Master Thesis Proposal and Report

supervised by Prof.Dr.Klakow

Ehsan Khoddammohammadi
Faculty of Computational Linguistics
Saarland University
ehsank@coli.uni-saarland.de

July 25, 2013

Abstract

The aim of this paper is to review recent and influential methods on Unsupervised Relation Extraction. In the first chapter, the definition and a general background for relation extraction task is provided. In the second chapter, four major previous works are briefly reviewed. And finally in the last chapter, the paper will be concluded by identifying important aspects and difficulties of the task and important lessons learned from each of the discussed relation extraction systems.

Contents

Chapter 1

Introduction

Nowadays searching through Internet is the first step we take if we want to get answer to our question. We convert our questions to sequences of keywords and search engines are trying to lead us to web pages where might contain the answer to our questions, they return us a set of related documents which are sorted by their popularity and similarity of their text to our query. In this way, users themselves are responsible to find the desired knowledge from documents. The next generation search engines should go further than current approaches in understanding the meaning of a query and the underlying semantics of documents on the web. The next natural improvement is to return a piece of information which directly approaches to answer the user question. For this reason, the elements of a query, concepts or entities and their relations should be identified and documents (which might have the answer) should be mapped to the same space of entities and relations in order to find the the desired information in the question.

Relation Extraction is the task of detecting and classifying semantic relationship between named entities (NE). The goal of this task is to find a triple of binary relations and their arguments [1]. For instance, we want to induce a relation like *bornIn* with its arguments which could be for example like this: *(bornIn, Beata Nyari, Budapest)*. Applications of such task is numerous in natural language processing; Question/Answering, machine translation and text summarization are systems that benefit from relation extraction [1].

In this task, we are trying to have a model to find paraphrases which means that we are interested to have all the similar semantically similar relations under one umbrella. So it is desired to have all different surface realizations of one relation like *isGivenBirth, isBorn, isFrom* in a same set, namely *bornIn*.

From a classic method, DIRT [2], to very famous frameworks , TextRunner[3] or distant supervision[4], and more recent works e.g. PATTY[5], all are examples of several different family of approaches. These methods could be categorized from different perspectives (1) amount of annotated data they need (2) if they can only handle a predefined enumeration of entities and relations or are open to any number of relations (3) organization of semantic interpretation and (4) underlying family of methods they use.

Chapter 2

Weakly Supervised on Relation Extraction Task

2.1 Distant Supervision

The content of [6] your proposal. Each topic occupies one section, each with their own conclusion and future work.

Chapter 3

Major Recent Works on Unsupervised Relation Extraction

In this chapter we will review four major works on unsupervised relation discovery. We will discuss their formulation of the task, their models and the features they incorporate, the model constraints and finally we will take a look at how good a model is performing.

3.1 DIRT

Lin and Pantel in [2] proposed an unsupervised method for finding paraphrases from text which proved to be very influential and their method, **DIRT**, has become a classic work in literature. In addition to this paper we will also review two major improvements to DIRT proposed in [7] and [8]. DIRT tries to cluster semantically equivalent relations based on the similarity of their arguments. Relations (or inference rules) that DIRT can discover are mostly limited to paraphrases which are a subset of possible types of inference rules.

In this section, first, the key idea of DIRT will be elaborated more and then we will review the algorithm and analyze the observations. Based on these observations, two other methods, ISP [8] and LEDIR [7], which try to address a subset of DIRT's problems, will be discussed.

3.1.1 Extended Distributional Hypothesis

Synonymy of words and other semantic relations in word level have been studied very well in literature. For example Pereira et al. in [9] cluster similar words which convey a similar meaning. The key assumption in this work and related ones is so called **Distributional Hypothesis** [10] . It tells that words which usually occur in similar context have similar meaning. This idea can be generalized to phrases or predicates-arguments. A set of relations or phrases that appear in a similar context are semantically equivalent. Lin and Pantel have extended this idea with giving a slightly different notion of context. The context that they define in their work is the dependency path between a predicate of relation and its arguments. As the Extended Distributional Hypothesis states:

If two paths tend to occur in similar contexts, the meaning of the path tend to be similar

In this sense, the task of finding paraphrases or semantically equivalent relations can be formulated as *finding paths with similar meaning*. An algorithm to find such a similar path is a topic of the next part.

3.1.2 Model Description

DIRT starts with dependency parsing of sentences and continues with pruning of these dependency trees. It amounts to several conditions on dependency path such as that only nouns will be kept as arguments. Additionally, all the function words will be filtered to have the dependency relations that only connect two content words. Dependency relations with less than a certain number of occurrence will be removed to make the task feasible. After the pruning phase, a set of binary relations with a list of nouns associated to each of their slots is generated. The similarity of relations can be expressed based on similarity of associated list of slot words. Lin and Pantel have used *pointwise mutual information (pmi)* two measure similarity of slots. It measures the independency of two random variables or the amount of information they contain for each other. The similarity function for relations is a geometric average of similarity of correspondent slots.

Instead of clustering of relations which seems to be a natural next phase, they maintain a databases of triples (relation and its slots).For each relation queried they measure the similarity of relations which share at least one common slot word and then report top-40 of them.

3.1.3 Evaluation and Analysis

Based on what Lin and Pantel have reported in [2] they extracted 231,000 unique paths from a newspaper corpus. They have used a manually generated paraphrases to evaluate their method. For six questions from TREC-8 they have generated top-40 paraphrases and then evaluate them manually to report percentage of correctly found paraphrases. This approach has a very obvious flaw which is that number of returned paraphrases is fixed. Clustering has this advantage that each cluster of relations can have arbitrary number of paraphrases. Average accuracy for these six questions is 50.3% and no paraphrases have been found for one of the questions. The recall can't be measured easily since many correctly found paraphrases by DIRT are not discovered by humans.

Beside the problem of having fixed number of returned paraphrases, there are three other major problems after observing the results of DIRT which two of them will be addressed by Pantel et al. and will be discussed in the next parts. These problems are:

1. Antonymy of relations can not be captured with DIRT and antonym relations will be returned as similar relations. After more than a decade, it is still an open problem

2. DIRT results are very noisy in a sense that it doesn't induce the type of arguments.

3. DIRT results or paraphrases are always considered as bi-directional. Relation *A* implies relation *B* and vice versa. This assumption doesn't hold for many instances and needs to be regarded in the model.

In the next two sections ISP and LEDIR methods will be covered as proposed improvements to DIRT to filter erroneous relations.

3.1.4 Inference Selectional Preferences

Selectional preference of a predicate is the constraint that it puts over the class of words can be used as its arguments. For example: *x drives y* has a selectional preference on the second argument for motorized vehicles with more than two wheels and *x rides y* has a preference for vehicles or animals which rider is positioned astride [11] .

In order to model this type of constraint on discovery of paraphrases, Pantel et al. [8] have proposed that relations which are similar but are not sharing a same class of arguments should be filtered out. For each relation, they compute a distribution of possible semantic classes for its arguments. For two equivalent relations, they will be kept if this distribution is similar for both.Yao et al. [12] also incorporate selectional preferences to their model and which we will see in detailed in the last section.

Two models are proposed to compute this distribution. The first one, the distribution is computed jointly for both of arguments and the second one two different distributions are computed for each argument. The computation is just a simple count.

The question that naturally arises is that how one can obtain these semantic classes? The answer is that they are obtainable from lexical resources such as WordNet or they can be induced by clustering words. The authors have used both methods. As the first approach, they have used top level nodes in WordNet and all their successors together as semantic classes. They have also used CBC clustering algorithm [13] to induce semantic classes. In [14] and [15] a similar approach but with different clustering algorithms has been chosen and the later is the current state-of-the-art method.

After computing selectional preferences for each relation, those relations which don't have similar selectional preference distribution will be considered as nonequivalent. The plausibility of an inference rule which contains two relations is discussed in detailed in [7] and [16] .

Inducing word clusters and using independent selectional preferences distributions has been shown to outperform other models and have the highest increase of accuracy compare to DIRT.

3.1.5 Direction of Inference Rules

So far all the similar relations are considered to be paraphrases and they have bi-directional semantic equivalency relation.

Since this assumption doesn't hold for all the inference rules that are discovered by DIRT, Pantel et al. pushed their model another step further to find the direction of implications among relations, in addition to just find equivalent relations.

They have used the results from DIRT and ISP to find out which direction holds for an implication over two relations. having $relation_i \iff relation_j$ they want to examine if it holds or either of $relation_i \implies relation_j$ or $relation_j \implies relation_i$ is the correct rule.

the key assumption in [7] is that the direction is most likely from a specific relation to a more general relation. By specific relation they mean a relation that has a narrow set of semantic classes and selectional preferences and a general relation is a relation with broader semantic classes. Number of acceptable semantic classes for each relation is a good measure for such purpose. If this

number is greater for the antecedent then the direction is right otherwise it should be flipped. A similar approach has been used in [16] which additionally confirm the results of this work.

In the next section we will review a method which operates in web-scale and extract relations with bootstrapping.

3.2 TextRunner

TextRunner is the first system that addresses open domain relation extraction in web-scale [17]. In this section we will describe the architecture of TextRunner and then we will review two main elements of this system, *Learner* and *Extractor*, in more details and shortly we will comment on the third element *Assessor*.

3.2.1 TextRunner Architecture

Original TextRunner paper proposed four elements for its architecture:

- Learner
- Extractor
- Assessor
- Query Processor

The last sub-system has no major role in relation extraction so we put our focus on the first three elements.

3.2.2 Model Description

TextRunner avoids using parsing for large-scale and inhomogeneous corpus like web. *Learner* sub-system is responsible for providing a substitution or an approximation for syntactic parsing which, called as extractor by the authors. The approach that they follow in [17] is that they use a small corpus to train a relation classifier and this classifer will be applied to a much bigger web corpus. After parsing the small corpus, authors have used a set of predefined heuristics just based on POS tags and syntactic role of words to generate positive ad negative relation examples. The heuristics were designed as prototypes of syntactic behavior of general relations and they are not dependent on any certain type of relation. For example: $E_1 \ \ Verb \ \ E_2$ as template of *X created Y* or $E_1 \ \ NP \ Prep \ \ E_2$ for *X is birthplace of Y* are such heuristics.

A Naive Bayes classifier is self-trained on this small dataset, it starts with small seed of relations for training then it labels more instances in the corpus. This classifier will be used as *Extractor* in the next phase. This classifier is not relying on any lexical or relation-specific features, hence it can operate in open domain like web [18]. In the extraction phase, a maximum-entropy classifer was used to find entities and then by using the extractor learned from the previous phase, they recognize explicit relations among named entities.

These extracted relations and named entities contain many redundant or equivalent relations. The role of *Assessor* is to find equivalent relations. It starts with normalization of relations and entities and then based on string-similarity and shared relational attributes it finds explicitly equivalent relations, it should be mentioned that this approach differs from DIRT which finds semantically equivalent relations.

3.2.3 Evaluation

TextRunner is evaluated against a manually tagged dataset of 500 sentences. While the precision is in the accepted level, 86.6%, recall is low (23.2%). The recall has been improved by using a *Conditional Random Field* [18] as classifier instead of Naive Bayes. The model precision is slightly improved by using a CRF for about 2% but we see a tremendous increase in recall which levels to 45.2% .

TextRunner suffers from using very shallow features. Its ability to find relations is limited to explicitly mentioned relations and it can only find them if they occur in a sentence. In the next part we will review a model which does not suffer from these disadvantages due to its deeper semantic analysis.

3.3 USP

In this section we will review the first unsupervised semantic parsing method which proposed by Poon and Domingos [19], we will first define what is semantic parsing and after describing the method in [19] we show its application for relation extraction. In the evaluation subsection we will compare this model to its related models and analyze its advantages and disadvantages.

3.3.1 Semantic Parsing

Semantic parsing is mapping a sentence to its formal meaning representation [19] The aim is to represent a natural language text with first-order logic. One can derive a semantic parse of a sentence by starting from a lexicon of atomic formulas and combining each fragment to build a composition of formulas combined with quantifiers and logical connectives. In [19] the lexicon will be induced from a raw corpus. It is in contrast to traditional means of semantic parsing with manually produced lexicons.

The main challenge in unsupervised semantic parsing is that for a single semantic representation there could be several syntactic realizations or even harder, different surface representations. For example, all of the sentences below has a same semantic representation:

- Microsoft buys Skype
- Microsoft acquires the VoIP company Skype
- Skype is acquired by Microsoft Corporation
- The Redmond software giant buys Skype
- Microsofts purchase of Skype,...

A simple lexicon to represent all of the examples above is:

$$BUY(n_1)$$
$$\lambda x_2.BUYER(n_1, x_2) \;\; \lambda x_3.BOUGHT(n_1, x_3)$$
$$MICROSOFT(n_2) \;\; SKYPE(n_3)$$

Having a corpus of sentences in natural language, USP [19] will induce such a lexicon and will also extract a formal representation for each sentence. In the next subsection we will review the necessary steps toward this goal.

3.3.2 Model Description

In this section we will first identify the three key ideas that the model is built upon them and then we will describe the necessary steps toward unsupervised semantic parsing.

Three observations are made by the authors which is crucial to understand the model assumptions:

1. Different syntactic variations of predicates and constants can be clustered together to express a same meaning. This can be learned from a raw corpus, in contrast to supervised methods which use meaning annotation of text.

2. Not only fixed elements in a relation can be clustered together but also arbitrary forms with same sub-forms to co-occurred forms can be put in a cluster. In this way the meaning composition will be learned through clustering of the forms.

3. Learning syntax and semantics jointly is a complex problem. Authors have shown that translating a syntactic analysis to semantic parse has superior performance over joint learning of syntax and semantics. In this way, a model can be built on the shoulder of state-of-the-art syntactic parsers.

Based on these assumptions, they propose a method to model joint probability of the dependency tree and its meaning representation. Markov Logic Networks (MLN) [20] is used to represent the meaning of a sentence. For each sentence we will have an undirected graphical model (Markov network) which each nodes corresponds to atoms and cliques correspond to first-order clauses. The best meaning representation is the one which maximizes the probability of the observed dependency [21]. These are the major steps of the USP system:

Dependency Parsing

the authors believe that dependency parsing is a better starting point that phrase-structure parsing, since it expresses the relation-argument structure at the lexical level. They have used Minipar [22] to make the results more comparable with previous methods, e.g. DIRT.

Converting Dependency Trees into Quasi-Logical Forms

Extracting lambda forms and atoms (lexical entries) is done by a deterministic procedure. Arguments of a dependency path will be converted to atoms, their predicate consists of the lemma and their part-of-speech tags. Each edge from the head to arguments will be a predicate labeled as type of the dependency path. This predicate together with two lambda function for each argument form a QLF. In this QLF, only one constant can be presented.

Clustering Lambda-Forms

It starts from atom levels and cluster them based on their relation, then it recursively clusters larger formula based on their subformula lambda forms. The composition of meaning of any formula can be obtained by applying *lambda reduction* and substituting constants with lambda variables. At the end we will have a clustering of lambda forms which each cluster contains a set of semantically equivalent lambda forms. this solves the problem of having different syntactic realizations with a same meaning.

Creating a Markov Logic Network

Each atom can be modeled as a node and each first-order clause can be modeled as a feature. This is still not a complete Markov Logical Network since this model needs a weighting for features.Thus, the weight of feature is defined as a weight of its correspondent clause. A log-linear distribution

$$P(x) = \frac{1}{Z} \exp \Sigma_i (w_i n_i(x))$$

is used to model the probability of each clique (configuration of nodes and features). Z is a normalization factor which its computation is infeasible. w_i is the weight of ith formula. n_i is number of satisfied atoms and formulas that appeared in the sentence.

Learning the weights of MLN

Given the dependency parse and QLF forms, weights can be learned from data such as the log-likelihood of having QLFs given the dependency trees will be maximized. The form of likelihood is:

$$L_\theta(QLF) = \log \Sigma_L P_\theta(QLF, L)$$

where L is a semantic parse.

Since the summation of all possible semantic parses is infeasible, a set of additional constraints on the form of clauses and also a prior over weights are added to model to make the learning feasible. Their algorithm merges any possible two clusters to form a bigger cluster or create a new cluster if either increases the likelihood.

Finding the Best Semantic Parse

A semantic parse is a partition of atoms of a QLF, assignment of its head and argument form to a cluster. This can be seen as a structure prediction problem. Given the described MLN and its parameters, inference can be done by finding a Maximum A Posterior solution. An exact solution needs a summing over all possible syntactic realizations and possible dependency parses, therefor a greedy algorithm to search for a good solution is proposed by the authors.

3.3.3 Evaluation

Since USP is the first system for unsupervised semantic parsing, it is compared against other models in a Question/Answering task. GENIA is an Q/A annotated dataset in medical domain and USP has been shown to find the answers with 88% precision which is about 10% higher than TextRunner and more than 30% better than DIRT. It is worthwhile to mention that GENIA is a rather small dataset.

There are two major problems observed by the authors themselves and others for USP.

- Same as the other models that we have already seen, it doesn't address the separation of antonym relations
- It is mentioned in [12] and [21] that USP suffers from using a large memory and therefore low ability to scale to bigger dataset.

Two modifications for USP runtime is proposed in [23] and [21]. In [21] it is proposed to use directed graphical models instead of undirected graphs(MLN) to increase the learning and inference speed. The result is slightly worse than the original model but it performs faster than USP. In [23] a new representation of the model is given in the framework of deep learning which yields the same performance but with faster learning algorithm.

In the next section we will see another model which shows improvement to DIRT, TextRunner and USP and is much more scalable than the former.

3.4 Rel-LDA & Type-LDA

In this section we will review the method proposed for relation discovery by Yao et. al. [24] The method they proposed is an unsupervised method using generative models which is based on modification of well-known model, Latent Dirichlet Allocation. They have invented three similar models which we will go through them in following. We elaborate on input of the three models and then we describe their generative story and at the end we will finish this part by discussing about model evaluation.

3.4.1 Input format and preprocessing

Authors have chosen a subset of New York Times articles from year 2000 to 2007 which some of its parts with abnormal style of writing like obituary content are filtered out. They have done several preprocessing steps:

1. Tokenization
2. Sentence split
3. POS tagging
4. Named entity tagging
5. Dependency parsing

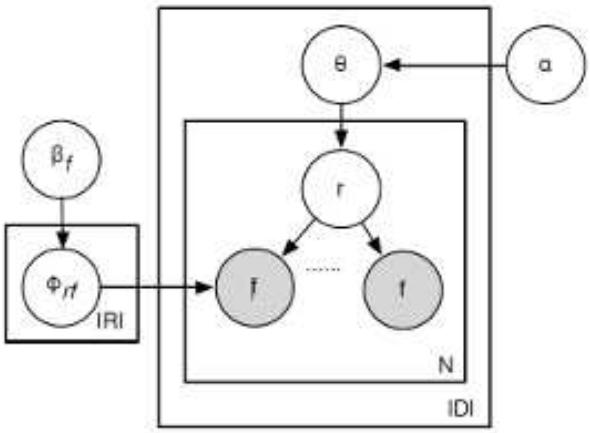
Among all dependency paths they are just interested in those which link two named entities. Authors assume that realization of one relation is available in this path most likely as a verb which relates two mentioned named entities. These types of paths are collected and based on some conditions that we have already discussed in Section 3.1.2 , non-important links are filtered out. Finally in this phase, 2.5 million dependency paths were collected which will serve as an input to the models that we describe in the next subsection.

3.4.2 Model Description

All the three models in this work are generative models. Generative models are jointly modeling hidden variables and observable variables and are in contrast to discriminative models. For more information on generative models please see [25].

Rel-LDA is the first model proposed in [24]. In this model, features, f, are generated from a distribution , Φ_{rf}. Each type of relation is represented with a binary indicator variable, r. Each document is a mixture of few relations and relations are generated from a multinomial distribution Θ_{doc}. Θ_{doc} is a distribution of relations for a specific document and can be shown as $P(r|doc)$. By choosing a right prior distribution for Θ and Φ we can impose an assumption that any document contains a few number of relation types. Another advantage of having prior is that it helps to avoid overfitting. Dirichlet prior with small parameter α is chosen for this purpose in the paper. For more on the role of priors and specially Dirichlet prior please read [26] and [27] . Rel-LDA is described with a graphical model in Figure 3.1 .

Figure 3.1: Rel-LDA



Shaded circles are showing the observable variables and other circles are hidden variables except the priors which are set by ourselves. The direction of arrow shows the dependency in this sense that destination node (variable) is dependent on source node.

Features or observable variables in Rel-LDA are:

- The dependency path
- Source of the path (the first named entity)
- Destination of the path (the second named entity)

An exact sampling method called *variational inference* is used by authors to compute the posterior distribution of the model, $P(r|f)$. After learning, we will have clusters of relation types which each instance of a specific cluster is a relation that is supposed to be semantically equivalent to the other members of this cluster.

In the second model, **Rel-LDA1**,The only difference is that more features are used in the learning procedure. The idea is that using more features could be useful as tie-breakers and also discriminators between instances to make new clusters. With direct reference to the paper, the authors believe that more features lead to better refinement of clusters. We will see later that this assumption holds in practice. The new features that they have introduced in this new model are:

- Trigger: Any word in a dependency path except than stop words.
- Part of speech sequence: The sequence of POS tags of a dependency path.
- Named entity pair: The type of source and destination named entity.
- Syntactic pair: The type of dependency edges connecting source and destination to the head.

Among all the newly introduced features, NE pair is the most interesting one which leads to a significant observation that the third model will try to address that. Authors observed that Rel-LDA will put these three relations in one cluster because the second argument of all of them is location:

1. *X was born in Y*
2. *X lives in Y*
3. *X, a company in Y*

While the first argument in the first two relations refers to a PER, the first argument of the third relation is an instance of ORG type. By using NER pairs as feature we can split this cluster to two clusters with respect to basic types of NE in relations arguments. This observation leads us to invest more on type identification of arguments to have more pure clusters and is the main focus of the third model, **Type-LDA**.

Selectional preferences of a relation are constraint over possible arguments for the relation. Basically, any relation only accepts a few number of entity types and this could be an important constraint for inducing relations. Relations of each cluster should have similar selectional preferences and accept same entity types. Type-LDA is proposed for this reason, it will induce entity types and relation clusters jointly to benefit more from selectional preferences of relations. As we have already mentioned in Section 3.1.4, this idea have been used in [8] .

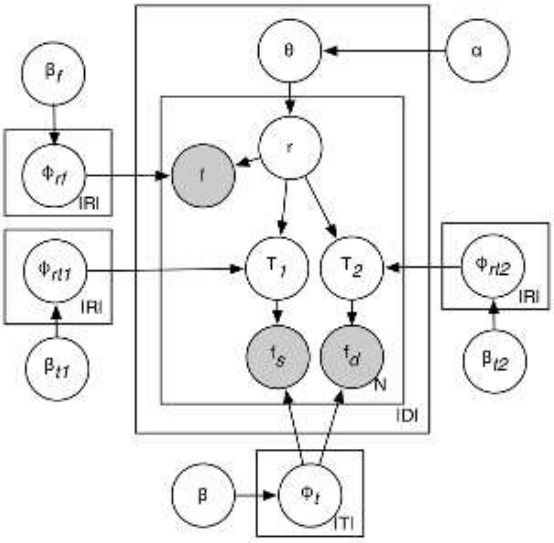
The generative model is modified in a way that features of arguments (source and destination) will be generated from two new distributions, T_1 & T_2 which are modeling entity types. The graphical model is shown in Figure 3.2 . This model not only clusters relations but also clusters entities to entity clusters.

For their experiments on all three models they have set the number of relation clusters to 100 and for Type-LDA they used 50 entity clusters. Choosing other numbers of relation clusters in the range of 50 to 200 is shown to be not very significant.

3.4.3 Evaluation

They have used human judgments to measure their models precision. Humans are asked to label 50 instances in each relation cluster and in order to measure the recall, induced relations are compared against Freebase. Rel-LDA1 and Type-LDA which are using selectional preferences features

Figure 3.2: Type-LDA



are performing better than Rel-LDA. Rel-LDA1 is shown to have the best precision among other models. Rel-LDA and Type-LDA have been also compared against USP and have been shown to have superior performance both in scalability and F-measure.

Like most of the other models, antonymy is not handled in their model and therefor for example they have *X was born in Y* and *X die in Y* in one cluster.

Entity clusters also sometimes suffer from high-frequency or low-frequency words and for example, ‘New York‘ , is in the same cluster as other publications like ‘New York Times‘, ‘Vanity Fair‘ and . . .

Yao et al. have shown that they have induced relations in different granularity from Freebase and reported some relations which are not mentioned in Freebase but positively hold. For example, Freebase relation *worksFor* is subsumed with more relations each indicates a different role of employment relation. *leaderOf* and *editorOf* are such examples. This will boost the idea of inducing hierarchical relations which will be discussed more in the last chapter. A similar approach which tries to induce hierarchical structures is [28] .

Chapter 4

Word Embeddings

In this chapter, we will define and justify the task of *Representation Learning* and we will see different families of methods for inducing word representation and its application in NLP.

4.1 Representation Learning

In machine learning specially in industry, most of the labor is dedicated to *Feature Engineering*. Extracting informative features is the crucial part of most supervised methods and it is done mostly manually. While many different applications share common learning models and classifiers, the difference in performance of competing methods mostly goes to the data representation and hand-crafted features that they use. This observation reveals an important weakness in current models, namely their inability to extract and organize discriminative features from data. Representation learning is an umberella term for a family of unsupervised methods to learn features from data. Most of recent works on the application of this idea in NLP focus on inducing word representations. *Word representation* is a mathematical object, usually a vector, which each dimension in this vector represents a grammatical or semantical feature to identify this word and is induced automatically from data [29]. Recently, it has been shown in [29] and [30] that using induced feautres can be helpful to improve state-of-the-art methods in different NLP tasks. It seems that relation extraction can also benefit from such features since similar tasks like semantic role labeling has been shown to benefit from induced word representations. In section 5.2 we will describe one possible way of incorporating this idea in to our task. In the next two sections, two major families of representations will be shortly reviewed.

4.2 Distributional Representation

In distributional semantics, the meaning of a word is expressed by the context that it appears in it [10]. Features that are used to represent the meaning of a word are other words in its neighborhood as it is so called the context. In some approaches like LDA and latent semantic analysis (LSA), the context is defined in the scope of a document rather than a window around a word. To represent word meanings in via distributional approach, one should start from count matrix (or zero-one co-occurence matrix) which each row represents a word and each column is a context. The representation can be limited to raw usage of the very same matrix or some transforms like

tf-idf will be applied first. A further analysis over this matrix to extract more meaningful features is applying dimensionality reduction methods or clustering models to induce latent distributional representations. A similar clustering method to k-means is used in [15] to represent phrase and word meanings and brown clustering algorithm [14] has been shown to have impact on near to state-of-the-art NLP tasks [29].

4.3 Distributed Representation

Distributed representation has been introduced in the literature for the first time in [31] where Bengio et al. introduced a first language model based on deep learning methods[32]. Deep learning is learning through several layers of neural networks which each layer is responsible to learn a different concept and each concecpt is built over other more abstract concepts. In the deep learning society, any word representation that is induced with a neural network is called *Word Embedding*. In contrast to raw count matrix in distributional representations, word embeddings are low-dimensional, dense and real-valued vectors. The term, **'Distributed'**, in this context refers to the fact that exponential number of objects (clusters) can be modeled by word embeddings. Here we will see two famous models to induce for such representations. One family will use n-grams to learn word representation jointly with a language model and the other family learns the embedding from structured resources.

4.3.1 Neural Language Models

In [33], Weston and Collobert use a non-probabilistic and discriminative model to jointly learn word embeddings and a language model that can separate plausible n-grams from noisy ones. For each word in a n-gram, they combine the word embeddings and use it as positive example. They put noise in the n-gram to make negative examples and then train a neural network to learn to classify postive labels from negative ones. The parameters of neural network (neural language model) and word embedding values will be learned jointly by an optimization method called *Stochastic Gradient Descent* [34].

A hierarchical dsitributed language model (HLBL) proposed by Mnih and Hinton in [35] is another influential work on word embeddings. In this model a probabilistic linear neural network(LBL) will be trained to combine word embeddings in first $n-1$ words of a n-gram to predict the n_th word.

Weston-Collobert model and HLBL by Mnih and Hinton are evaluated in [29] in two NLP tasks: chunking and named entity recognition. With using word embeddings from these models combined with hand-crafted features, the performance of both tasks are shown to be improved.

4.3.2 Representation Learning from Knowledge Bases

Bordes et al. in [36] and [37] have attempted to use deep learning to induce word representations from lexical resources such as WordNet and knowledge bases (KB) like Freebase. In Freebase for example, each named entity is related to another entity by an instance of a specific type of relation. In [36], each entity is represented as a vector and each relation is decomposed to two matrices. Each of these matrices transform left and right-hand-side entities to a semantic space. Similarity

of transformed entities indicates that the relation holds between the entities. A prediction task is defined to evaluate the embeddings. Given a relation and one of the entities, the task is to predict the missing entity. The high accuracy (99.2%) of the model on prediciton of training data shows that learnt representation highly captures attributes of the entities and relations in Freebase.

Chapter 5

Thesis work plot

5.1 Problem Identification

Based on what we have seen in recent works, we can now give a list of vital attributes that a state-of-the-art model for extracting relations from open text should be able to carry out. The author will use these facts to suggest a list of possible improvements in the next section. Modeling all of these factors in a joint model is the necessary step to push forward the previous works.

The number of relations and entities is an unknown parameter.
The model can not be confined to a limited set of relations or entities. Being able to extract relations in open domain text is the first and (most likely) a trivial attribute of the model. More non-trivial feature of the model should be its ability to extract as many relations as there are in text. Giving this freedom about the model complexity to have no assumption about the exact number of entities and relations is suggested by the applicant to be beneficial and is also supported in the literature. [4] [17]

Relations may not be expressed explicitly in text.
Relation extraction task is definitely more than finding paraphrases. The model should be able to handle long-distance relations among entities as well as hidden semantic indications of a relation. [19]

Relations and entities have their inner organization and types.
It is shown by several recent works that relations of relations play a substantial role in identifying relations. Relations and entities belong to a hierarchy of types and therefore the constraints they put on each other should be learned as well.[12] [28] [5]

Using KB is necessary but not enough.
There are more relations among entities than what is collected in Knowledge Bases e.g. Freebase. At the same time, it is statistically shown that a supervision from such resources strongly contributes to convergence of any model to a better objective configurations.[12] [4]

Relations and entities are sharing information within each other.
Relations and entities should be learned jointly since they share same explanatory factors

(hidden variables) . Meaning of an entity can be learned from its relation to other entities and same argument holds among relations. Basically, the model should be able to carry out multi-task learning. [12]

5.2 Our Contribution

The main cotribution of the thesis will be:

1. Inducing distributed representation for words and named entities jointly from different resources (corpus, lexical resources and knowledge bases) based on models described in chapter 4 with focus on the section 4.3.2 (Bordes et al. [36])

2. Incorporating learnt word embeddings to distant supervision method described in the section 2.1. (Mintz et al. [4])

3. Based on if the results from previous phases are promising or not, the word embeddings will be also used within Yao et al. method [12] as it is described in the section 3.4.

Currently a system is implemented (1) to extract relations from different structured data like Freebase, WordNet and FramNet and (2) induce a distributed representation for both entities and relations. The system has the possibility of using GPU to do linear algebra calculations. Preliminary results shows the strength of representations learnt jointly from diferent resources. Map-Reduce algorithms (in MongoDB) are also used to boost runtime of dataset preparation in the pipeline. For writing the dissertation, correspondent chapters are being written in parallel to the progress of research.

5.3 Plan for Completion of The Research

Table 5.1 shows my plan for completion of the research.

Timeline	Work	Progress
	literature review	completed
	preparing dataset and code to organize and access too dataset	completed
	distributed representation learning	completed
Aug.	incorporating features to distant supervision model and evaluation	ongoing
Sep.	revising resources fo representation learning phase and re-running experiments with new features	
Oct.	Thesis defense	

Table 5.1: Plan for completion of our research

Bibliography

[1] I. Androutsopoulos and P. Malakasiotis, "A survey of paraphrasing and textual entailment methods," *arXiv preprint arXiv:0912.3747*, vol. 38, pp. 135–187, 2009.

[2] D. Lin and P. Pantel, "DIRT - Discovery of Inference Rules from Text," in *. . . conference on Knowledge discovery and data mining*, pp. 323–328, Association for Computing Machinery, 2001.

[3] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni, "Open Information Extraction from the Web," in *IJCAI*, pp. 2670–2676, 2007.

[4] M. Mintz, S. Bills, R. Snow, and D. Jurafsky, "Distant supervision for relation extraction without labeled data," in *of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, vol. 2, pp. 1003 – 1011, 2009.

[5] N. Nakashole, G. Weikum, and F. Suchanek, "PATTY: a taxonomy of relational patterns with semantic types," *EMNLP*, pp. 1135–1145, 2012.

[6] E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng, "Improving word representations via global context and multiple word prototypes," in *ACL*, pp. 873–882, Association for Computational Linguistics, July 2012.

[7] R. Bhagat, P. Pantel, E. Hovy, and M. Rey, "LEDIR: An Unsupervised Algorithm for Learning Directionality of Inference Rules.," in *EMNLP-CoNLL*, no. June, pp. 161–170, 2007.

[8] P. Pantel and R. Bhagat, "ISP: Learning inferential selectional preferences," *ACL*, no. April, pp. 564–571, 2007.

[9] F. Pereira, N. Tishby, and L. Lee, "Distributional clustering of English words," in *ACL*, pp. 183–190, 1993.

[10] Z. Harris, *Distributional structure*. Springer Netherlands, 1981.

[11] M. Mechura, *Selectional Preferences, Corpora and Ontologies*. PhD thesis, Trinity College, University of Dublin, 2008.

[12] L. Yao, A. Haghighi, S. Riedel, and A. McCallum, "Structured relation discovery using generative models," in *EMNLP*, pp. 1456–1466, 2011.

[13] P. Pantel and D. Lin, "Discovering word senses from text," in *Proceedings of the eighth ACM SIGKDD international . . .* , pp. 613–619, 2002.

[14] P. Brown, P. Desouza, and R. Mercer, "Class-based n-gram models of natural language," *Computational Linguistics*, vol. 4, pp. 467–479, 1992.

[15] D. Lin and X. Wu, "Phrase clustering for discriminative learning," in *ACL-AFNLP*, pp. 1030–1038, 2009.

[16] L. Kotlerman, I. Dagan, I. Szpektor, and M. Zhitomirsky-Geffet, "Directional distributional similarity for lexical inference," *Natural Language Engineering*, vol. 16, pp. 359–389, Oct. 2010.

[17] A. Yates, M. Cafarella, and M. Banko, "TextRunner: open information extraction on the web," in *ACL*, no. April, pp. 25–26, 2007.

[18] M. Banko, *Open Information Extraction from the Web.* PhD thesis, University of Washingotn, 2009.

[19] H. Poon and P. Domingos, "Unsupervised semantic parsing," in *EMNLP*, vol. 1, (Morristown, NJ, USA), pp. 1–10, Association for Computational Linguistics, 2009.

[20] M. Richardson and P. Domingos, "Markov logic networks," *Machine learning*, pp. 107–136, 2006.

[21] I. Titov and A. Klementiev, "A Bayesian model for unsupervised semantic parsing," *ACL*, pp. 1445–1455, 2011.

[22] D. Lin and P. Pantel, "Discovery of inference rules for question-answering," *Natural Language Engineering*, vol. 7, Feb. 2002.

[23] H. Poon and P. Domingos, "Deep Learning for Semantic Parsing," tech. rep., 2013.

[24] S. Riedel and L. Yao, "Relation Extraction with Matrix Factorization and Universal Schemas," *Proceedings of NAACL- ...*, no. June, pp. 74–84, 2013.

[25] K. P. Murphy, *Machine Learning A probabilistic perspective*. The MIT Press, 2012.

[26] Y. Teh, "Dirichlet Processes: Tutorial and Practical Course," *Machine Learning Summer School*, no. August, 2007.

[27] S. J. Gershman and D. M. Blei, "A tutorial on Bayesian nonparametric models," *Journal of Mathematical Psychology*, vol. 56, pp. 1–12, Feb. 2012.

[28] E. Alfonseca and K. Filippova, "Pattern learning for relation extraction with a hierarchical topic model," in *ACL*, no. July, pp. 54–59, 2012.

[29] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: a simple and general method for semi-supervised learning," *ACL*, pp. 384–394, July 2010.

[30] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural Language Processing (almost) from Scratch," *Machine Learning Research*, vol. 12, pp. 2493–2537, Mar. 2011.

[31] Y. Bengio and R. Ducharme, "A neural probabilistic language model," *The Journal of Machine ...*, vol. 3, pp. 1137–1155, 2003.

[32] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, 2009.

[33] R. Collobert and J. Weston, "A unified architecture for natural language processing: deep neural networks with multitask learning," *... international conference on Machine learning*, 2008.

[34] L. Bottou, "Large-scale machine learning with stochastic gradient descent," *Proceedings of COMPSTAT'2010*, no. x, 2010.

[35] A. Mnih and G. Hinton, "A scalable hierarchical distributed language model," *Advances in neural information processing systems*, pp. 1–8, 2009.

[36] A. Bordes, J. Weston, R. Collobert, and Y. Bengio, "Learning structured embeddings of knowledge bases," *Proceedings of the 25th ...*, no. Bengio, pp. 301–306, 2011.

[37] A. Bordes and X. Glorot, "Joint learning of words and meaning representations for open-text semantic parsing," *... of Machine Learning ...*, vol. 22, 2012.