manually labeled the extractions got from these extractors. Each extraction was labeled by two independent annotators for two rounds. Both annotators are highly proficient and literate in English. The two annotators are asked to evaluate without

[1] https://github.com/dbpedia-spotlight/dbpedia-spotlight

Table 7–4 Performance comparison with state-of-the-art Open IE systems on two datasets from different domains, using Precision@K, Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (NDCG).

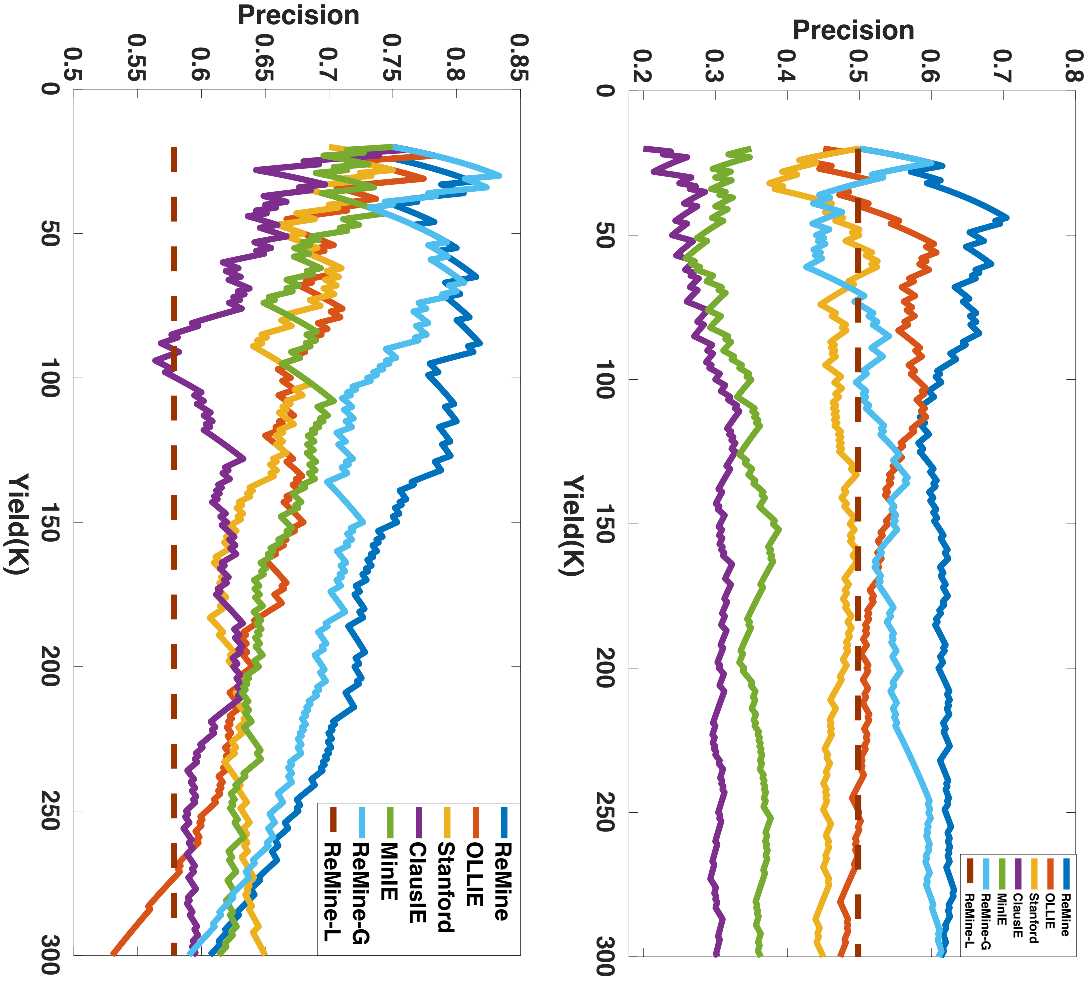
Methods	**NYT**[171]					**Twitter**[172]				
	P@100	P@200	MAP	NDCG@100	NDCG@200	P@100	P@200	MAP	NDCG@100	NDCG@200
ClausIE	0.580	0.625	0.623	0.575	0.667	0.300	0.305	0.308	0.332	0.545
Stanford	0.680	0.625	0.665	0.689	0.654	0.390	0.410	0.415	0.413	0.557
OLLIE	0.670	0.640	0.683	0.684	0.775	0.580	0.510	0.525	0.519	0.626
MinIE	0.680	0.645	0.687	0.724	0.723	0.350	0.340	0.361	0.362	0.541
ReMine-L	0.578	0.578	0.585	0.578	0.631	0.498	0.499	0.506	0.500	0.533
ReMine-G	0.730	0.695	0.734	0.751	0.783	0.510	0.580	0.561	0.522	0.610
ReMine	**0.780**	**0.720**	**0.760**	**0.787**	**0.791**	**0.610**	**0.610**	**0.627**	**0.615**	**0.651**

knowing which model produced the results, eliminating potential bias in evaluation. One extraction is treated as correct only if both labelers think it is correct. Similar to the settings in previous studies[168], we ignore the context of the extracted tuples during labeling. For example, both (*“we", “hate", “it"*) and (*“he", “has", “father"*) will be treated as correct as long as they meet the fact described in the sentence. However, tuples cannot read smoothly will be labeled as incorrect propositions. For example, (*“he", “is", “is the professor"*) and (*“he", “is", “the professor and"*) will not be counted since they have mistakes in the word segmentation level. We measured the agreement between the two labelers using Cohen’s Kappa value. The scores are 0.79 and 0.73 for the NYT dataset and the Twitter dataset respectively.

7.3.2 Experiments and Performance Study

Open IE systems can extract information tuples from open domain corpus. We compared ReMine with its own ablations ReMine-L and ReMine-G as well as four other Open IE systems mentioned above.

Sometimes existing systems unintentionally paraphrase extractions as arguments may have overlapped boundaries. An extreme case, imagine there are two systems, one reports N correct tuples and the other with $2N$ paraphrased. Since $P@k$ curves are usually monotone decreasing, we will favor system generating $2N$ tuples. Fixed phrase boundaries prevent ReMine from generating redundant facts from the corpus, a detailed study can be found in case study. 7.4. Among all the Open IE system described above, ReMine and OLLIE extract a relatively small number of tuples. For example, for the first 100 sentences in the NYT test set, both ReMine and OLLIE get about 300 tuples. In contrast, Stanford OpenIE returns more than 1,000 tuples. To alleviate the “unintentional paraphrasing” issue, since each extracted tuple is also assigned a confidence score, we select 300 tuples for both datasets with the highest scores for each Open



(a) NYT

(b) Twitter

Figure 7–3 The Precision@K curves of different Open IE systems on NYT and Twitter datasets.

IE system to plot the curves. By selecting 100 sentences from NYT test set and 300 tweets from Twitter test set, we believe ~3 tuples per sentence in News domain and ~1 tuple per sentence in Twitter seems to be reasonable. The results are shown in Figure 7–3 and Table 7–4.

‘‘Does ReMine performs consistently well on different domains?”

According to the curves in Figure 7–3a and 7–3b, ReMine achieves the best performance among all Open IE systems. All methods experience performance drop in Twitter, while ReMine declines less than any other methods on the rank-based measures. In the NYT dataset, all the systems except OLLIE have similar overall precision (*i.e.P*@300). But ReMine has a “higher” curve since most tuples obtained by Stanford OpenIE and ClausIE will be assigned score 1. Therefore we may not rank them in a very rational way. In contrast, the scores of different tuples obtained by ReMine-G and ReMine are usually distinct from each other. In Table 7–4, ReMine also consistently performs the best . In the Twitter dataset, ReMine shows its power in dealing with short and noisy text. Both ClausIE and MinIE have a rather low score since there are lots of non-standard language usages and grammatic errors in tweets. Dependency parsing attached more wrong arguments and labels. All methods investigated depends on dependency parsing in varying degrees, while clause-based methods rely heavily on it and may not achieve a satisfying performance.

Does global cohesiveness improve quality of Open IE? Model-wise we believe global cohesiveness helps Open IE from two aspects: (1) rank tuples (2) update entity argument pairs. From Figure 7–3 and Table 7–4, We found ReMine outperforms ReMine-G and ReMine-L on each evaluation metric on both datasets. In particular, ReMine-G differs from ReMine-L only on extraction scores, and global cohesiveness σ provide better ranking performance(P@300) than random. The gain between ReMine and ReMine-G also clearly shows the updated entity pairs and extractions have better quality in general. In the twitter dataset, a larger performance gap proves that global cohesiveness is more robust to low quality and short text compared with pattern and clause.

7.4 Case Study

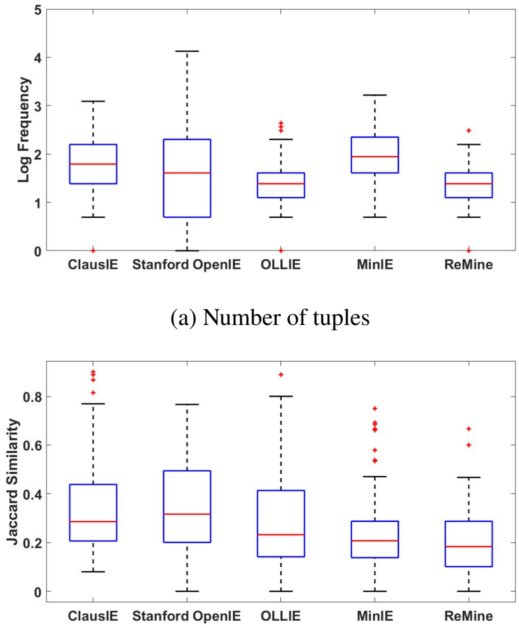
Our studies reveal overall quality of extractions compared with other Open IE systems and effectiveness of specific component.

Clearness and correctness on extractions. In Table. 7–5, we show the extraction samples of the NYT sentence *"Gov. Tim Pawlenty of Minnesota ordered the state health department this month to monitor day-to-day operations at the Minneapolis Veterans Home after state inspectors found that three men had died there in the previous month because of neglect or medical errors."*. We could see that all the extractors share consensus on that *"Gov. Tim Pawlenty of Minnesota ordered the state health department"* (R_2, R_3, R_7, R_{11} and R_{13}). But some other actions do not belong to "Tim Pawlenty". Both Stanford OpenIE and OLLIE make mistakes on that (R_4 and R_9). In contrast, ClausIE has no logic mistakes in the samples. However, the objective component of R_1 is too complicated to illustrate one proposition clearly. As we mentioned above, this kind of tuples will be labeled as incorrect ones. R_{15} is the only correct tuple to identify the location *"Minneapolis Veterans Home"*, and ReMine also carefully selects the words to form the predicate *"order_to_monitor_at"* to prevent excessively long relation phrase.

Table 7–5 Extraction samples of one sentence in the NYT dataset using different methods. “T” means correct tuples and “F” means incorrect ones. *The tuple is too complicated to clearly explain one proposition. $^\#$The tuple cannot read smoothly. †The tuple is logically wrong.

	ClausIE	
R_1	*(“Gov. Tim Pawlenty of Minnesota", “ordered", “the state health department this month to monitor day-to-day operations after state inspectors found that three men had died there in the previous month because of neglect or medical errors")*	F*
R_2	*(“Gov. Tim Pawlenty of Minnesota", “ordered", “the state health department this month to monitor day-to-day operations")*	T
	Stanford OpenIE	
R_3	*(“Gov. Tim Pawlenty", “ordered", “state health department")*	T
R_4	*(“Gov. Tim Pawlenty", “monitor", “operations")*	F†
R_5	*(“three men", “died there because of", “neglect")*	T
R_6	*(“men", “died in", “month")*	F$^\#$
	OLLIE	
R_7	*(“Gov. Tim Pawlenty of Minnesota", “ordered the state health department in", “this month")*	T
R_8	*(“three men", “had died there in", “the previous month")*	T
R_9	*(“Gov. Tim Pawlenty of Minnesota", “had died because of", “neglect errors")*	F†
	MinIE	
R_{10}	*(“Tim Pawlenty", “is", “Gov.")*	T
R_{11}	*(“Tim Pawlenty of Minnesota", “ordered state health department", "this month")*	T
R_{12}	*(“QUANT_S_1 men", “had died because of", “neglect errors")*	F†
	ReMine	
R_{13}	*(“Gov. Tim Pawlenty of Minnesota", “order", “the state health department")*	T
R_{14}	*(“Gov. Tim Pawlenty of Minnesota", “order_to_monitor", “day-to-day operation")*	T
R_{15}	*(“Gov. Tim Pawlenty of Minnesota", “order_to_monitor_at", “Minneapolis Veterans Home")*	T
R_{16}	*(“three man", “have_die_there", “medical error")*	F$^\#$

Distinctiveness of extractions. In our formulation, we try to cover every entity detected in the target sentence while avoid extracting duplicate tuples. In Fig. 7–4a, we show the distribution of the number of extractions obtained by each Open IE system on the first 100 sentences in NYT dataset. We can see that OLLIE’s and ReMine ’s distributions are relatively balanced. In contrast, Stanford OpenIE returns extractions with a large variance. Among 1054 tuples it extracted, there are 228 tuples belong to a single sentence and 157 belong to another. In fact, the latter sentence has only 39 words. This reminds us that the number of extractions may not be a good alternative of “recall”. A more direct way to examine distinctiveness of our extractions is



(a) Number of tuples

(b) Jaccard similarity

Figure 7–4 Distribution over number of extractions and distinctiveness of extractions for different Open IE systems.

calculating average Jaccard similarity between extractions from same sentence. We present the Jaccard similarity distribution of different systems at Fig. 7–4b, we can clearly see MinIE and ReMine extracts most distinctive facts as they both consider not to be overly specific.

Effectiveness of global evidence. Corpus-level cohesiveness can help reduce local error while generating relation tuples. Especially on twitter set, local linguistic structure fails to attach argument correctly at the first place whereas global cohesiveness module corrects those extractions. In table 7–6, considering sentence *"Dudamel conduct his score from Liberador#BeastMode @Hollywood Bowl"* ReMine rejects entity pair (*Liberador*, *Hollywood*) which is not compatible with the predicate "@". This is because in the twitter corpus, it is more common to see *Person @ Place*. Therefore ReMine attaches Hollywood to Dudamel after updating entity pairs.

7.5 Related Work

Information Extraction. Open domain information extraction has been extensively studied in literature. Most of the existing work follows two lines of work, that is, pattern based methods or clause based methods. Pattern based information extraction can be as early as Hearst

Table 7–6 Different entity pairs discovered by ReMine and ReMine-G , where blue ones are incorrect extractions.

ReMine-G	ReMine
(Dudamel; “conduct”; Liberador)	*(Dudamel; “conduct”; Liberador)*
(Dudamel; “conduct...from”; #BeastMode)	*(Dudamel; “conduct... @”; Hollywood Bowl)*
(Liberador, “@”, Hollywood Bowl)	

patterns like “NP_0 such as $\{NP_1, NP_2, \ldots\}$” for hyponymy relation extraction[176]. Carlson and Mitchell *et al.* introduced Never-Ending Language Learning (NELL) based on free-text predicate patterns[177, 178]. ReVerb[163] identified relational phrases via part-of-speech-based regular expressions. Besides part-of-speech tags, recent works start to use more linguistic features, like dependency parsing, to induct long distance relationships[15, 179]. Similarly, ClausIE[168] inducted short but coherent pieces of information along dependency paths, which is typically subject, predicate and optional object with complement. Angeli *et al.* adopted a clause splitter using distant training and mapped predicates to a known relation schema statistically[174]. MinIE[175] further improves the clearness of relation tuples by introducing different statistical measures like polarity, modality, attribution, and quantities. Compared with these works, this paper differs in several aspects: (1) previous works rely on external tools for phrase extraction, which may suffer from domain-shift and sparsity problem, while we provide an End-to-End solution towards Open IE on target domain. (2) The correctness of extracted facts is measured via global cohesiveness instead of using local context alone.

Knowledge Base Population. Knowledge bases (KBs), such as DBpedia[180] and Freebase[181], extract tuples from World Wide Web. However, they are all built upon existing and specific relation types. Knowledge base population or completion aims at predicting whether tuples not in knowledge base are likely to be true or not. Embedding models[169] has been widely used to learn semantic representation for both entities and relations. Recent research[170, 182] shows that relation path is traversable and contains richer information. People also try to construct web-scale knowledge base using statistical learning and pre-defined rules and predicates[183]. All these approaches start with clean knowledge base tuples, our proposed start from noisy extractions but share similar semantic measures as them. In other words, we output comparable clean relation tuples rather than taking gold tuples as input.

Summary

In this thesis we explored various research problems in information extraction, with a focus on utilizing indirect supervision by exploiting outside supplementary data or the data itself inherent traits. We first start with tackling the named entity recognition problem, then relation extraction problem. We extend to open domain information extraction and also propose novel tasks related to extracting cultural differences in the social media domain.

In the first chapter, we proposed a sequence labeling framework, LM-LSTM-CRF , which effectively leverages the language model to extract character-level knowledge from the self-contained order information. Highway layers are incorporated to overcome the discordance issue of the naive co-training Benefited from the effectively captured such task-specific knowledge, we can build a much more concise model, thus yielding much better efficiency without loss of effectiveness (achieved the state-of-the-art on three benchmark datasets) . In the future, we plan to further extract and incorporate knowledge from other "unsupervised" learning principles and empower more sequence labeling tasks.

In the second chapter, we present a novel multi-channel BiLSTM-CRF model for emerging named entity recognition in social media messages. We find that BiLST-CRF architecture with our proposed comprehensive word representations built from multiple information are effective to overcome the noisy and short nature of social media messages.

In the thrid chapter, we present a novel study on indirect supervision (from question-answering datasets) for the task of relation extraction. We propose a framework, ReQuest, that embeds information from both training data automatically generated by linking to knowledge bases and QA datasets, and captures richer semantic knowledge from both sources via shared text features so that better feature embeddings can be learned to infer relation type for test relation mentions despite the noisy training data. Our experiment results on two datasets demonstrate the effectiveness and robustness of ReQuest. Interesting future work includes identifying most relevant QA pairs for target relation types, generating most effective questions to collect feedback (or answers) via crowd-sourcing, and exploring approaches other than distant supervision[184, 185].

In the fourth chapter, we propose to identify LocatedNear relation from literature text and construct a knowledge base of object pairs that would commonly appear near each other in real world. In this chapter, we present a novel study on enriching LocatedNear relationship from textual corpora. Based on our two newly-collected benchmark datasets, we propose several methods to solve the sentence-level relation classification problem. We show that existing methods do not work as well on this task and discovered that LSTM-based model does not have significant edge over simpler feature-based model. Whereas, our multi-level sentence normalization turns out to be useful. Future directions include: 1) better leveraging distant supervision to reduce human efforts, 2) incorporating knowledge graph embedding techniques, 3) applying the LocatedNear knowledge into downstream applications in computer vision and natural language processing.

In the fifth chapter, we present the SocVec method to compute cross-cultural differences and similarities, and evaluate it on two novel tasks about mining cross-cultural differences in named entities and computing cross-cultural similarities in slang terms. Through extensive experiments, we demonstrate that the proposed lightweight yet effective method outperforms a number of baselines, and can be useful in translation applications and cross-cultural studies in computational social science. Future directions include: 1) mining cross-cultural differences in general concepts other than names and slang, 2) merging the mined knowledge into existing knowledge bases, and 3) applying the SocVec in downstream tasks like machine translation.

In the final chapter, we study the task of open information extraction and proposes a principled framework, ReMine , to unify local contextual information and global structural cohesiveness for effective extraction of relation tuples. ReMine leverages distant supervision in conjunction with existing knowledge bases to provide automatically-labeled sentence and guide the entity and relation segmentation. The local objective is further learned together with a translating-based objective to enforce structural cohesiveness, such that corpus-level statistics are incorporated for boosting high-quality tuples extracted from individual sentences. We develop a joint optimization algorithm to efficiently solve the proposed unified objective function and can output quality extractions by taking into account both local and global information. Experiments on two real-world corpora of different domains demonstrate that ReMine system achieves superior precision when outputting same number of extractions, compared with several state-of-the-art open IE systems. As a byproduct, ReMine also demonstrates competitive performance on detecting mentions of entities from text when compared to several named entity recognition algorithms.

Bibliography

[1] SARAWAGI S. Information extraction[J]. Foundations and trends in databases, 2008, 1(3): 261–377.

[2] DONG X L, STROHMANN T, SUN S, et al. Knowledge Vault: A Web-scale approach to probabilistic knowledge fusion[C]// KDD. [S.l.]: [s.n.], 2014.

[3] FADER A, ZETTLEMOYER L, ETZIONI O. Open Question Answering Over Curated and Extracted Knowledge Bases[J]. KDD, 2014.

[4] SUN H, MA H, YIH W.-T, et al. Open domain question answering via semantic enrichment[C]// WWW. [S.l.]: [s.n.], 2015.

[5] DAGAN I, DOLAN B, MAGNINI B, et al. Recognizing textual entailment: Rational, evaluation and approaches[J]. Natural Language Engineering, 2009, 15(4): i–xvii. DOI: 10.1017/S1351324909990234.

[6] BOWMAN S R, ANGELI G, POTTS C, et al. A large annotated corpus for learning natural language inference[C/OL]// Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Lisbon, Portugal: Association for Computational Linguistics, 2015: 632–642. http://www.aclweb.org/anthology/D15-1075. DOI: 10.18653/v1/D15-1075.

[7] ZHU Y, FATHI A, FEI-FEI L. Reasoning about object affordances in a knowledge base representation[C]// European conference on computer vision. Springer. [S.l.]: [s.n.], 2014: 408–424. DOI: 10.1007/978-3-319-10605-2_27.

[8] SPEER R, HAVASI C. Representing General Relational Knowledge in ConceptNet 5[C/OL]// Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012). Istanbul, Turkey: European Language Resources Association (ELRA), 2012. http://www.aclweb.org/anthology/L12-1639.

[9] MINTZ M, BILLS S, SNOW R, et al. Distant supervision for relation extraction without labeled data[C]// ACL/IJCNLP. [S.l.]: [s.n.], 2009.

[10] RIEDEL S, YAO L, MCCALLUM A. Modeling Relations and Their Mentions without Labeled Text[C]// ECML/PKDD. [S.l.]: [s.n.], 2010.

[11] HOFFMANN R, ZHANG C, LING X, et al. Knowledge-Based Weak Supervision for Information Extraction of Overlapping Relations[C]// ACL. [S.l.]: [s.n.], 2011.

[12] LIN Y, SHEN S, LIU Z, et al. Neural Relation Extraction with Selective Attention over Instances[C]// ACL. [S.l.]: [s.n.], 2016.

[13] BANKO M, CAFARELLA M J, SODERLAND S, et al. Open information extraction from the web[C]// IJCAI. [S.l.]: [s.n.], 2007.

[14] CARLSON A, BETTERIDGE J, WANG R C, et al. Coupled semi-supervised learning for information extraction[C]// WSDM. [S.l.]: [s.n.], 2010.

[15] SCHMITZ M, BART R, SODERLAND S, et al. Open language learning for information extraction[C]// EMNLP-CoNLL. [S.l.]: [s.n.], 2012.

[16] MA X, HOVY E. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF[C]// ACL. [S.l.]: [s.n.], 2016.

[17] SHA F, PEREIRA F. Shallow parsing with conditional random fields[C]// NAACL-HLT. [S.l.]: [s.n.], 2003.

[18] LIU L, REN X, ZHU Q, et al. Heterogeneous Supervision for Relation Extraction: A Representation Learning Approach[J]. Proc. EMNLP, 2017.

[19] LUO G, HUANG X, LIN C.-Y, et al. Joint named entity recognition and disambiguation[C]// EMNLP. [S.l.]: [s.n.], 2015.

[20] PENG N, DREDZE M. Improving named entity recognition for chinese social media with word segmentation representation learning[C]// ACL. [S.l.]: [s.n.], 2016.

[21] MIKOLOV T, SUTSKEVER I, CHEN K, et al. Distributed representations of words and phrases and their compositionality[C]// NIPS. [S.l.]: [s.n.], 2013.

[22] PENNINGTON J, SOCHER R, MANNING C. Glove: Global Vectors for Word Representation[C/OL]// Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar: Association for Computational Linguistics, 2014: 1532–1543. http://www.aclweb.org/anthology/D14-1162. DOI: 10.3115/v1/D14-1162.

[23] LAMPLE G, BALLESTEROS M, KAWAKAMI K, et al. Neural Architectures for Named Entity Recognition[C]// NAACL-HLT. [S.l.]: [s.n.], 2016.

[24] PETERS M E, AMMAR W, BHAGAVATULA C, et al. Semi-supervised sequence tagging with bidirectional language models[J]. ArXiv:1705.00108, 2017.

[25] REI M. Semi-supervised Multitask Learning for Sequence Labeling[C]// ACL. [S.l.]: [s.n.], 2017.

[26] SRIVASTAVA R K, GREFF K, SCHMIDHUBER J. Highway networks[J]. ArXiv:1505.00387, 2015.

[27] YANG Z, SALAKHUTDINOV R, COHEN W W. Transfer learning for sequence tagging with hierarchical recurrent networks[J]. ArXiv:1703.06345, 2017.

[28] KARPATHY A. The Unreasonable Effectiveness of Recurrent Neural Networks. http://karpathy.github.io/2015/05/21/rnn-effectiveness/. Accessed: 2017-08-22. 2015.

[29] FERNANDEZ J, YU Z, DOWNEY D. VecShare: A Framework for Sharing Word Representation Vectors[J]. 2017.

[30] TJONG KIM SANG E F, DE MEULDER F. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition[C]// Natural language learning at NAACL-HLT. [S.l.]: [s.n.], 2003.

[31] TJONG KIM SANG E F, BUCHHOLZ S. Introduction to the CoNLL-2000 shared task: Chunking[C]// Learning language in logic and CoNLL. [S.l.]: [s.n.], 2000.

[32] MARCUS M P, MARCINKIEWICZ M A, SANTORINI B. Building a large annotated corpus of English: The Penn Treebank[J]. Computational linguistics, 1993.

[33] MANNING C D. Part-of-speech tagging from 97% to 100%: is it time for some linguistics?[C]// International Conference on Intelligent Text Processing and Computational Linguistics. Springer. [S.l.]: [s.n.], 2011.

[34] RATINOV L, ROTH D. Design challenges and misconceptions in named entity recognition[C]// CoNLL. [S.l.]: [s.n.], 2009.

[35] GLOROT X, BENGIO Y. Understanding the difficulty of training deep feedforward neural networks[C]// Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. [S.l.]: [s.n.], 2010.

[36] JOZEFOWICZ R, ZAREMBA W, SUTSKEVER I. An empirical exploration of recurrent network architectures[C]// ICML. [S.l.]: [s.n.], 2015.

[37] CHIU J P C, NICHOLS E. Named Entity Recognition with Bidirectional LSTM-CNNs[J]. TACL, 2016.

[38] COLLOBERT R, WESTON J, BOTTOU L, et al. Natural Language Processing (almost) from Scratch[J]. JMLR, 2011.

[39] SØGAARD A, GOLDBERG Y. Deep multi-task learning with low level tasks supervised at lower layers[C]// ACL. [S.l.]: [s.n.], 2016.

[40] SUN X. Structure regularization for structured prediction[C]// NIPS. [S.l.]: [s.n.], 2014.

[41] JOZEFOWICZ R, VINYALS O, SCHUSTER M, et al. Exploring the limits of language modeling[J]. ArXiv:1602.02410, 2016.

[42] REIMERS N, GUREVYCH I. Reporting Score Distributions Makes a Difference: Performance Study of LSTM-networks for Sequence Tagging[J]. ArXiv preprint arXiv:1707.09861, 2017.

[43] SØGAARD A. Semisupervised condensed nearest neighbor for part-of-speech tagging[C]// NAACL-HLT. [S.l.]: [s.n.], 2011.

[44] HASHIMOTO K, XIONG C, TSURUOKA Y, et al. A joint many-task model: Growing a neural network for multiple NLP tasks[J]. ArXiv:1611.01587, 2016.

[45] LAFFERTY J D, MCCALLUM A, PEREIRA F. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data[C]// ICML. [S.l.]: [s.n.], 2001.

[46] MCCALLUM A, LI W. Early results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons[C]// CoNLL. [S.l.]: [s.n.], 2003.

[47] FLORIAN R, ITTYCHERIAH A, JING H, et al. Named Entity Recognition through Classifier Combination[C]// CoNLL. [S.l.]: [s.n.], 2003.

[48] CHIEU H L, NG H T. Named Entity Recognition: A Maximum Entropy Approach Using Global Information[C]// COLING. [S.l.]: [s.n.], 2002.

[49] SHANG J, LIU J, JIANG M, et al. Automated Phrase Mining from Massive Text Corpora[J]. ArXiv:1702.04457, 2017.

[50] BALDWIN T, KIM Y.-B, DE MARNEFFE M C, et al. Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition[J]. ACL-IJCNLP, 2015, 126: 2015.

[51] DERCZYNSKI L, MAYNARD D, RIZZO G, et al. Analysis of named entity recognition and linking for tweets[J]. Information Processing & Management, 2015, 51(2): 32–49.

[52] AUGENSTEIN I, DERCZYNSKI L, BONTCHEVA K. Generalisation in named entity recognition: A quantitative analysis[J]. Computer Speech & Language, 2017, 44: 61–83.

[53] ZHANG B, HUANG H, PAN X, et al. Context-aware Entity Morph Decoding[C]// Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers. [S.l.]: [s.n.], 2015: 586–595.

[54] MA X, HOVY E H. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF[C]// Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers. [S.l.]: [s.n.], 2016.

[55] LAMPLE G, BALLESTEROS M, SUBRAMANIAN S, et al. Neural Architectures for Named Entity Recognition[C]// NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016. [S.l.]: [s.n.], 2016: 260–270.

[56] OWOPUTI O, O'CONNOR B, DYER C, et al. Improved part-of-speech tagging for online conversational text with word clusters[C]//. Association for Computational Linguistics. [S.l.]: [s.n.], 2013.

[57] KONG L, SCHNEIDER N, SWAYAMDIPTA S, et al. A dependency parser for tweets[J]. 2014.

[58] GIMPEL K, SCHNEIDER N, O'CONNOR B, et al. Part-of-speech tagging for twitter: Annotation, features, and experiments[C] // Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2. Association for Computational Linguistics. [S.l.]: [s.n.], 2011: 42–47.

[59] XU Y, JIA R, MOU L, et al. Improved relation classification by deep recurrent neural networks with data augmentation[J]. ArXiv preprint, 2016, arXiv:1601.03651.

[60] PENNINGTON J, SOCHER R, MANNING C D. Glove: Global Vectors for Word Representation[C] // EMNLP. [S.l.]: [s.n.], 2014.

[61] HE K, ZHANG X, REN S, et al. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification[C] // 2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015. [S.l.]: [s.n.], 2015: 1026–1034.

[62] GLOROT X, BENGIO Y. Understanding the difficulty of training deep feedforward neural networks[C] // Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010. [S.l.]: [s.n.], 2010: 249–256.

[63] JÓZEFOWICZ R, ZAREMBA W, SUTSKEVER I. An Empirical Exploration of Recurrent Network Architectures[C] // Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015. [S.l.]: [s.n.], 2015: 2342–2350.

[64] KINGMA D, BA J. Adam: A method for stochastic optimization[J]. ArXiv preprint arXiv:1412.6980, 2014.

[65] LAFFERTY J D, MCCALLUM A, PEREIRA F C N. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data[C] // Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001. [S.l.]: [s.n.], 2001: 282–289.

[66] MCCALLUM A, LI W. Early results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons[C/OL] // Proceedings of the Seventh Conference on Natural Language Learning, CoNLL 2003, Held in cooperation with HLT-NAACL 2003, Edmonton, Canada, May 31 - June 1, 2003. [S.l.]: [s.n.], 2003: 188–191. http://aclweb.org/anthology/W/W03/W03-0430.pdf.

[67] HUANG Z, XU W, YU K. Bidirectional LSTM-CRF Models for Sequence Tagging[J]. CoRR, 2015, abs/1508.01991.

[68] LIMSOPATHAM N, COLLIER N. Bidirectional LSTM for Named Entity Recognition in Twitter Messages[C]//. [S.l.]: [s.n.], 2016.

[69] STRAUSS B, TOMA B E, RITTER A, et al. Results of the WNUT16 Named Entity Recognition Shared Task[C]//. [S.l.]: [s.n.], 2016.

[70] MENDES P N, JAKOB M, GARCIA-SILVA A, et al. DBpedia spotlight: shedding light on the web of documents[C]// I-Semantics. [S.l.]: [s.n.], 2011.

[71] HOFFART J, YOSEF M A, BORDINO I, et al. Robust disambiguation of named entities in text[C]// EMNLP. [S.l.]: [s.n.], 2011.

[72] MANNING C D, SURDEANU M, BAUER J, et al. The Stanford CoreNLP Natural Language Processing Toolkit[C]// ACL. [S.l.]: [s.n.], 2014.

[73] CHAN Y S, ROTH D. Exploiting background knowledge for relation extraction[C]// COLING. [S.l.]: [s.n.], 2010.

[74] ZHOU G, SU J, ZHANG J, et al. Exploring Various Knowledge in Relation Extraction[C]// ACL. [S.l.]: [s.n.], 2005.

[75] NGUYEN N, CARUANA R. Classification with partial labels[C]// KDD. [S.l.]: [s.n.], 2008.

[76] TANG J, QU M, WANG M, et al. Line: Large-scale information network embedding[C]// WWW. [S.l.]: [s.n.], 2015.

[77] RAO J, HE H, LIN J J. Noise-Contrastive Estimation for Answer Selection with Deep Neural Networks[C]// CIKM. [S.l.]: [s.n.], 2016.

[78] SHALEV-SHWARTZ S, SINGER Y, SREBRO N, et al. Pegasos: Primal estimated sub-gradient solver for svm[J]. Mathematical programming, 2011, 127(1): 3–30.

[79] LING X, WELD D S. Fine-Grained Entity Recognition[C]// AAAI. [S.l.]: [s.n.], 2012.

[80] ELLIS J, GETMAN J, MOTT J, et al. Linguistic Resources for 2013 Knowledge Base Population Evaluations[C]// TAC. [S.l.]: [s.n.], 2014.

[81] REN X, WU Z, HE W, et al. CoType: Joint Extraction of Typed Entities and Relations with Knowledge Bases[C]// WWW. [S.l.]: [s.n.], 2017.

[82] WANG M, SMITH N A, MITAMURA T. What is the Jeopardy Model? A Quasi-Synchronous Grammar for QA[C]// EMNLP-CoNLL. [S.l.]: [s.n.], 2007.

[83] WANG Z, ITTYCHERIAH A. FAQ-based Question Answering via Word Alignment[J]. ArXiv preprint, 2015, arXiv:1507.02628.

[84] TAN M, SANTOS C D, XIANG B, et al. Lstm-based deep learning models for non-factoid answer selection[J]. ArXiv preprint, 2015, arXiv:1511.04108.

[85] Dos SANTOS C N, TAN M, XIANG B, et al. Attentive Pooling Networks[C]//. Vol. arXiv:1602.03609. [S.l.]: [s.n.], 2016.

[86] YAO X, DURME B V, CALLISON-BURCH C, et al. Answer Extraction as Sequence Tagging with Tree Edit Distance[C]// NAACL. [S.l.]: [s.n.], 2013.

[87] YAO X, DURME B V, CLARK P. Automatic Coupling of Answer Extraction and Information Retrieval[C]// ACL. [S.l.]: [s.n.], 2013.

[88] MOONEY R J, BUNESCU R C. Subsequence kernels for relation extraction[C]// NIPS. [S.l.]: [s.n.], 2005.

[89] PEROZZI B, AL-RFOU R, SKIENA S. Deepwalk: Online learning of social representations[C]// KDD. [S.l.]: [s.n.], 2014.

[90] GORMLEY M R, YU M, DREDZE M. Improved relation extraction with feature-rich compositional embedding models[C]// EMNLP. [S.l.]: [s.n.], 2015.

[91] XU Y, MOU L, LI G, et al. Classifying Relations via Long Short Term Memory Networks along Shortest Dependency Paths.[C]// EMNLP. [S.l.]: [s.n.], 2015.

[92] HENDRICKX I, KIM S N, KOZAREVA Z, et al. SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals[C]// SemEval@ACL. [S.l.]: [s.n.], 2010.

[93] MIWA M, BANSAL M. End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures[J]. ArXiv preprint, 2016, arXiv:1601.00770.

[94] DODDINGTON G R, MITCHELL A, PRZYBOCKI M A, et al. The Automatic Content Extraction (ACE) Program - Tasks, Data, and Evaluation[C]// LREC. [S.l.]: [s.n.], 2004.

[95] LI Q, JI H. Incremental Joint Extraction of Entity Mentions and Relations[C]// ACL. [S.l.]: [s.n.], 2014.

[96] BACH N, BADASKAR S. A Review of Relation Extraction[C]//Literature review for Language and Statistics II. [S.l.]: [s.n.], 2007.

[97] SURDEANU M, JI H. Overview of the English Slot Filling Track at the TAC2014 Knowledge Base Population Evaluation[C]//TAC. [S.l.]: [s.n.], 2014.

[98] SAVENKOV D, LU W.-L, DALTON J, et al. Relation Extraction from Community Generated Question-Answer Pairs[C]//NAACL. [S.l.]: [s.n.], 2015.

[99] SOCHER R, PENNINGTON J, HUANG E H, et al. Semi-supervised recursive autoencoders for predicting sentiment distributions[C]//EMNLP. [S.l.]: [s.n.], 2011.

[100] EBRAHIMI J, DOU D. Chain Based RNN for Relation Classification[C]//NAACL. [S.l.]: [s.n.], 2015.

[101] QUIRK C, POON H. Distant Supervision for Relation Extraction beyond the Sentence Boundary[J]. ArXiv preprint, 2016, arXiv:1609.04873.

[102] HAN X, SUN L. Global Distant Supervision for Relation Extraction[C]//AAAI. [S.l.]: [s.n.], 2016.

[103] XU W, HOFFMANN R, ZHAO L, et al. Filling Knowledge Base Gaps for Distant Supervision of Relation Extraction[C]//ACL. [S.l.]: [s.n.], 2013.

[104] BANKO M, CAFARELLA M J, SODERLAND S, et al. Open Information Extraction from the Web[C]//IJCAI. [S.l.]: [s.n.], 2007.

[105] POON H, DOMINGOS P M. Joint Unsupervised Coreference Resolution with Markov Logic[C]//EMNLP. [S.l.]: [s.n.], 2008.

[106] TOUTANOVA K, CHEN D, PANTEL P, et al. Representing Text for Joint Embedding of Text and Knowledge Bases[C]//EMNLP. [S.l.]: [s.n.], 2015.

[107] NARASIMHAN K, YALA A, BARZILAY R. Improving Information Extraction by Acquiring External Evidence with Reinforcement Learning[C]//EMNLP. [S.l.]: [s.n.], 2016.

[108] KANANI P H, MCCALLUM A. Selecting actions for resource-bounded information extraction using reinforcement learning[C]//WSDM. [S.l.]: [s.n.], 2012.

[109] SAVENKOV D, AGICHTEIN E. When a Knowledge Base Is Not Enough: Question Answering over Knowledge Bases with External Text Data[C]//SIGIR. [S.l.]: [s.n.], 2016.

[110] ITTYCHERIAH A, FRANZ M, ZHU W.-J, et al. IBM's Statistical Question Answering System[C]// TREC. [S.l.]: [s.n.], 2000.

[111] ELWORTHY D. Question Answering Using a Large NLP System[C]// TREC. [S.l.]: [s.n.], 2000.

[112] KHALID M, VERBERNE S. Passage Retrieval for Question Answering using Sliding Windows[C]// IRQA@COLING. [S.l.]: [s.n.], 2008: 26–33.

[113] WADE C, ALLAN J. Passage Retrieval and Evaluation[C]// Tech. Reports of DTIC. [S.l.]: [s.n.], 2005.

[114] CLARKE C L A, CORMACK G V, KISMAN D I E, et al. Question Answering by Passage Selection (MultiText Experiments for TREC-9)[C]// TREC. [S.l.]: [s.n.], 2000.

[115] CORRADA-EMMANUEL A, CROFT W B, MURDOCK V. Answer Passage Retrieval for Question Answering[C]// Tech. Reports of CIIR UMass. [S.l.]: [s.n.], 2003.

[116] YATSKAR M, ORDONEZ V, FARHADI A. Stating the Obvious: Extracting Visual Common Sense Knowledge[C/OL]// Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. San Diego, California: Association for Computational Linguistics, 2016: 193–198. http://www.aclweb.org/anthology/N16-1023. DOI: 10.18653/v1/N16-1023.

[117] LI X, TAHERI A, TU L, et al. Commonsense knowledge base completion[C/OL]// Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL), Berlin, Germany, August. Association for Computational Linguistics. [S.l.]: [s.n.], 2016: 1445–1455. http://www.aclweb.org/anthology/P16-1137. DOI: 10.18653/v1/P16-1137.

[118] HOCHREITER S, SCHMIDHUBER J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735–1780.

[119] ZENG D, LIU K, LAI S, et al. Relation Classification via Convolutional Deep Neural Network[C/OL]// Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers. Dublin, Ireland: Dublin City University, Association for Computational Linguistics, 2014: 2335–2344. http://www.aclweb.org/anthology/C14-1220.

[120] ZAREMBA W, SUTSKEVER I, VINYALS O. Recurrent neural network regularization[J/OL]. ArXiv preprint arXiv:1409.2329, 2014. https://arxiv.org/abs/1409.2329.

[121] XU Y, JIA R, MOU L, et al. Improved relation classification by deep recurrent neural networks with data augmentation[C/OL]// Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. Osaka, Japan: The COLING 2016 Organizing Committee, 2016: 1461–1470. http://www.aclweb.org/anthology/C16-1138.

[122] LIN T.-Y, MAIRE M, BELONGIE S, et al. Microsoft coco: Common objects in context[C]// European Conference on Computer Vision. Springer. [S.l.]: [s.n.], 2014: 740–755.

[123] LE Q V, MIKOLOV T. Distributed Representations of Sentences and Documents[C]// Proc. of ICML. [S.l.]: [s.n.], 2014. DOI: 10.1.1.646.3937.

[124] RUDER S, VULI I, SØGAARD A. A survey of cross-lingual embedding models[J/OL]. ArXiv preprint arXiv:1706.04902, 2017. https://arxiv.org/pdf/1706.04902.pdf.

[125] CAMACHO-COLLADOS J, PILEHVAR M T, COLLIER N, et al. SemEval-2017 Task 2: Multilingual and Cross-lingual Semantic Word Similarity[C/OL]// Proc. of SemEval@ACL. [S.l.]: [s.n.], 2017. http://www.aclweb.org/anthology/S17-2002. DOI: 10.18653/v1/S17-2002.

[126] SARATH C A P, LAULY S, LAROCHELLE H, et al. An Autoencoder Approach to Learning Bilingual Word Representations[C/OL]// Proc. in NIPS. [S.l.]: [s.n.], 2014. https://papers.nips.cc/paper/5270-an-autoencoder-approach-to-learning-bilingual-word-representations.pdf.

[127] KOISKÝ T, HERMANN K M, BLUNSOM P. Learning Bilingual Word Representations by Marginalizing Alignments[C/OL]// Proc. of ACL. [S.l.]: [s.n.], 2014. http://www.aclweb.org/anthology/P14-2037. DOI: 10.3115/v1/P14-2037.

[128] UPADHYAY S, FARUQUI M, DYER C, et al. Cross-lingual models of word embeddings: An empirical comparison[C/OL]// Proc. of ACL. [S.l.]: [s.n.], 2016. http://www.aclweb.org/anthology/P16-1157. DOI: 10.18653/v1/P16-1157.

[129] KITAYAMA S, MARKUS H R, KUROKAWA M. Culture, emotion, and well-being: Good feelings in Japan and the United States[J]. Cognition & Emotion, 2000, 14(1): 93–124. DOI: 10.1080/026999300379003.

[130] GAREIS E, WILKINS R. Love expression in the United States and Germany[J/OL]. International Journal of Intercultural Relations, 2011, 35(3): 307–319. https://doi.org/10.1016%2Fj.ijintrel.2010.06.006. DOI: 10.1016/j.ijintrel.2010.06.006.

[131] TAUSCZIK Y R, PENNEBAKER J W. The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods[J/OL]. Journal of Language and Social Psychology, 2009, 29(1): 24–54. https://doi.org/10.1177%2F0261927x09351676. DOI: 10.1177/0261927x09351676.

[132] ELAHI M F, MONACHESI P. An Examination of Cross-Cultural Similarities and Differences from Social Media Data with respect to Language Use.[C/OL]// Proc. of LREC. [S.l.]: [s.n.], 2012. http://www.lrec-conf.org/proceedings/lrec2012/pdf/942_Paper.pdf.

[133] GARIMELLA A, MIHALCEA R, PENNEBAKER J W. Identifying Cross-Cultural Differences in Word Usage[C/OL]// Proc. of COLING. [S.l.]: [s.n.], 2016. http://www.aclweb.org/anthology/C16-1065.

[134] FU K.-W, CHAN C.-H, CHAU M. Assessing censorship on microblogs in China: Discriminatory keyword analysis and the real-name registration policy[J]. IEEE Internet Computing, 2013, 17(3): 42–50. DOI: 10.1109/MIC.2013.28.

[135] HAN B, COOK P, BALDWIN T. Automatically constructing a normalisation dictionary for microblogs[C/OL]// Proc. of EMNLP-CoNLL. [S.l.]: [s.n.], 2012. http://www.aclweb.org/anthology/D12-1039.

[136] CHE W, LI Z, LIU T. Ltp: A chinese language technology platform[C/OL]// Proc. of COLING 2010: Demonstrations. [S.l.]: [s.n.], 2010. http://www.aclweb.org/anthology/C10-3004.

[137] RATINOV L, ROTH D, DOWNEY D, et al. Local and global algorithms for disambiguation to wikipedia[C/OL]// Proc. of ACL. [S.l.]: [s.n.], 2011. http://www.aclweb.org/anthology/P11-1138.

[138] CHENG X, ROTH D. Relational Inference for Wikification[C/OL]// Proc. of EMNLP. [S.l.]: [s.n.], 2013. http://www.aclweb.org/anthology/D13-1184.

[139] MIKOLOV T, SUTSKEVER I, CHEN K, et al. Distributed representations of words and phrases and their compositionality[C/OL]// Proc. of NIPS. [S.l.]: [s.n.], 2013. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

[140] FAST E, CHEN B, BERNSTEIN M S. Empath: Understanding topic signals in large-scale text[C]// Proc. of CHI. [S.l.]: [s.n.], 2016. DOI: 10.1145/2858036.2858535.

[141] CHOI Y, CARDIE C, RILOFF E, et al. Identifying sources of opinions with conditional random fields and extraction patterns[C/OL]// Proc. of HLT-EMNLP. [S.l.]: [s.n.], 2005. http://www.aclweb.org/anthology/H05-1045.

[142] GAO R, HAO B, LI H, et al. Developing simplified Chinese psychological linguistic analysis dictionary for microblog[C]// Proceedings of International Conference on Brain and Health Informatics. Springer. [S.l.]: [s.n.], 2013. DOI: 10.1007/978-3-319-02753-1_36.

[143] HARRIS Z S. Distributional structure[J]. Word, 1954, 10(2-3): 146–162. DOI: 10.1080/00437956.1954.11659520.

[144] LANDIS J R, KOCH G G. The measurement of observer agreement for categorical data.[J]. Biometrics, 1977, 33 1: 159–74. DOI: 10.2307/2529310.

[145] WANG S, BOND F. Building the chinese open wordnet (cow): Starting from core synsets[C/OL]// Proceedings of the 11th Workshop on Asian Language Resources, a Workshop at IJCNLP. [S.l.]: [s.n.], 2013. http://www.aclweb.org/anthology/W13-4302.

[146] MIKOLOV T, LE Q V, SUTSKEVER I. Exploiting similarities among languages for machine translation[J]. ArXiv preprint arXiv:1309.4168, 2013. DOI: 10.1.1.754.2995.

[147] AMMAR W, MULCAIRE G, TSVETKOV Y, et al. Massively multilingual word embeddings[J/OL]. ArXiv preprint arXiv:1602.01925, 2016. https://arxiv.org/pdf/1602.01925.pdf.

[148] FARUQUI M, DYER C. Improving Vector Space Word Representations Using Multilingual Correlation[C/OL]// Proc. of EACL. [S.l.]: [s.n.], 2014. http://aclweb.org/anthology/E/E14/E14-1049.pdf.

[149] DUONG L, KANAYAMA H, MA T, et al. Learning Crosslingual Word Embeddings without Bilingual Corpora[C/OL]// Proc. of EMNLP. [S.l.]: [s.n.], 2016. http://www.aclweb.org/anthology/D16-1136. DOI: 10.18653/v1/D16-1136.

[150] PETROVIC S, OSBORNE M, LAVRENKO V. Streaming First Story Detection with application to Twitter[C/OL]// Proc. of HLT-NAACL. [S.l.]: [s.n.], 2010. http://www.aclweb.org/anthology/N10-1021.

[151] PAUL M J, DREDZE M. You Are What You Tweet: Analyzing Twitter for Public Health[C/OL]// Proc. of ICWSM. [S.l.]: [s.n.], 2011. http://www.cs.jhu.edu/~mpaul/files/2011.icwsm.twitter_health.pdf.

[152] ROSENTHAL S, MCKEOWN K. I Couldn't Agree More: The Role of Conversational Structure in Agreement and Disagreement Detection in Online Discussions[C]// Proc. of SIGDIAL. [S.l.]: [s.n.], 2015. DOI: 10.18653/v1/W15-4625.

[153] WANG W Y, YANG D. That's So Annoying!!!: A Lexical and Frame-Semantic Embedding Based Data Augmentation Approach to Automatic Categorization of Annoying Behaviors using #petpeeve Tweets[C/OL]// Proc. of EMNLP. [S.l.]: [s.n.], 2015. http://www.aclweb.org/anthology/D15-1306. DOI: 10.18653/v1/D15-1306.

[154] ZHANG B, HUANG H, PAN X, et al. Context-aware Entity Morph Decoding[C/OL]// Proc. of ACL. [S.l.]: [s.n.], 2015. http://www.aclweb.org/anthology/P15-1057. DOI: 10.3115/v1/P15-1057.

[155] LIN B Y, XU F F, LUO Z, et al. Multi-channel BiLSTM-CRF Model for Emerging Named Entity Recognition in Social Media[C/OL]// Proc. of W-NUT@EMNLP. [S.l.]: [s.n.], 2017. https://aclanthology.info/papers/W17-4421/w17-4421. DOI: 10.18653/v1/w17-4421.

[156] NAKASAKI H, KAWABA M, YAMAZAKI S, et al. Visualizing Cross-Lingual/Cross-Cultural Differences in Concerns in Multilingual Blogs.[C/OL]// Proc. of ICWSM. [S.l.]: [s.n.], 2009. https://pdfs.semanticscholar.org/484f/

9d44345015338e49c59d0a67210c276f7707.pdf?_ga=2.209852949.1831208069.1525515737-2139274556.1519386756.

[157] GARIMELLA K, MORALES G D F, GIONIS A, et al. Quantifying Controversy in Social Media[C]// Proc. of WSDM. [S.l.]: [s.n.], 2016. DOI: 10.1145/2835776.2835792.

[158] GUTIÉRREZ E D, SHUTOVA E, LICHTENSTEIN P, et al. Detecting Cross-cultural Differences Using a Multilingual Topic Model[J/OL]. TACL, 2016, 4: 47–60. http://www.aclweb.org/anthology/Q16-1004.

[159] ELSAHAR H, ELBELTAGY S R. A Fully Automated Approach for Arabic Slang Lexicon Extraction from Microblogs[C]// Proc. of CICLing. [S.l.]: [s.n.], 2014. DOI: 10.1007/978-3-642-54906-9_7.

[160] NI K, WANG W Y. Learning to Explain Non-Standard English Words and Phrases[C/OL]// Proc. of IJCNLP. [S.l.]: [s.n.], 2017. http://aclweb.org/anthology/I17-2070.

[161] LING W, XIANG G, DYER C, et al. Microblogs as Parallel Corpora[C/OL]// Proc. of ACL. [S.l.]: [s.n.], 2013. http://www.aclweb.org/anthology/P13-1018.

[162] KLEMENTIEV A, TITOV I, BHATTARAI B. Inducing Crosslingual Distributed Representations of Words[C/OL]// Proc. of COLING. [S.l.]: [s.n.], 2012. http://www.aclweb.org/anthology/C12-1089.

[163] FADER A, SODERLAND S, ETZIONI O. Identifying relations for open information extraction[C]// EMNLP. [S.l.]: [s.n.], 2011.

[164] LIU J, SHANG J, WANG C, et al. Mining quality phrases from massive text corpora[C]// SIGMOD. [S.l.]: [s.n.], 2015.

[165] ALLAHVERDYAN A, GALSTYAN A. Comparative analysis of viterbi training and maximum likelihood estimation for hmms[C]// NIPS. [S.l.]: [s.n.], 2011.

[166] XU Y, KIM M.-Y, QUINN K, et al. Open Information Extraction with Tree Kernels[C]// HLT-NAACL. [S.l.]: [s.n.], 2013.

[167] GAMALLO P, GARCIA M, FERNÁNDEZ-LANZA S. Dependency-Based Open Information Extraction[C]// ROBUS-UNSUP. [S.l.]: [s.n.], 2012.

[168] DEL CORRO L, GEMULLA R. Clausie: clause-based open information extraction[C]// WWW. [S.l.]: [s.n.], 2013.

[169] BORDES A, USUNIER N, GARCIA-DURAN A, et al. Translating embeddings for modeling multi-relational data[C]// NIPS. [S.l.]: [s.n.], 2013.

[170] GUU K, MILLER J, LIANG P. Traversing knowledge graphs in vector space[C]// EMNLP. [S.l.]: [s.n.], 2015.

[171] RIEDEL S, YAO L, MCCALLUM A, et al. Relation extraction with matrix factorization and universal schemas[C]// NAACL. [S.l.]: [s.n.], 2013.

[172] ZHANG C, ZHOU G, YUAN Q, et al. Geoburst: Real-time local event detection in geo-tagged tweet streams[C]// SIGIR. [S.l.]: [s.n.], 2016.

[173] DAIBER J, JAKOB M, HOKAMP C, et al. Improving Efficiency and Accuracy in Multilingual Entity Extraction[C]// I-Semantics. [S.l.]: [s.n.], 2013.

[174] ANGELI G, PREMKUMAR M J, MANNING C D. Leveraging linguistic structure for open domain information extraction[C]// ACL. [S.l.]: [s.n.], 2015.

[175] GASHTEOVSKI K, GEMULLA R, DEL CORRO L. MinIE: minimizing facts in open information extraction[C]// EMNLP. [S.l.]: [s.n.], 2017.

[176] HEARST M A. Automatic acquisition of hyponyms from large text corpora[C]// ACL. [S.l.]: [s.n.], 1992.

[177] CARLSON A, BETTERIDGE J, KISIEL B, et al. Toward an architecture for never-ending language learning[C]// AAAI. [S.l.]: [s.n.], 2010.

[178] MITCHELL T M, COHEN W W, HRUSCHKA JR E R, et al. Never ending learning[C]// AAAI. [S.l.]: [s.n.], 2015.

[179] NAKASHOLE N, WEIKUM G, SUCHANEK F. PATTY: A taxonomy of relational patterns with semantic types[C]// EMNLP-CoNLL. [S.l.]: [s.n.], 2012.

[180] BOLLACKER K, EVANS C, PARITOSH P, et al. Freebase: a collaboratively created graph database for structuring human knowledge[C]// SIGMOD. [S.l.]: [s.n.], 2008.

[181] LEHMANN J, ISELE R, JAKOB M, et al. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia[J]. Semantic Web, 2015, 6(2): 167–195.

[182] LUO Y, WANG Q, WANG B, et al. Context-dependent knowledge graph embedding.[C]// EMNLP. [S.l.]: [s.n.], 2015.

[183] NIU F, ZHANG C, RÉ C, et al. Deepdive: Web-scale knowledge-base construction using statistical learning and inference.[C]// VLDS. [S.l.]: [s.n.], 2012.

[184] RIEDEL S, YAO L, MCCALLUM A, et al. Relation Extraction with Matrix Factorization and Universal Schemas[C]// NAACL. [S.l.]: [s.n.], 2013.

[185] ARTZI Y, ZETTLEMOYER L S. Weakly Supervised Learning of Semantic Parsers for Mapping Instructions to Actions[J]. TACL, 2013, 1: 49–62.

Acknowledgements

I would like to express my deepest gratitude to my advisor Prof. Kenny Zhu, who has always been there to listen and give practical advice. His patience, enthusiasm, great kindness and immense knowledge helped me in all the time of research. Whenever I encountered difficulties and become frustrated, Prof. Zhu would be always willing to contribute his valuable feedback, advice and encouragement, illuminating the way for me to continue to search for the answers. I could not have asked for a better advisor. In addition to the academic collaboration, I greatly value Prof. Zhu's guidance on the personal integrity and the rigorous attitude toward research. Without his consistent instructions, this thesis could not be in this shape.

I am also thankful to ADAPT Lab for providing wonderful academic facilities and atmosphere. In particular, many thanks to my friends Bill Yuchen Lin, Zhiyi Luo in ADAPT Lab for numerous discussions on related topics and collaborations. I would also like to thank Prof. Jiawei Han and their group members for collaborating with me during my successful visit there. It's my pleasure to work with you during these years.

My gratitude would go to my beloved family, for their consistent encouragement and loving considerations during these years, wherein I gained great confidence in developing a good research work.

Publications

[1] Frank F. Xu*, Bill Y. Lin*, Kenny Q. Zhu, Seung-won Hwang. Mining Cross-Cultural Differences and Similarities in Social Media. ACL, 2018.

[2] Frank F. Xu*, Bill Y. Lin*, Kenny Q. Zhu. Automatic Extraction of Commonsense LocatedNear Knowledge. ACL, 2018.

[3] Zhiyi Luo, Shanshan Huang, Frank F. Xu, Bill Yuchen Lin, Hanyuan Shi, Kenny Zhu. ExtRA: Extracting Prominent Review Aspects from Customer Feedback. EMNLP, 2018.

[4] Liyuan Liu, Jingbo Shang, Frank F. Xu, Xiang Ren, Huan Gui, Jian Peng, Jiawei Han. Empower Sequence Labeling with Task-Aware Neural Language Model. AAAI, 2018.

[5] Zeqiu Wu, Xiang Ren, Frank F. Xu, Ji Li, Jiawei Han. ReQuest: Indirect Supervision for Relation Extraction using Question-Answer Pairs. WSDM, 2018.

[6] Qi Zhu, Xiang Ren, Jingbo Shang, Yu Zhang, Frank F. Xu, Jiawei Han. Open Information Extraction with Global Structure Constraints. WWW Poster, 2018.

[7] Frank F. Xu*, Bill Y. Lin*, Zhiyi Luo, Kenny Q. Zhu. Multi-channel BiLSTM-CRF Model for Emerging Named Entity Recognition in Social Media. W-NUT at EMNLP, 2017.

Projects

[1] NSFC grant 91646205

[2] NSFC grant 61373031