Integrating Word Embedding Offsets into the Espresso System for Part-Whole Relation Extraction

Van-Thuy Phi
Nara Institute of Science and Technology
Computational Linguistics Laboratory
8916-5 Takayama, Ikoma, Nara Japan
phi.thuy.ph8@is.naist.jp

Yuji Matsumoto
Nara Institute of Science and Technology
Computational Linguistics Laboratory
8916-5 Takayama, Ikoma, Nara Japan
matsu@is.naist.jp

Abstract

Part-whole relation, or *meronymy* plays an important role in many domains. Among approaches to addressing the part-whole relation extraction task, the Espresso bootstrapping algorithm has proved to be effective by significantly improving recall while keeping high precision. In this paper, we first investigate the effect of using fine-grained subtypes and careful seed selection step on the performance of extracting part-whole relation. Our multi-task learning and careful seed selection were major factors for achieving higher precision. Then, we improve the Espresso bootstrapping algorithm for part-whole relation extraction task by integrating word embedding approach into its iterations. The key idea of our approach is utilizing an additional ranker component, namely *Similarity Ranker* in the Instances Extraction phase of the Espresso system. This ranker component uses embedding offset information between instance pairs of part-whole relation. The experiments show that our proposed system achieved a precision of 84.9% for harvesting instances of the part-whole relation, and outperformed the original Espresso system.

1 Introduction

A major information extraction task is relation extraction, whose goal is to predict semantic relations between entities or events expressed in the structured or unstructured text. There are several different kinds of semantic relations that connect two or more concepts. Among those semantic relations, *part-whole* relation, or *meronymy* plays an important role in many domains and applications. Extracting part-whole relations in the text is also a crucial step towards applications in several fields, such as Information Extraction, Web/Text Mining and Ontology Building. Such systems often need to recognize part-whole relations for better understanding semantic relationships between concepts. Therefore, in our research, we aim at extraction of part-whole relation. We are interested in relations between entities in the newswire domain.

Among approaches to addressing the part-whole relation extraction problem, the Espresso bootstrapping algorithm (Pantel and Pennacchiotti, 2006) has proved to be effective by significantly improving recall while keeping high precision. Espresso is a well-known bootstrapping algorithm that uses a set of seed instances to induce extraction patterns for the target relation and then acquire new instances in an iterative bootstrapping manner. Nevertheless, it has a bias, called *semantic drift,* to select unrelated instances if a polysemous instance has been extracted as the iteration proceeds.

Recently, Mikolov et al. (2013) have introduced the skip-gram text modeling architecture. It has been shown efficiently to learn meaningful distributed representations of terms from unannotated text. The vectors have some of the semantic characteristics in element-wise addition and subtraction. For example, the result of a vector subtraction vec("*Madrid*") - vec("*Spain*") is close to vector subtraction vec("*Paris*") - vec("*France*"). That is an example of *the country to capital city* relationship. It indicates that the embedding offsets represent the shared semantic relation between the two

word pairs.

The example above raises a question whether we can apply those semantic characteristics for part-whole relation? In this paper, we would like to address two important questions:

1. Is Word2Vec model appropriate for pairs of part-whole relation? That is, we investigate typical instances of part-whole relation to measure their similarities by cosine distance.

2. How to integrate Word2vec model efficiently into the Espresso system?

The details of our contribution are as follows:

- We apply the Espresso bootstrapping algorithm for part-whole relation, and study the effect of using careful seed sets and fine-grained subtypes on the performance of extracting part-whole relation.

- We investigate similarities between two instances of the part-whole relation. Then, we integrate an additional ranker component into the Espresso bootstrapping algorithm to improve the performance when using iterative bootstrapping process and reduce semantic drift phenomenon for extracting part-whole relation. That ranker component uses similarity score between embedding offsets to keep similar instances in each iteration.

To the best of our knowledge, ours is the first study to integrate word embedding approach in a bootstrapping algorithm for part-whole relation extraction task.

2 Related Work

In this section, we provide an overview of previous studies related to relation extraction problem. Approaches for relation extraction are divided into three classes: rule-based methods, supervised methods, and semi-supervised and unsupervised methods.

The first approach is usually used in domain-specific tasks. Systems which use this one rely on some linguistic rules to capture patterns in text. Patterns are manually defined for a particular semantic relation. Hearst (1992) describes the usage of lexico-syntactic patterns for extracting "*is-a*" relations, for example, "*such as*", "*including*", "*especially*", etc. However, the author notes that this method does not work well for some other kinds of relations, for example, meronymy (part-whole relation).

Supervised approaches for relation extraction are divided into feature-based methods and kernel methods. In feature-based methods, syntactic and semantic features can be extracted from the text given a set of positive and negative relation examples. Kambhatla (2004) employs Maximum Entropy model to combine diverse lexical, syntactic and semantic features derived from the text in relation extraction. Zhou et al. (2005) explore various features in relation extraction using Support Vector Machine (SVM). They report that chunking information contributes to most of the performance improvement from the syntactic aspect. In kernel methods, a kernel is used to calculate the similarity between two objects. Kernel-based relation extraction methods were first attempted by Zelenco et al. (2003). They devise contiguous subtree kernels and sparse subtree kernels for recursively measuring the similarity of two parse trees to apply them to binary relation extraction. Bunescu and Mooney (2005) present a different kernel based on the shortest path between two relation entities in the dependency graph. Zhao and Grishman (2005) define a feature-based composite kernel to integrate diverse features for relation extraction. Girju et al. (2006) present a domain independent approach for the automatic extraction of part-whole relation. Their method discovers the lexico-syntactic patterns and the semantic classification rules needed for the disambiguation of these patterns.

Annotated data is lacking and expensive to create in large quantities, therefore making semi-supervised or unsupervised techniques is desirable. Early semi-supervised learning and bootstrapping methods are DIPRE (Brin, 1999) and Snowball (Agichtein and Gravano, 2000). They rely on a few learning collections for making the use of bootstrapping for gathering syntactic patterns that express relations between the two entities in a large web-based text corpus. Ittoo and Bouma (2013) use a minimally-supervised approach to extract part-whole relations from text iteratively. Wikipedia is

the knowledge base, from which they first select a seed set of reliable patterns. Other works include Espresso bootstrapping algorithm (Pantel and Pennacchiotti, 2006), TextRunner (Yates et al., 2007).

3 Part-Whole Relation Extraction Task

The part-whole relation is a relationship between the parts of things and the wholes which comprise them. We are interested in relations between two entities in the English newswire domain. If the entities X and Y are related in such a manner that X is one of the constituents of Y, then there is a part-whole relation between X and Y. In the context of knowledge representation and ontologies, the study of part-whole relations has three axioms (Rector et al., 2005):

- Transitive - "*parts of parts are parts of the whole*" - If A is part of B and B is part of C, then A is part of C.
- Reflexive - "*Everything is part of itself*" - A is part of A.
- Antisymmetric - "*Nothing is a proper part of its parts*" - if A is part of B and A != B then B is not part of A.

Given a piece of text that contains two entity mentions, the goal of part-whole relation extraction task is to decide whether that text contains part-whole relation between the two entities. Let the triple *T = (arg1, P, arg2)* denote a part-whole relation, where *arg1* and *arg2* are two entities contained in text, and:

- *P* is a lexical pattern,
- *(arg1, arg2)* is an instance, where *arg1* represents the part and *arg2* represents the whole, or vice versa.

4 Our Approach

One problem of supervised approach is that it requires large amounts of annotated data. Therefore, we choose a bootstrapping method. In this approach, we only need a few high-precision examples as the input.

In this section, we first describe how we apply the Espresso bootstrapping algorithm for part-whole relation. We focus on seed selection since it is one of the most important steps in bootstrapping algorithms. Then, we propose an effective method for integrating word embedding approach into the Espresso system, after similarity between instances of part-whole relation were investigated.

4.1 The Espresso Bootstrapping Algorithm for Part-Whole Relation

Currently, Espresso (Pantel and Pennacchiotti, 2006) is well known as an efficient algorithm for extracting pairs of entities in a particular relationship. It is a pattern-based and minimally supervised bootstrapping algorithm of extracting lexical-semantic relations. It takes as input a few seed instances and iteratively learns surface patterns to acquire more instances. The Espresso bootstrapping algorithm iterates between the following 3 phases:

1. Pattern Induction: Induce a set of patterns *P* that connects the seed instances in a given corpus. Patterns may be surface text patterns or lexico-syntactic patterns.

2. Pattern Ranking/Selection: Create a pattern ranker, and select the top-k patterns by pattern reliability score. The reliability of a pattern p, $r_\pi(p)$ is average strength of association across input i in the set of instances *I*, weighted by the reliability of each instance i:

$$r_\pi(p) = \frac{\sum_{i \in I}(\frac{pmi(i,\ p)}{max_{pmi}} * r_\iota(i))}{|I|} \quad (1)$$

where $r_\iota(i)$ is the reliability of instance i (defined below) and max_{pmi} is the maximum pointwise mutual information between all patterns and all instances. The value of $r_\pi(p)$ ranges from [0, 1], and the reliability of the manually provided seed instances are $r_\iota(i) = 1$. The pointwise mutual information (PMI) between instance *i = (x, y)* and pattern *p* is measured using the following formula:[1]

$$pmi(i,\ p) = log\frac{|x,\ p,\ y|}{|x,\ *,\ y||*,\ p,\ *|}$$

where $|x,\ p,\ y|$ is the frequency of pattern p linked with the instance (x, y). Then, $pmi(i,\ p)$ is multiplied with the discounting factor used in (Pantel and Ravichandran, 2004) to mitigate a bias towards infrequent events.

[1] In that formula, the asterisk (*) represents a wildcard.

3. Instance Extraction: Retrieve from the corpus the set of instances *I* that match any of the patterns in *P*, then create an instance ranker, and select the top-m instances by the instance reliability score. Calculating the reliability of an instance is similar to calculating the reliability of a pattern. The reliability of an instance i, $r_{\iota(i)}$, is defined as:

$$r_{\iota(i)} = \frac{\sum_{p \in P} (\frac{pmi(i,\,p)}{max_{pmi}} * r_{\pi}(p))}{|P|} \quad (2)$$

A reliable instance should be highly associated with as many reliable patterns as possible. Espresso iterates the above three phases several times until stopping criteria are met.

Unfortunately, like other bootstrapping algorithms, Espresso is prone to *semantic drift*. This phenomenon often occurs when ambiguous or unrelated terms and/or patterns are acquired and then dominate the iterative process (Curran et al., 2007). Ranking patterns and instances by their reliability is an effective way to avoid semantic drift (Equations (1) and (2)). However, bootstrapping is indeed a seed set expansion, therefore selecting good seeds is the most important step to reduce semantic drift. Moreover, semantic drift still occurs in later iterations if the seed set is not good.

To cover the variety of part-whole relation, we classify its subtypes systematically before the seed selection step. There are several subtypes of part-whole relation mentioned in previous ontological studies. In WordNet, part-whole relation is classified into 3 basic subtypes: *Stuff-of*, *Member-of*, and *Part-of*. Chaffin et al. (1988) defined 7 subtypes of part-whole relation, namely *Component-Object, Member-Collection, Portion-Mass, Stuff-Object, Feature-Activity, Place-Area*, and *Phase-Process*. In recent research, Keet and Artale (2008) identified 8 subtypes of part-whole relation. From their taxonomy, Mereological (or transitive) relations include *Involved-In, Located-In, Contained-In*, and *Structured-Part-Of*; while Meronymic (or intransitive) relations consist of *Member-Of, Constituted-Of*, and *Sub-Quantity-Of*.

We reorganize subtypes of part-whole relation as follows:

1. **Component-Of**: or Part-Of; between integrals and their functional components, e.g. *(finger, hand)*.

2. **Member-Of**: between a physical object (or role) and an aggregation (team or organization, etc.), e.g. *(player, team)*.

3. **Portion-Of**: or Sub-Quantity-Of; between amounts of matter or units, e.g. *(oxygen, water)*.

4. **Stuff-Of**: or Substance-Of, or Constituted-Of; between a physical object and an amount of matter, e.g. *(steel, car)*.

5. **Located-In**: between an entity and its 2-dimensional region, e.g. *(Tokyo, Japan)*.

6. **Contained-In**: between an entity and its 3-dimensional region, e.g. *(chip, processor)*.

7. **Phase-Of**: or Involved-In, or Feature-Activity; between a phase and a process, e.g. *(chewing, eating)*.

8. **Participates-In**: between an entity and a process, e.g. *(enzyme, reaction)*.

Basically, our classification is similar to Keet and Artale's taxonomy, which also contains subtypes in other ontological studies. However, we normalize name of subtypes, to find a coherent name set over studies, for example, in WordNet or in (Chaffin et al., 1988).

Seed Selection: We use the following strategy to select seeds for part-whole relation:

- First, for each subtype, we find unambiguous lexical patterns that always convey a part-whole relation, for example, "*a component of*", "*consist of*", etc.
- Then, we search for instances pairs (e.g. Wikipedia dataset) connected by patterns above.
- We select at most 5 instances for each subtype. The most frequent pairs are selected, and we try to select pairs that do not overlap each other. Also, we try not to extract the same instances crossing over the subtypes.

Then, we use simultaneously all the instances to perform the Espresso bootstrapping algorithm for part-whole relation extraction task.

4.2 Similarity Between Instances of Part-Whole Relation

One interesting feature of Word2vec model is that the vectors conserve some of the semantic charac-

teristics in operations regarding the semantic information that they capture, for example, *the country to capital city relationship*: vec("*Madrid*") - vec("*Spain*") ≈ vec("*Paris*") - vec("*France*"), or *gender relationship*: vec("*woman*") - vec("*man*") ≈ vec("*queen*") - vec("*king*"). They indicate that the embedding offsets actually represent the shared semantic relation between the two word pairs.

Can we apply those characteristics for part-whole relation, for example, vec("*pedal*") - vec("*bike*") ≈ vec("*engine*") - vec("*car*")? To answer this question, we investigate typical instances of part-whole relation to measure their similarities by cosine distance. Our calculation is mainly based on the recently proposed Word2vec model. We use *word2vec* tool, and pre-trained vectors published by Google.[2] The model contains 300-dimensional vectors for 3 million words and phrases.

The Word2vec model gives us a vector for each word. To measure the similarity between two instances of part-whole relation, for example, *(pedal, bike)* and *(engine, car)*, we first compute the embedding offsets between two terms in instances, that is, calculate vec("*pedal*") - vec("*bike*") and vec("*engine*") - vec("*car*"). Then, we calculate the cosine distance between those embedding offsets. Here, the bigger the cosine value is, the more similar the two instances will be.

Two instances	**Similarity by cosine distance**
(husband, marriage) & *(wife, marriage)*	0.852828
(Paris, France) & *(Beijing, China)*	0.536129
(pedal, bike) & *(engine, car)*	0.347589

Table 1: Similarities between two instances of part-whole relation

In Table 1, we show the similarities between two instances of the part-whole relation. The part-whole relation is a combination of several subtypes; therefore, it is more complicated than other semantic relations. From the results in Table 1, we can see that instances of part-whole relation are quite similar by cosine distance: vec("*husband*") - vec("*marriage*") ≈ vec("*wife*") - vec("*marriage*"); vec("*pedal*") - vec("*bike*") ≈ vec("*engine*") - vec("*car*"), etc. That means the instance *(husband, marriage)* is close to the instance *(wife, marriage)*, the instance *(pedal, bike)* is close to the instance *(engine, car)*, etc., in semantic space. Therefore, we can leverage such characteristic to apply for part-whole relation extraction task.

4.3 Integrating Word Embedding Offsets into the Espresso Bootstrapping Algorithm

In the Espresso bootstrapping algorithm, ranking instances in Instance Extraction phase is very important. The Espresso system creates an instance ranker to keep only high-confidence instances at this phase, as they are used as seed instances for the next iteration.

By using word embedding approach, our purpose is to keep high-precision over iterations for part-whole relation extraction task. The key idea of our approach is utilizing an additional ranker component, namely *Similarity Ranker* in the Instances Extraction phase of the Espresso system. We still use the reliability of instances in the first ranker. Similarity Ranker operates when the instance ranker is completed. It takes top-m instances from the instance ranker as the input and returns top-n instances as the output. This ranker component uses cosine distance as the *similarity score* between instance pairs of part-whole relation to measure their similarities, and remove unrelated instances in each iteration.

The details of our approach are described in the following:

- An additional ranker is used in Instance Extraction phase, namely *Similarity Ranker*.
- We assume that each instance of part-whole relation is represented by the embedding offset between its terms, for example, the instance (*pedal, bike*) corresponds to: vec("*pedal*") - vec("*bike*").
- *Similarity Ranker* takes top-m instances from the instance ranker as the input. For each new instance, our ranker calculates average similarity score between this instance and previous instances. The similarity score of an instance i,

[2]https://code.google.com/p/word2vec/

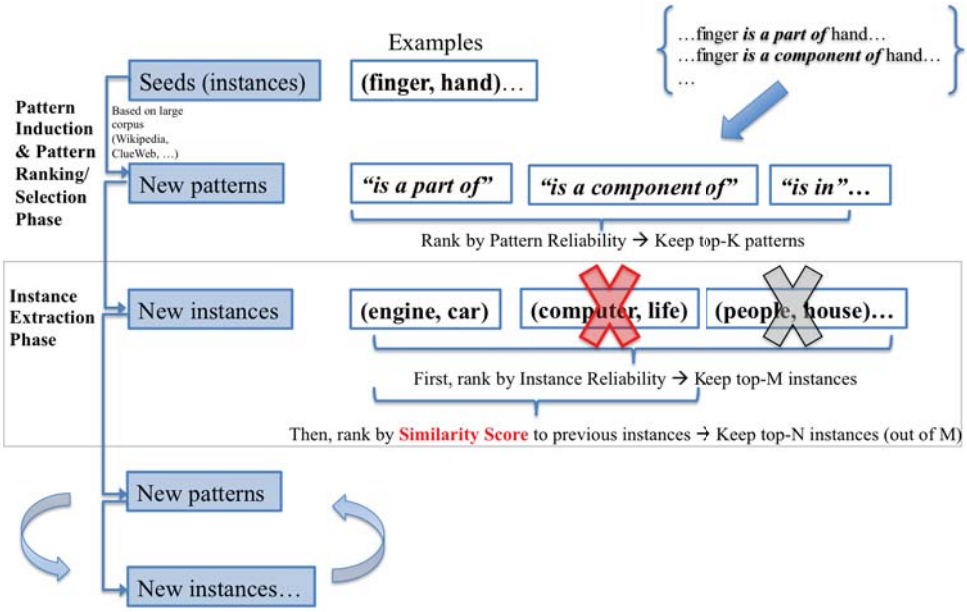

Figure 1: Illustration of our proposed model (Espresso + Word2vec)

SIM(i), is defined as:

$$SIM(i) = \frac{\sum_{j \in I_{Previous}} Cos\_sim(i,j)}{|I_{Previous}|}$$

where $Cos\_sim(i,j)$ is the cosine similarity between two instances, and $I_{Previous}$ is the set of extracted instances.

- If a particular term does not appear in Word2vec model, we use the average vector of its tokens to represent it. For example, if Word2vec model can not recognize the phrase "*Japanese car*", this term corresponds to (vec("*Japanese*") + vec("*car*")) / 2. And in the worst case, if the token "*Japanese*" or the token "*car*" does not appear in Word2vec model, the similarity score of the term "*Japanese car*" is 0.

- Then, m instances are ranked by their similarity score to old instances. Similarity Ranker discards all but the top-n instances as input for the subsequent iteration.

Figure 1 provides an illustration of our proposed model. In Instance Extraction phase, the Espresso bootstrapping algorithm ranks instances first by instance reliability, and removes unrelated pairs, e.g. *(people, house)*. Then, the Similarity Ranker ranks the remaining instances and keeps top-n instances that have the highest similarity score. In our illustration, the instance *(computer, life)* is eliminated to keep high-precision over iterations.

5 Experiments

In this section, we present the results of the Espresso bootstrapping algorithm, and our proposed method (Espresso + Word2vec) on the task of extracting part-whole relations.

Below we describe the systems used in our empirical evaluation of the new proposed model.

- **ESP**: The original *Espresso* algorithm without careful selection of seeds. In this system, part-whole relation is not classified into subtypes and instances are selected randomly.

- **ESP_W2V**: Our proposed method (Espresso + Word2vec) for integrating word embedding approach in the *Espresso* algorithm. Instances are selected randomly. We perform two experiments for two different seed sets, and calculate the average precisions.

- **ESP***: The original *Espresso* algorithm with the careful seed selection step in Section 4.1.

- **ESP*_W2V**: Our proposed method for integrating word embedding approach in the *Espresso* algorithm, with the careful seed selection step.

5.1 Data

We use *ReVerb Extractions 1.1 dataset* as a knowledge-base for our task. ReVerb (Fader et al., 2011) is a program that automatically identifies and extracts binary relationships from English sentences. It contains a set of *(x, r, y)* extraction triples of binary relations (part-whole and other relations), for example, *(bananas, be source of, potassium)*.

A collection of 15 million high-precision ReVerb extractions is available for academic use.[3] The following statistics are the number of distinct tuples, argument strings, and relation strings in the data set: 14,728,268 triples, 2,263,915 instances, and 664,746 patterns.

5.2 Evaluation Method

Our evaluation method is similar to the one introduced by Pantel and Pennacchiotti (2006). We measure the precision of systems by evaluating instances in their output manually. For each instance, we assign a score of 1 if it is correct, 0 if it is incorrect, and 1/2 if it is ambiguous. An example of ambiguous instances is *(energy, economic growth)*. The Cohen's kappa coefficient on our task was 0.689. In total, 4080 instances (6 experiments * 680) were annotated per judge.

5.3 The Espresso System with Careful Seed Selection

As mentioned before, we classify systematically part-whole relation into 8 subtypes before seed selection step to cover the variety of this relation. Then, we select at most 5 instances for each subtype and use this seed set to perform the Espresso bootstrapping algorithm for part-whole relation extraction task. We denote this system by *ESP**.

In total, 35 instances are manually selected as the seed set for our problem. Table 2 lists examples of the seeds.

We experimentally set the number of instances and patterns that the system keeps in each iteration.

[3] http://reverb.cs.washington.edu

Seed	Subtype
(iron, hemoglobin)	Component-Of
(the committee, the president)	Member-Of
(caffeine, coffee)	Portion-Of
(paper, trees)	Stuff-Of
(Shanghai, China)	Located-In
(references, request)	Contained-In
(treatment, surgery)	Phase-Of
(students, class)	Participates-In

Table 2: Sample seeds used for part-whole relation

The parameters for the Espresso bootstrapping algorithm are as follows:

- In the 1st iteration: keep top-10 patterns and top-100 instances.
- In next iterations: keep top-5 patterns and top-20 instances.

We perform experiments with the random selection of seeds (*ESP* system), to compare the effect of the careful seed selection step in the Espresso bootstrapping algorithm. In contrast to the careful seed selection step, we do not separate subtypes of part-whole relation. Each seed set contains 35 instances selected randomly such that those instances always convey part-whole relation. Part-whole relation is single-type in those experiments. The results of the *ESP* system is the average precision when we conduct experiments with two different seed sets of part-whole relation. Each seed set contains 35 instances selected randomly.

The results are reported in Table 3. We evaluated the results after 30 iterations since the precision was nearly constant. At this point, 680 instances were extracted by both systems. The *ESP* system achieved a precision of 74.3%, while the precision of instances harvested by the *ESP** system is 83.3%. We extracted new instances keeping them in different subtypes to be non-overlapping, this can be consider as a kind of multi-task learning. The results show that multi-task learning and seed selection are effective ways to get high precision.

SYSTEM	INSTANCES	CORRECT INSTANCES	AMBIGUOUS INSTANCES	PRECISION
ESP	680	466	65	74.3%[a]
	680	482	60	
ESP_W2V	680	493	54	78.5%[a]
	680	563	45	
ESP*	680	549	35	83.3%
ESP*_W2V	680	554	47	**84.9%**

[a] The result of this system is the average precision when we perform two experiments with two random seed sets of part-whole relation.

Table 3: System performance for part-whole relation extraction task

5.4 Our Proposed System (Espresso + Word2vec)

In this experiment, we present the result of our proposed system for integrating word embedding offsets into the Espresso system. We use the careful seed set in the previous experiment to perform our system for part-whole relation extraction task. The parameters for our proposed system are as follows:

- In the 1st iteration: keep top-10 patterns and top-100 instances (the same as in previous experiment).
- In next iterations: keep top-5 patterns; for instances, keep top-100 by instance reliability, then keep top-20 (out of 100) by similarity score (similar to the previous experiment).

One problem of Word2vec model is that it tends to select very similar instances to given instances, for example, *(team, seven players)*, *(team, six players)*, *(team, three players)*, etc. Therefore, we keep only one of them that has the highest instance reliability in each iteration.

We denote the system in this experiment by *ESP*_W2V*. Then, we evaluated the results after 30 iterations. At this point, 680 instances were extracted. The *ESP*_W2V* system achieved a precision of 84.9%. It outperformed *ESP** system, which is reported 83.3% precision.

We also conduct another experiment for integrating word embedding approach into the Espresso system, without careful seed selection step. That is, we use random seed sets (the same for ESP system) in a new system, which is denoted by *ESP_W2V*. The results of the *ESP_W2V* system is the average precision when we conduct experiments with two different seed sets. From Table 3, we can see that the precision is increased from 74.3% (*ESP* system) to 83.3% (*ESP** system) by using careful selection of seeds, and improved from 78.5% (*ESP_W2V* system) to 84.9% (*ESP*_W2V* system) by integrating word embedding approach into the Espresso system.

All above results showed that our proposed system (Espresso + Word2vec) can keep high-precision over iterations, and outperformed the original Espresso system for part-whole relation extraction.

6 Conclusion and Future Work

In this paper, we considered the part-whole relation extraction task. Subtypes of part-whole relation are separated before seed selection step to cover the variety of this relation. We evaluated 4 systems (6 experiments: 2 for ESP, 2 for ESP_W2V, 1 for ESP* and 1 for ESP*_W2V) to show that by using fine-grained subtypes of part-whole relation and careful seed selection step, the precisions were increased (74.3/83.8 - 78.5/84.9). That is, multi-task learning and seed selection are factors for achieving higher precision. To improve the performance of extracting part-whole relation, we integrated word embedding approach into the Espresso system. Our results illustrated that the proposed model can keep high-precision (84.9%) over iterations, and it outperformed the original Espresso system (74.3/78.5 - 83.3/84.9). We plan to set thresholds in the *Similarity Ranker* to get more accurate results. Our system can be extended for extracting other binary relations.

References

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

Sergey Brin. 1999. Extracting patterns and relations from the world wide web. In *The World Wide Web and Databases*, pages 172–183. Springer.

Razvan C Bunescu and Raymond J Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 724–731. Association for Computational Linguistics.

Roger Chaffin, Douglas J Herrmann, and Morton Winston. 1988. An empirical taxonomy of part-whole relations: Effects of part-whole relation type on relation identification. *Language and Cognitive processes*, 3(1):17–48.

James R Curran, Tara Murphy, and Bernhard Scholz. 2007. Minimising semantic drift with mutual exclusion bootstrapping. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, volume 6.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545. Association for Computational Linguistics.

Roxana Girju, Adriana Badulescu, and Dan Moldovan. 2006. Automatic discovery of part-whole relations. *Computational Linguistics*, 32(1):83–135.

Zhou GuoDong, Su Jian, Zhang Jie, and Zhang Min. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 427–434. Association for Computational Linguistics.

Marti A Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pages 539–545. Association for Computational Linguistics.

Ashwin Ittoo and Gosse Bouma. 2013. Minimally-supervised extraction of domain-specific part–whole relations using wikipedia as knowledge-base. *Data & Knowledge Engineering*, 85:57–79.

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

C Maria Keet and Alessandro Artale. 2008. Representing and reasoning over a taxonomy of part–whole relations. *Applied Ontology*, 3(1):91–110.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 113–120. Association for Computational Linguistics.

Patrick Pantel and Deepak Ravichandran. 2004. Automatically labeling semantic classes. In *HLT-NAACL*, volume 4, pages 321–328.

Alan Rector, Chris Welty, Natasha Noy, and Evan Wallace. 2005. Simple part-whole relations in owl ontologies.

Alexander Yates, Michael Cafarella, Michele Banko, Oren Etzioni, Matthew Broadhead, and Stephen Soderland. 2007. Textrunner: open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 25–26. Association for Computational Linguistics.

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

Shubin Zhao and Ralph Grishman. 2005. Extracting relations with integrated information using kernel methods. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 419–426. Association for Computational Linguistics.