StuffIE: Semantic Tagging of Unlabeled Facets Using Fine-Grained Information Extraction

Radityo Eko Prasojo
Free University of Bozen-Bolzano
Bozen-Bolzano, Italy
RPrasojo@unibz.it

Mouna Kacimi
Free University of Bozen-Bolzano
Bozen-Bolzano, Italy
Mouna.Kacimi@unibz.it

Werner Nutt
Free University of Bozen-Bolzano
Bozen-Bolzano, Italy
Werner.Nutt@unibz.it

ABSTRACT

Recent knowledge extraction methods are moving towards ternary and higher-arity relations to capture more information about binary facts. An example is to include the time, the location, and the duration of a specific fact. These relations can be even more complex to extract in advanced domains such as news, where events typically come with different facets including reasons, consequences, purposes, involved parties, and related events. The main challenge consists in first finding the set of facets related to each fact, and second tagging those facets to the relevant category.

In this paper, we tackle the above problems by proposing StuffIE, a fine-grained information extraction approach which is facet-centric. We exploit the Stanford dependency parsing enhanced by lexical databases such as WordNet to extract nested triple relations. Then, we exploit the syntactical dependencies to semantically tag facets using distant learning based on Oxford dictionary. We have tested the accuracy of the extracted facets and their semantic tags using DUC'04 dataset . The results show the high accuracy and coverage of our approach with respect to ClausIE, OLLIE, SEMAFOR SRL and Illinois SRL .

CCS CONCEPTS

• **Computing methodologies → Natural language processing**; **Information extraction**; **Supervised learning**;

KEYWORDS

Facet extraction; distant learning; semantic labeling

ACM Reference Format:
Radityo Eko Prasojo, Mouna Kacimi, and Werner Nutt. 2018. StuffIE: Semantic Tagging of Unlabeled Facets Using Fine-Grained Information Extraction. In *The 27th ACM International Conference on Information and Knowledge Management (CIKM '18), October 22–26, 2018, Torino, Italy*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3269206.3271812

1 INTRODUCTION

News articles typically report various levels of detail about facts. In Figure 1, we can see three headlines talking about the same fact, *"Trump dumps Rex Tillerson."* However, they provide three different pieces of information. A first headline reports that the firing was *"in a tweet,"* a second one says that it was *"over different views in foreign policy,"* and a third one focuses on what happened after, meaning *"to be replaced by CIA director."* We call *facets* the additional pieces of information, about a given fact. A facet can be verb-less, such as *"in a tweet,"* or verbial like *"to be replaced by CIA director."* In the case of verbial facets, we deal with relations between facts. In the above example, the fact *"to be replaced by CIA director"* is related to the fact *"Trump dumps Rex Tillerson."* Moreover, a facet has a semantic role with respect to its related fact. For example, the facet *"over different views in foreign policy"* is the *reason* of the fact *"Trump dumps Rex Tillerson,"* while the facet *"to be replaced by CIA director"* is the *subsequent*.

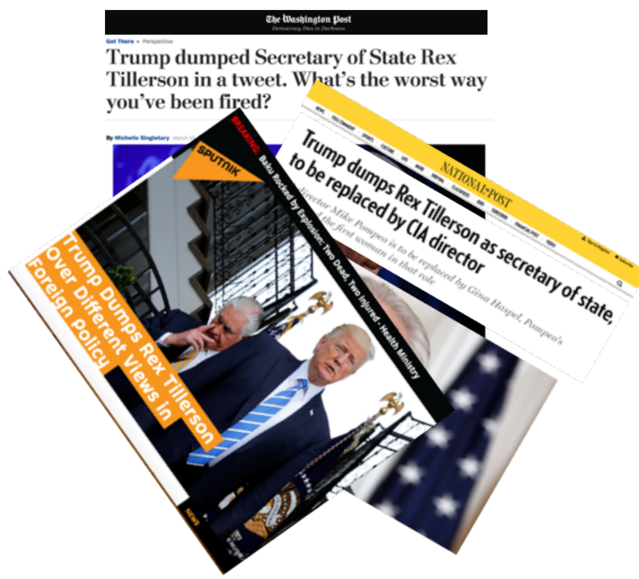

Figure 1: Example of facts and facets in news headlines

Capturing nested relations and their semantics can significantly improve the quality of Open Information Extraction (OIE), which is the core of a wide range of applications, such as Summarization and Information Aggregation. Most of existing (OIE) systems [1, 9, 12, 19, 24] use *n*-ary relations to capture more information about binary facts. However, they suffer from a number of limitations. First, they do not extract nested relations. Second, they use *n*-ary relations to represent related facts and facets. However, *n*-ary relations are flat capturing the co-occurrence of facts and facets but not how they are linked or what kind of relationship they have (e.g.via the preposition *"over"*). Third, relation arguments are not fine-grained.

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10...$15.00
https://doi.org/10.1145/3269206.3271812

For example, the relation ⟨Trump; dumps; Rex Tillerson in a tweet⟩ has a complex argument where *"in a tweet"* is not detected as facet but as a part of the object. The closest approach to our work is NestIE [4], which extracts nested relations. However, like all other OIE systems, it does not provide facet semantic labeling. There are several Semantic Role Labeling approaches [7, 16], which deal mostly with verbial clauses. Some other SRL approaches focus on labeling noun phrase arguments using prepositions [6, 22, 26], however, they consider only few one-word prepositions, which does not cover the complexity of news content.

To address the above limitations we propose StuffIE[1], a facet-centric information extraction approach. The main characteristics of StuffIE are threefold. First, it extracts nested relations by capturing the various links between facts and their facets in a fine-grained way. Second, it avoids having relations with complex arguments not to loose information about facets. Third, it captures the semantic role of facets. Our main contributions are summarized as follows:

(1) We exploit Stanford NLP, and lexical databases such as WordNet to capture fine-grained nested relations and discover implicit facts.
(2) We enhance the extraction process using co-reference resolution to capture relations between facts belonging to different sentences.
(3) We exploit Wiktionary and Oxford Dictionary to extract the meanings of English prepositions, and define seed labels for verb-less facets.
(4) We employ a distant learning approach to label verb-less facets using multi-nominal regression data model.
(5) We run comprehensive experiments on news clauses, comparing StuffIE with OLLIE, ClausIE, and MinIE. The results show that while precision is comparable, our approach achieves a higher recall. Moreover, we use SEMAFOR SRL as a baseline for facet labeling. The results show that the best recall and precision is achieved when combining the baseline with StuffIE.

2 RELATED WORK

Several research areas are related to our work: N-ary Fact Harvesting, Event Extraction, Open Information Extraction, and Semantic Role Labeling. Approaches to N-ary Fact Harvesting [3, 11, 15, 18, 21] are constrained to predefined schemas, where they need canonicalized relations and entities. Thus, they suffer from rigid extractions which are not suitable for complex and dynamic text such as news content. More flexible and relevant approaches address Event Extraction [10, 17, 25] and provide structured event representations by creating links between entities and events. However, they are not sufficiently fine-grained to capture facets and they do not perform semantic labeling. The closest areas to our work are Open Information Extraction and Semantic Role Labeling, which we describe in the following:

Open Information Extraction. Open IE or OIE, refers to relation extraction from free text without pre-defined schema or ontology. There exist several widely used OIE systems [1, 9, 12, 19, 24]. OLLIE [24] and ClausIE [9] are the most popular OIE tools. Feature-wise, OLLIE and CLausIE are similar. They are able to extract *n*-ary relations from sentences, typically in the form of extra adverbials (time, location, etc.) appearing in the sentence. OLLIE additionally is able to add contextual information, i.e., whether a tuple is a belief or a consequence of some condition. While ClausIE is fully unsupervised, OLLIE is distantly supervised using Freebase, leveraging dependency parsing, POS-tagging, and some surface patterns as features. Stanford IE [1] is also distantly supervised, but it does not have the capability to output *n*-ary relations or additional context information. Instead, it is able to infer new facts using natural logic.

Apart from OLLIE [24], which uses few predefined labels to create relations between facts, the other OIE tools do not handle nested relations. Moreover they are neither facet-centric nor fine-grained because they output fairly complex arguments. Few approaches addressed the problem of nested relations [4, 5]. SRL-IE [5] generates nested relations when a given argument to one verb is long and contains a full semantic tuple with a different verb. This approach fails in the presence of incomplete tuples, for example with a missing subject, because it does not exploit grammatical dependencies to find missing arguments. Moreover, it can only extract verbial facets. The closest approach to our work is NestIE[4], which uses a nested representation to extract higher-order relations. The difference with our approach is that, like most of OIE tools, NestIE does not perform semantic labeling. The only work that employs semantic labeling is SRL-IE[5] which was described earlier. The problem is that the labeling is done around verbs, and thus limited only to verbial facets.

Semantic Role Labeling. Semantic Role Labeling (SRL) [13, 20], in contrast to OIE, provides relation extraction with deeper semantics. It assigns roles to tokens appearing in the sentence according to context information (also called frame). SRL methods strongly need training data and a lexical database of roles and frames. Some of the most-used SRL tools are SEMAFOR [16] and SENNA SRL [7], which use FrameNet [2] and PropBank [14] as their lexical database, respectively. Most of SRL approaches work for verbial clauses. Some other SRL approaches focused on labeling relation arguments based on prepositions [6, 22, 26]. However, they use only 35 prepositions, focusing on those consisting only of a single word. By contrast, in our work we consider all English prepositions, using Wiktionary and Oxford Dictionary. This includes multi-word prepositions like *"because of"* and *"due to"*. Covering all prepositions is crucial for news content, which has a complex and dynamic nature.

Our work combines Open Information Extraction and Semantic Role Labeling. With respect to existing approaches in both areas, StuffIE handles nested relations, extracts implicit facts by exploiting grammatical dependencies, captures relations between facts belonging to different sentences using co-reference resolution, provides fine-grained arguments to favor the discovery of facets, performs semantic labeling of verb-less facets, and exploits existing SRL techniques to label verbial facets.

3 PROBLEM STATEMENT

Facts of the form $\langle arg1; predicate; arg2 \rangle$ typically occur with complementary information that we call *facets*. A facet can be (1) verb-less or (2) verbial, and thus dependent on another fact. We consider the following example:

[1] StuffIE's resources are available at https://gitlab.inf.unibz.it/rprasojo/stuffie

"S1: U.S. President Donald Trump fired Secretary of State Rex Tillerson on Tuesday. Trump has nominated CIA Director Mike Pompeo to replace Tillerson as America's new top diplomat."

We can see that the fact *"Trump fired Secretary of State Rex Tillerson"* has the facet *"on Tuesday"*, with no verb. By contrast, the fact *"Trump has nominated CIA Director Mike Pompeo"* has two facets. The first one *"to replace Tillerson"* contains a verb, which is the predicate of the fact *"Pompeo replaces Tillerson"*. The second facet is *"America's new top diplomat,"* which is verb-less.

There are several challenges related to the extraction of facets, both of verb-less and verbial forms. For verb-less forms, prepositions are the main indicators of the presence of facets. However, they can be sometimes misleading. Let us take the following example:

"S2: Mr Tillerson received a phone call on Friday from Chief of Staff John Kelly, who told him that he was being let go."

"S3: Mr. Tillerson learned he had been fired on Tuesday morning when a top aide showed him a tweet from Mr. Trump announcing the change."

We can see that both sentences contain the preposition "*from.*" In sentence $S2$, *"from Chief of Staff John Kelly"* is a facet of the fact *"Mr. Tillerson received a phone call"* because it completes the action of the fact. In other words, it answers the question, *"from where did Mr. Tillerson receive the call?"* However, in sentence $S3$, *"from Mr. Trump"* is not a facet of the fact *"a top aide showed a tweet"* because it complements the object and not the action. It does not answer the question *"from where did the top aide show a tweet?"* but rather *"from where did the tweet come?"* Regarding verbial facets, the main challenge is how to extract the implicit facts contained in them. For example, in sentence $S1$ the fact *"Pompeo replaces Tillerson"* is not explicitly mentioned, but needs to be extracted by finding the subject of the verb. In some cases, the subject can be extracted but in the form of a reference, (e.g., he, him, etc). In this case, we need to find the referenced entity to have self-contained relations.

To solve the above problems, we need to exploit the universal dependencies of the grammatical tree, which we need to enhance using lexical databases. The goal is to differentiate between objects, indirect objects, and verb complements, and thus have a more accurate detection of facets. Moreover, we need to define some rules for detecting implicit subjects and perform co-reference resolution in order to find more facts, which are also self-contained.

Once facets are detected, it is important to label them for a more informative knowledge representation. This can be useful for several applications such as summarization and schema learning. For examples the two facets of the fact *"Trump has nominated CIA Director Mike Pompeo"* shown in sentence $S1$ can be labeled as *"purpose"* and *"function."* Most Semantic Role Labeling approaches deal with verbial facets, and few effort was made for verb-less facets because they are more challenging. First, we need human effort for defining the labels and building training sets. Second, prepositions can give a hint on what a facet is about but they can also be ambiguous. To solve the above problems, we need to exploit English dictionaries to get accurate information about the different use cases of prepositions. Moreover, we need to use a distant learning approach to reduce the cost of human effort.

4 EXTRACTION OF FACTS AND FACETS

To extract facts and facets from unstructured text, we focus on two main aspects. First, we identify all facets related to each fact, so we can effectively select query relevant content. Second, we capture relations between relations to facilitate the navigation between the different levels of detail. In the following, we first describe how relations are represented, then how they are extracted.

4.1 Definitions

We extract facts and facets in the form of nested triple relations. We consider each relation r to have the following form:

$$r = \langle i, \mu, F \rangle,$$

where i is an integer identifier of r, $\mu = \langle s; p; o \rangle$ is a triple representing a fact and $F = \{\langle c_j; f_j \rangle \mid 1 \leq j \leq |F|\}$ is the set of facets related to the fact μ. In the following, we show an example of a relation extracted from a news sentence:

Sentence: US President Donald Trump has fired Secretary of State Rex Tillerson via Twitter, naming CIA Director Mike Pompeo as his replacement. [*BBC News*]
Extracted Relation $\boldsymbol{r}$: $\langle \boldsymbol{i} = \#1, \boldsymbol{\mu} = \langle$Donald Trump; has fired; Rex Tillerson$\rangle$, $\boldsymbol{F} = \{\langle$via; Twitter$\rangle$, $\langle$naming; Mike Pompeo$\rangle$, $\langle$as; his replacement$\rangle\}\rangle$

Facts. We use $\langle s; p; o \rangle$ triples to represent facts. Subject s and object o can be either a noun phrase or a reference to another relation. In this way, relations can become nested in the case of complex sentences. Let us take the following example, where we can see that the object of the fact relation points to another relation:

Sentence: Mr. Tillerson learned he had been fired on Tuesday morning. [*New York Times*]
Extracted Relations:
#2: ⟨Mr. Tillerson, learned, #3⟩
#3: ⟨he; had been fired, ⟨_⟩⟩
⟨on; Tuesday morning⟩

The predicate p of fact triples can have one of the following forms: (1) a verb phrase, if the subject s or the object o is a noun phrase, or (2) a clause connector, if the subject s and the object o point to another relation. Let us take the following example:

Sentence: Trump fired Tillerson Because Russia Wanted Him Fired. [*TODAY In POLITICS*]
Extracted Relations:
#4: ⟨Trump; fired; Tillerson⟩
#5: ⟨Russia; wanted; #6⟩
#6: ⟨Him; fired; ⟨_⟩⟩
#7: ⟨#4; because; #5⟩

We can see that relations #4, #5, and #6 have predicates as verbs, while the predicate of relation #7 is a clause connector. Note that the symbol ⟨_⟩ is used to represent an empty placeholder, like in relation #6.

Facets. As described earlier, for each fact, we extract the list of related facets of the form $F = \{\langle c_j; f_j \rangle \mid 1 \le j \le |F|\}$. Each f_j is a facet detail that can be either a noun phrase or a reference to another relation. By contrast, c_j is a modifier that we call *facet connector* such as *"of", "at", "on",* or *"in."* In some cases, c_j may consist of more than one word, such as *"in case of"* and *"in response to."*

4.2 Triple Extraction

To extract triples from unstructured text, we proceed as follows. We exploit the Stanford dependency parser to get the grammatical tree of each sentence. The nodes of the grammatical tree are the words of the sentence, while the edges represent their syntactical relationships. Note that edges are directed, so each node is the *head* of its outgoing edges and the *target* of its incoming edges.

We start by taking each verb node as predicate of a triple relation. Then, we process all the paths from that verb node to find the corresponding subject, object, and facets. Note that paths from verbs can include both incoming and outgoing edges depending on the type of the verb. To find subjects, objects, and facets, we use handcrafted rules based on the type of dependencies in the grammatical tree. The rules for finding the subject of a predicate are based on edges representing dependencies of type subject (i.e., containing *"subj"*) or adjectival clause (i.e., *"acl"*). Formally, we use Algorithm 1 to find the subject of a given predicate.

Algorithm 1: Find subject

```
Data: A = {"nsubj", "nsubjpass", "csubj"}
Result: s the subject of predicate p
FindSubject(p)
begin
    n = getPredicateNode(p);
    e = getOutgoingEdge(n);
    if e =∈ A then
        return Target(e);
    end
    if e == "acl" then
        return Head(e);
    end
    e = getIncomingEdge(n);
    e_next = getIncomingEdge(Target(e));
    if e == "cop" & e_next == "nsubj" then
        return Target(e_next);
    end
end
```

In Figure 2, we show an example of two subject dependencies among the ones exploited by our algorithm. In the first sentence, *"Tillerson"* is the subject of verb *"was"* because they are connected via the path *"cop→nsubj"*. In the second sentence, *"reasons"* is the subject of the verb *"causing"* due to the dependency *acl.*

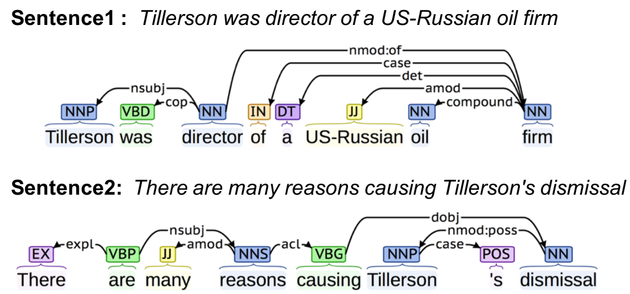

Figure 2: Examples of subject dependencies

In some cases, none of the above dependencies can find the subject of some verbs. An example is given in Figure 3, where the verb *dismissing* does not have a subject. In this case, we perform a recursive procedure. We start from the verb *p* with a missing subject and get its head verb *h* via the *"advcl"* (or *"xcomp"*) dependency. If we get the head *h* via the *"advcl"* and *"for"* is the head of *p* via *"mark"* then we get the object of the head verb as subject. By contrast, if we get the head *h* via the *"xcomp"* and *"to"* is the head of *p* via *"mark"* then we get the subject of the head verb as subject. If the subject is not found, we repeat the process starting from verb *h* until we find a subject. In the example of Figure3, we go from the *"dismissing"* to the verb *"praises"* via the *"advcl"* dependency, then we return *"Trump"* as the subject of *"dismissing."*

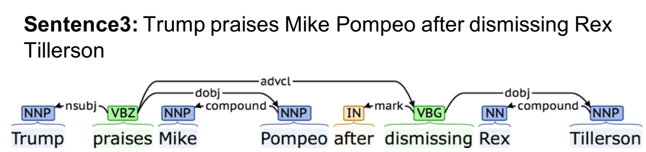

Figure 3: Examples of missing subject

The rules for finding the object of a predicate are based on edges representing dependencies of type *"dobj", "iobj", "nmod", "ccomp",* and *"advcl".* Unlike existing information extraction tools, we additionally tackle the problem of indirect objects indicated by the *"iobj"* dependency. To do that, we exploit a lexical database containing indirect object transformation rules [8] to find the direct object of the verb. Formally, we use the following algorithm to find the object of a given predicate:
An example of indirect objects is given in Figure 4, where we extract *the boot* as the object for the predicate *gave*, and we transform the indirect object Tillerson to a facet as described further.

The procedures described above start from single nodes in the grammatical tree. In many cases, objects, subjects and predicates have compound forms. Thus we use the following dependencies to complete the extraction process. For predicates, we exploit *"xcomp", "auxpass","mwe", "advmod"* dependencies to have their complete forms. Examples of predicates of more than one word include: *"started to work," "has been fired," "finally denounces,"* and *"appointed instead of."* In some other cases, predicates can be implicit and thus

Algorithm 2: Find missing subject

```
Data: A = {"advcl", "xcomp"}
Result: s the subject of predicate p
FindMissingSubject(p)
begin
    n = getPredicateNode(p);
    e = getIncomingEdge(n);
    if e =∈ A then
        headVerb = Target(e);
        if e == "xcomp"& getTarget(p, "mark") ==
         "to" or e == "advcl"& getTarget(p, "mark") ==
         "for" then
            ms=FindObject(headVerb);
        else
            ms=FindSubject(headVerb);
        end
        if ms == null then
            return FindMissingSubject(headVerb);
        end
        return ms;
    end
end
```

Algorithm 3: Find object.

```
Data: B = {"dobj", "nmod, "ccomp", "advcl"}
Result: o the object of predicate p
FindObject(p)
begin
    n = getPredicateNode(p);
    e = getOutgoingEdge(n);
    if e =∈ B then
        return Target(e);
    end
    if e == "iobj" then
        return getDirectObject(Target(e));
    end
end
```

Sentence4 : *Trump gave Tillerson the boot*

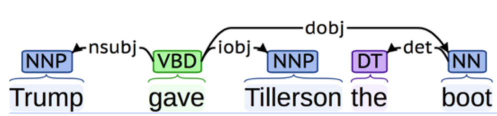

Figure 4: Examples of object dependencies

indicated using the dependency *appos*. An example of such case is shown in Figure 5. We can see that *Tillerson* is connected to *Secretary* via the dependency *appos* and there is no verb in between.

In such cases, we create a synthetic predicate with the head noun as the subject and the target noun as the object. We use the verb *"to be"* in the tense of the clause source. In the previous example, we would output the triple ⟨*Rex Tillerson*, *is*, *the Secretary of State*⟩. Formally, the algorithm for finding the complete compound form of the predicates works as shown in Algorithm 4.

Algorithm 4: Find compound predicates

```
Data: C = {"xcomp", "auxpass","mwe", "advmod"}
Result: cp the completed form of predicate p
CompletePredicate(p)
begin
    n = getPredicateNode(p);
    e = getOutgoingEdge(n);
    if e =∈ C then
        return Target(e) + p;
    end
end
```

Sentence5 : *Trump ousted Rex Tillerson, the Secretary of State*

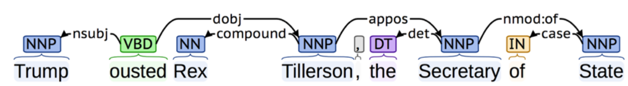

Figure 5: Example of a missing predicate

Similarly, subjects and objects are often noun phrases rather than single words, such as *"Donald Trump"* and *"the Secretary of State"*. To find the complete subjects and objects we exploit *"compound"*, *"nummod"*, *"det"*, *"advmod"*, *"amod"* dependencies, as shown in the following algorithm:

Algorithm 5: Find compound subject and object

```
Data: D = {"compound", "nummod", "det", "advmod", "amod"}
Result: cn the complete form of the input noun
CompleteSubjectAndObject(noun)
begin
    n = getPredicateNode(p);
    e = getOutgoingEdge(n);
    if e =∈ D then
        return Target(e) + p;
    end
end
```

Note that when compounding words, the original ordering is preserved. We use the Stanford enhanced++ dependency graph for conjunction processing. In case of multiple subjects or objects because of conjunction, we copy multiple versions of the verbs according to the number of conjuncts and process them in parallel.

4.3 Facet Recognition

After extracting facts by finding the subject and object of each predicate in a sentence, we proceed with the identification of the facets related to each fact. Typically, facets occur when a verb has complements. For example, in the sentence shown in Figure 6, the

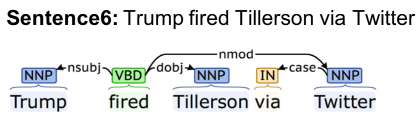

Figure 6: Examples of verb complement

verb *"fired"* has a prepositional complement (*"via Twitter"*) through the dependency *"nmod"*.

The direct object of the predicate is included in the fact relation, then complements are converted into facets. In the case of a missing direct objects, one of the complements is chosen as the object of the fact relation. The priority of choosing is as follows: (1) *"dobj"* dependency, (2) *"nmod"* dependency, and (3) other dependencies. As a tiebreaker, we use the word distance from the object to the verb in the sentence, with negative distance (i.e., objects appearing before the verb) having lower priority than positive distance.

Once facets are identified, we need to find how they connect to the corresponding fact. To achieve that, we exploit the dependencies *"case"*, *"mark"* and *"advmod"*. The first dependency is used for facets that are treated as a separate syntactic word such as prepositions. For example, in the above sentence shown in Figure 6, the case connector is the word *"via,"* which connects the facet *"Twitter"* to the fact *"Trump fired Tillerson."* By contrast, the second and third dependencies indicate as connector the word that introduces the facet clause to the fact clause. For example, when dealing with complement clauses, the connector can be *"that"* or *"whether"*, while for adverbial clauses, the connector can be *"while"* or *"although"*. Let us take the following sentence: *"President Donald Trump announced Tuesday morning that he had fired Secretary of State Rex Tillerson"*. The sentence contains two facts connected with the mark *"that"*, where the second fact is a facet of the first one.

The connectors are crucial, not only for having an intuitive, understandable, and fine-grained representation of facts, but also for facet labeling. They are the main features for learning facet labels as described in Section 5.

4.4 Nested Relations

We have seen how facts and facets are extracted and how each of their components are identified, including subjects, objects, and facet connectors. Obviously, these arguments are not always noun phrases. They can also be clauses containing others facts and facets. So, to capture the complexity of sentences that often occur in unstructured text, particularly in news content, we propose to include links in the extracted facts and facets providing nested relations. So, whenever we find a subject, an object, or a facet detail of complex form, meaning that it depends on other facts, we replace them by the ids of the corresponding triples. The example below shows several nested relations. We can see that the fact ⟨Mr. Tillerson; learned; #9⟩ has as object the fact number #9: ⟨he; has been; fired⟩. Similarly, the facet ⟨when; #10⟩ points to the fact number #10: ⟨a top aide; showed; a tweet from Mr. Trump⟩.

In addition to connecting relations trough their arguments, we perform reference resolution to give more details about the context of each relation. As we can see in the example below, the subject of the fact ⟨he⟨ref#8s⟩; had been; fired⟩ has a reference to the subject of the preceding fact, where #8s indicates that *"he"* refers to the subject of fact number #8, *Mr. Tillerson.* While there is an an attempt by NestIE[4] to have nested relations, their work is restricted to relations between facts belonging to the same sentence. By contrast, in our work, we integrate co-reference resolution in the extraction process to create links between relations belonging to different sentences. In this way, we can have more context information that help in several tasks such as the reconstruction or the summarization of a news story from the extracted relations.

Sentence: Mr. Tillerson learned he had been fired on Tuesday morning when a top aide showed him a tweet from Mr. Trump announcing the change, according to a senior State Department official. [*New York Times*]
Extracted Triples and Facets:
#8: ⟨Mr. Tillerson; learned; #9⟩
#9: ⟨he⟨ref#8*s*⟩; had been; fired⟩
⟨on; Tuesday morning⟩
⟨when; #10⟩
⟨according to; a senior State Department official⟩
#10: ⟨a top aide; showed; a tweet from Mr. Trump⟩
⟨to; him⟨ref#8*s*⟩⟩
⟨announcing; the change⟩

5 LABELING FACETS

After extracting relations, we proceed with labeling the facets to indicate their role given the corresponding facts. For example, a facet can be the *"reason,"* the *"purpose,"* or the *"consequence"* of its related fact. To achieve that, we employ a distant learning approach, which is divided into three main steps. First, we construct the set of facet labels. Second, we build the training data based on lexical data sources. Third, we build a classification model for facet labeling. In the following, we describe the details of each step.

5.1 Defining Facet Semantic Labels

The connector of a facet represents a strong indicator of what the facet is about. For example, if a fact and a facet are related via the connector *"because,"* this means that the facet is the *"reason"* of the fact. Starting from this observation, we use connectors as seeds for defining facet labels. In this work, we consider only prepositions, since they give more precise hints than complement clause connectors such as *"that"* or *"whether"*. While *"because"* indicates that what comes after is a *"reason"*, *"that"* and *"whether"* introduce the next clause without indicating what it is about. Note that although we use only prepositions as seeds, the labeling at the learning phase is done with any type of connectors.

We have used Wiktionary[2] as a lexical data source for getting information about prepositions. Wiktionary provides a complete list of English prepositions in a dictionary-like style. For each preposition, it gives its possible meanings, and for each meaning it provides a gloss consisting of some example sentences. Using Wiktionary, we have analyzed all the set of English prepositions and their roles. For example, the preposition *"for"* has 18 roles including *"reason"*, *"purpose"* and *"duration"*. As a result of this analysis, we have came up with 22 handcrafted labels. Then we have consolidated these

[2] https://en.wiktionary.org/

labels by comparing them with the labels provided by Illinois SRL [6] using the same descriptions. Then we have added three more labels that were not present in [6] resulting in 35 labels. The list of labels is given in following: {*"Activity", "Agent", "Attribute", "Beneficiary", "Cause", "Comparison", "Condition", "Conjunction", "Co-Participants", "Destination", "Direction", "EndState", "Experience", "Instrument", "Journey", "Location", "Manner", "MediumOfCommunication", "Numeric", "ObjectOfVerb", "Opponent/Contrast", "Other", "Participant/Accompanier", "PartWhole", "PhysicalSupport", "Possessor", "ProfessionalAspect", "Purpose", "Recipient", "Separation", "Source", "Species" , "StartState", "Temporal", "Topic"*}.

5.2 Building Training Data

We have used the connectors to define the set of possible facet labels. Since a connector can have several roles depending on how it is used in a sentence, we need to build the training data based on the glosses of each connector. So, each gloss will be mapped to one facet label. We can see in the following example three meanings and glosses related to the preposition *"for"*, which correspond to the facet labels *"purpose"*, *"reason"*, and *"duration"*.

> **for:**
> 1. In order to obtain or acquire. **(gloss1)**
> *"I'm saving up* **for** *a car".*
> 2. Because of. **(gloss2)**
> *"He looks better* **for** *having lost weight".*
> 3. Over a period of time. **(gloss3)**
> *"I've lived here for three years".*

The idea is to label each gloss by computing its similarity with all the predefined facet labels. Then we choose the most similar label. Since labels are single words and glosses contain more words, we need first to enrich facet labels with context information.

We enrich each facet label using WordNet.[3] As shown in Algorithm 6, for a given label l, we first take its related synsets $\{s_i\}$ and all glosses of those synsets. All words in the synsets that we consider synonyms are included in the context of l. By contrast, for each gloss of a given synset s_i, we include only similar words to s_i in the context of l, as shown in line 7. As a further step, we enrich also the synonyms of label l using their context. To do that, we get all the synsets on WordNet that contain the synonyms of l. Then, for or each word contained in those synsets, we get its relevant context using recursion, as shown in line 13. The results are then added to the context of l. Note that to avoid noise which can be generated using recursion, we increase the threshold of finding similar words at each recursive step. For similarity between words, we use WordNet::Similarity with the Wu and Palmer similarity metric [27]. We also normalize the words by using only noun forms, so adjectives, adverbs, and verbs are converted into nouns.

After the enrichment step, each facet label consists of a set of words. We use the Bag-Of-Words model to model both facet labels and glosses. Then, we proceed with the gloss labeling as described in Algorithm 7.

For each gloss, we compute its similarity with all facet labels and we take the most similar one as the label of the gloss. We consider

[3] https://wordnet.princeton.edu

Algorithm 6: Enriching facet labels

```
   Data: l, a facet label, C, a set of words, the initial context
   Result: C, the context of label l
1  getContext(l, C)
2  begin
3      S = getRelatedWordNetSynsets(l);
4      for each synset s_i ∈ S do
5          SW = getAllWordsInSynset(s_i);
6          GW = getAllWordsInGloss(s_i);
7          C = C + SW + getSimilarWords(GW, s_i);
8          for each word w_i ∈ SW do
9              S' = getWordNetSynsetsContaining(w_i);
10             SW' = getAllWordsInSynsetSet(S');
11             for each word d_k ∈ SW' do
12                 if w_i ∉ C then
13                     C = C + getContext(w_i, C);
14                 end
15             end
16         end
17     end
18     return C;
19 end
```

Algorithm 7: Labeling glosses

```
   Data: g, a gloss
   L: the set all facet labels
   Result: S, a label of g
1  Label(g)
2  begin
3      S = getSimilarLabels_Overlap(L, g);
4      if S.size() == 1 then
5          return S.getUniqueElement();
6      end
7      S = getSimilarLabels_EdgeDistance(L, g);
8      if S.size() == 1 then
9          return S.getUniqueElement();
10     end
11     S = getSimilarLabels_WordDistance(L, g);
12     if S.size() == 1 then
13         return S.getUniqueElement();
14     end
15     S = getSimilarLabels_WordPaiwise(L, g);
16     if S.size() == 1 then
17         return S.getUniqueElement();
18     end
19     return null;
20 end
```

each occurrence of a word in a gloss to be a unique item, so the same word appearing in two different positions will count twice. We use several distance measures between a label and a gloss in the following order: (1) count the number of overlapping words (line 3);

(2) count the number of overlapping words, where the words of the gloss are weighted: we weight each word by its edge distance to the root of the Stanford dependency parsing tree, the closer it is, the higher is the weight (line 7); (3) count the number of overlapping words, where the words of the gloss are weighted: we weight each word by its word distance to the root of the Stanford dependency parsing tree (line 11); (4) use average pairwise comparison between the words of the label and the words in the gloss. We use a hybrid similarity measure between words using wordnet::similarity and word2vec. The average of the two scores is used for similarity computation. Each similarity distance is used as a tiebreaker if the previous one fails.

We have used the above distances with the proposed order after an extensive experimental study that showed that the proposed approach achieves the most accurate results. For that, we have manually labeled 100 randomly selected glosses, and then compared those labels with the outcome of our approach. We achieved a precision of 93%. Then, using the proposed approach, we have automatically labeled all the glosses of the English prepositions taken from both Wiktionary and Oxford Dictionary.

5.3 Learning Model for Semantic Labels

We obtained, from the previous step, a training data set consisting of 7000 labeled instances where each instance is a sentence. Note that a gloss can contain more than one example sentence. The next step is how to represent these data instances. We distinguish two types of features: *textual* and *structural*. Textual features describe the content of each instance, while structural features capture its grammatical dependencies.

Textual features. We use the following textual features to represent each sentence: the subject, object, predicate, facet phrases, and the remaining content of the sentence. We have converted each of these features to a 300-dimensional vector based on sentence encoding [23], which uses the word vector data model. Additionally, we use some more specific features than the previous ones, namely the headwords of the subject, predicate, object, and facet. Since these are single words, we encode them using word2vec embeddings by 300-dimensional vectors. For each feature pointing to another relation, we replace it by the target relation and perform the word embedding accordingly.

Structural features. We use the types of the textual features described above. So, entities are changed to their types (e.g., Person, Location) using Stanford NER. Moreover, WordNet is used to tag tangible and intangible objects. Predicates and headwords are changed into their types using SEMAFOR SRL [16]. Similarly to textual features, we use word embeddings to encode structural features. We additionally use word counts and dependency arcs. The encoding based on word embeddings results in 3000 textual features and 7204 structural features. We feed these features with their labeled instances into a multi-nominal logistic regression model to perform facet labeling. We have chosen multi-nominal logistic regression because it is a scalable model that handles high dimensional data, besides the fact that it is suitable for textual data and multi-class classification.

6 EXPERIMENTS

In this section, we describe the setup of our experiments, then we present and discuss the results.

6.1 Setup

Dataset. We used two datasets in our experiments. The first one is DUC'04,[4] containing 1,000 news articles with 20,000 sentences in total. The second dataset is a set of 1,000 English glosses extracted from Wiktionary and the Oxford Dictionary. The facets of the example sentences of each gloss are automatically labeled, using the distant learning approach described in section 5.2

Baselines. We have used four baseline approaches to evaluate StuffIE. Two baselines from Open Information Extraction (OIE) and two baselines from Semantic Role Labeling (SRL). Although NestIE [4] is the most similar to our work, we did not use it as a baseline because (1) there is no source code or demo tool available for it, and (2) the paper does not contain enough details for a correct implementation of NestIE. For example, we do not have access to the hypothesis and the labeled data upon which the system is built. Instead, we have used OLLIE [24] and ClausIE [9], one of the most popular OIE tools. Since these tools do not capture nested relations and for a fair comparison, we have made the following assumption. We take the n-ary relations produced by OLLIE and ClausIE and we consider all arguments after predicates as facets. Regarding Semantic Role Labeling (SRL), we have used SEMAFOR [16] for labeling verbial facets, and Illinois SRL [6] for labeling verb-less facets using prepositions.

Method	*Rels.*	*Facets*	*Miss.*	*Nested*
ClausIE	76906	105840	158	16381
OLLIE	70862	69781	642	6161
StuffIE	**83265**	**107210**	**0**	**22740**

Table 1: Fact and facet extraction statistics

Evaluation.

We have performed four main evaluations:

(a) The first evaluation consists in comparing StuffIE with ClausIE and OLLIE to compute statistics based on the number of discovered facets. This evaluation focuses on the granularity aspect of the approach. To achieve that, we use the DUC'04 dataset and compute the following measures: (1) the number of extracted relations (*Rels.*); (2) the number of extracted facets (*Facets*); (3) the number of missing extractions from the sentence, either because of errors in the tool or because simply the tool does not output anything (*Miss.*); (4) the number of nested relations (*Nested*), which we capture in the case of ClausIE and StuffIE by computing the redundancy between relations. We consider a relation extracted from a sentence to be redundant when it is contained as an argument in another relation extracted from the same sentence.

(b) The second evaluation is about facet labeling, where we assess the accuracy of the multinomial logistic regression using 10-fold cross validation over the labeled dataset of English glosses. The accuracy is computed as the fraction of correctly labeled facets. We use three types of configuration for the used features: textual

[4] http://duc.nist.gov/duc2004/tasks.html

features (*T*), a combination of textural and structural features (*T+S*), structural features and Textual features of the facet and its connector facet (*Facet T+S*). We also used Illinois SRL as a baseline approach.
(c) The third evaluation is about facet coverage, where we compute the percentage of facets that are labeled by our model compared to SEMAFOR SRL and Illinois SRL, using the DUC'04.
(d) The fourth evaluation aims at computing the precision of StuffIE, SEMAFOR SRL, and Illinois SRL using manual assessment on 100 clauses extracted randomly from the DUC'04. We asked human assessors to indicate whether the label assigned to the fact using each of these approaches was correct or not.

6.2 Results

Table 1 shows that StuffIE extracts more relations than ClausIE and OLLIE and detects 1370 more facets than the best performing baseline. Similarly, StuffIE captures 6359 nested relations and 21102 more than OLLIE. Moreover, StuffIE is able to process all sentences, unlike ClausIE and OLLIE which cannot handle hundreds of sentences.

Method	*Precision*
Illinois SRL	0.293
SEMAFOR SRL	0.660
StuffIE (S + facet T)	**0.833**

Table 2: Facet labeling performance on manually-labeled data; S = structural feature, T = textual feature

The results about the accuracy of the classification model built using distant learning are shown in Table 3. We can see that all the different versions of our approach perform better than Illinois SRL. Using only textual features achieves only 54.5% of accuracy, which increases up to 73.9% when combining the textual features of facets and their connectors with the structural features. This configuration of StuffIE is used for all the subsequent evaluations. In contrast to accuracy on the automatically labeled data, the precision values for StuffIE are higher with the manually labeled data. Note that the manually labeled data is completely different from the automatically labeled one. The first one is extracted from news articles, and the second one is from Wiktionary and Oxford dictionary. We can see the results in Table 2 where StuffIE beats the baselines with a large margin reaching 83% of precision.

Method	*Accuracy*
Illinois SRL	0.278
StuffIE (T)	0.545
StuffIE (T+S)	0.698
StuffIE (S + facet T)	**0.739**

Table 3: Facet labeling performance on automatically-labeled data; S = structural feature, T = textual feature

The last set of results is shown in Table 4. We can see that StuffIE performs worse than SEMAFOR SRL in detecting verbial facets, but better than both SEMAFOR and Illinois SRL in detecting verb-less facets using prepositions. However, the overall number of discovered facets by StuffIE reaches, 76.2% which outperforms the baseline approaches by a large margin.

Discussion. The results show that StuffIE outperforms baseline approaches with a large margin. Regarding facet extraction, StuffIE is shown to be more fine-grained than ClausIE and OllIE, given that it discovers more facets. Additionally, it captures more nested relations. Interestingly, we can also see that StuffIE performs much better on news sentences than on glosses. This can be explained by the fact that the model is constructed using all English preposition glosses and tested with a completely different dataset. Thus, having complete knowledge of the prepositions increases the predictive power of the model. This does not happen with the 10-fold cross validation where we are constrained to sacrifice a part of the training data to build the model. Another point is that the validation is done using manually labeled data, which is more precise than the automatic one. Although the precision of the ground truth is 93%, a 7% of noise still provides highly accurate results.

Regarding semantic labeling, it is clear that StuffIE does not perform well with verbial facets because it is designed for labeling verb-less facets. We can see that StuffIE performs much better than Illinois SRL because we take into account all English prepositions while Illinois SRL deals only with 35 of them focusing only on one-word prepositions. We also extend the labeling to other types of clause connectors using distant learning, which explains the high accuracy of our approach.

Method	*Verb-less. facets*	*Verbial. facets*	*Total*
Illinois SRL	46651	0	46651
SEMAFOR SRL	3612	**13973**	17585
StuffIE	**79614**	961	**80575**

Table 4: Facet detection statistics

7 CONCLUSION

We have proposed StuffIE, a fine-grained information extraction approach, which is facet-centric. StuffIE is able to extract verbial and verb-less facets related to each fact using a nested representation of relations. Moreover, it uses semantic labeling to indicate the role of each facet with respect to its relevant fact. We have exploited a number of linguistic databases to build such a system. We have exploited Stanford dependencies and WordNet to extract more facts and detect facets. We exploited Wiktionary and Oxford Dictionary to use prepositions for facet labeling using distant learning. The results show very promising results towards reducing the complexity and the ambiguity of existing Open Information Extraction (OIE) approaches. A natural next step for this work is to combine it with exiting SRL approaches to cover verbial facets.

Acknowledgements. This work has been partially supported by the projects TaDaQua and TQTK, funded by the Free University of Bozen-Bolzano.

Appendices

A EXAMPLE OF SEMANTIC LABELING OF FACETS WITH PREPOSITIONS

Sentence. "It may be that we are looking at a consolidation on the world level that looked like the consolidation on the national level 100 years ago".

Illinois SRL output:
at -> **facet label: Location**
on -> **facet label: Location**
like -> **facet label: Location**
on -> **facet label: Location**

StuffIE output:
1.4: ⟨It; may be; ⟨_⟩ ⟩;
⟨that; #1.8⟩; **(facet label: Other/Details)**
1.8: ⟨we; are looking;⟨_⟩⟩ ;
⟨at; a consolidation on the world level⟩;
(facet label: Direction)
1.17: ⟨that; looked; ⟨_⟩ ⟩;
⟨like; the consolidation⟩; **(facet label: Comparison)**
⟨on; the national level 100 years ago⟩;
(facet label: Temporal)

B EXAMPLES OF EXTRACTED RELATIONS BY STUFFIE AND CLAUSIE

Sentence. "President Donald Trump announced Tuesday morning that he had fired Secretary of State Rex Tillerson and appointed CIA Director Mike Pompeo to replace him, ending months of speculation about how much longer the embattled Tillerson would last in the job". [*News*]
ClausIE:
1:⟨President Donald Trump; announced; Tuesday morning; that he had fired Secretary of State Rex Tillerson⟩;
2:⟨President Donald Trump; announced; Tuesday morning; that he had appointed CIA Director Mike Pompeo to replace him ending months of speculation about how much longer the embattled Tillerson would last in the job⟩;
3:⟨he; had fired; Secretary of State Rex Tillerson⟩;
4:⟨he; had appointed; CIA Director Mike Pompeo to replace him; ending months of speculation about how much longer the embattled Tillerson would last in the job⟩;
5:⟨CIA Director Mike Pompeo; to replace; him⟩;
6:⟨longer the embattled Tillerson; would last; in the job⟩;
StuffIE:
1.4: ⟨Donald Trump; announced; Tuesday morning⟩;
—–- ⟨that; #1.10⟩;
1.10: ⟨he⟨ref#1.4s⟩; had fired; Rex Tillerson⟩;
—–- ⟨and appointed; Mike Pompeo⟩;
—–- ⟨to; replace him⟨ref#1.4s⟩⟩;
—–- ⟨ending; months of speculation⟩;
—–- ⟨about how much longer; #1.38⟩;
1.38: ⟨the embattled Tillerson⟨ref#1.10o⟩; would last in; the job⟩;
1.1: ⟨Donald Trump⟨ref#1.4s⟩⟩; ⟨is⟩; President⟩;
1.11: ⟨Rex Tillerson⟨ref#1.10o⟩; ⟨is⟩; Secretary of State;⟩
1.18: ⟨Mike Pompeo⟨ref#1.17f1⟩; ⟨is⟩; CIA Director⟩;

REFERENCES

[1] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. 2015. Leveraging Linguistic Structure For Open Domain Information Extraction. In *Proceedings of the 53rd ACL and the 7th IJCNLP (Volume 1: Long Papers)*. ACL, Beijing, China, 344–354.
[2] Collin F Baker, Charles J Fillmore, and John B Lowe. 1998. The berkeley framenet project. In *Proceedings of the 36th ACL and 17th ICCL - Volume 1*. Association for Computational Linguistics, 86–90.
[3] Soumia Lilia Berrahou, Patrice Buche, Juliette Dibie, and Mathieu Roche. 2016. Xart System: Discovering and Extracting Correlated Arguments of N-ary Relations from Text. In *Proceedings of the 6th WIMS (WIMS '16)*. ACM, New York
[4] Nikita Bhutani, H. V. Jagadish, and Dragomir R. Radev. 2016. Nested Propositions in Open Information Extraction. In *EMNLP*. The Association for Computational Linguistics, 55–64.
[5] Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2010. Semantic Role Labeling for Open Information Extraction. In *Proceedings of the NAACL HLT 2010 FAM-LbR (FAM-LbR '10)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 52–60.
[6] James Clarke, Vivek Srikumar, Mark Sammons, and Dan Roth. 2012. An NLP Curator (or: How I Learned to Stop Worrying and Love NLP Pipelines). In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*. Istanbul, Turkey, x–y.
[7] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.
[8] SP Corder. 1968. Double-object verbs in English. (1968).
[9] Luciano Del Corro and Rainer Gemulla. 2013. Clausie: clause-based open information extraction. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 355–366.
[10] George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The Automatic Content Extraction (ACE) Program Tasks, Data, and Evaluation. In *Proceedings of LREC-2004*. ELRA, Lisbon, Portugal.
[11] Patrick Ernst, Amy Siu, and Gerhard Weikum. 2018. HighLife: Higher-arity Fact Harvesting. In *Proceedings of the 2018 WWW (WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1013–1022.
[12] Kiril Gashteovski, Rainer Gemulla, and Luciano Del Corro. 2017. MinIE: Minimizing Facts in Open Information Extraction. In *EMNLP*. Association for Computational Linguistics, 2630–2640.
[13] Daniel Gildea and Daniel Jurafsky. 2002. Automatic Labeling of Semantic Roles. *Comput. Linguist.* 28, 3 (Sept. 2002), 245–288.
[14] Paul Kingsbury and Martha Palmer. 2002. From TreeBank to PropBank.. In *LREC*. 1989–1993.
[15] Sebastian Krause, Hong Li, Hans Uszkoreit, and Feiyu Xu. 2012. Large-Scale Learning of Relation-extraction Rules with Distant Supervision from the Web. In *Proceedings of the 11th ISWC (ISWC'12)*. Springer-Verlag, Berlin, 263–278.
[16] Meghana Kshirsagar, Sam Thomson, Nathan Schneider, Jaime Carbonell, Noah A Smith, and Chris Dyer. 2015. Frame-semantic role labeling with heterogeneous annotations. *people* 3 (2015), A0.
[17] Erdal Kuzey, Jilles Vreeken, and Gerhard Weikum. 2014. A Fresh Look on Knowledge Bases: Distilling Named Events from News. In *Proceedings of the 23rd CIKM*. 1689–1698.
[18] Hong Li, Sebastian Krause, Feiyu Xu, Andrea Moro, Hans Uszkoreit, and Roberto Navigli. 2015. Improvement of n-ary Relation Extraction by Adding Lexical Semantics to Distant-Supervision Rule Learning. In *ICAART 2015 - Proceedings of the International Conference on Agents and Artificial Intelligence*,
[19] Filipe Mesquita, Jordan Schmidek, and Denilson Barbosa. 2013. Effectiveness and Efficiency of Open Relation Extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 447–457.
[20] Martha Palmer, Daniel Gildea, and Nianwen Xue. 2010. *Semantic Role Labeling*. Morgan & Claypool Publishers.
[21] Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. 2017. Cross-Sentence N-ary Relation Extraction with Graph LSTMs. *TACL* 5 (2017), 101–115.
[22] Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The Importance of Syntactic Parsing and Inference in Semantic Role Labeling. *Comput. Linguist.* 34, 2 (June 2008), 257–287.
[23] Tengyu Ma Sanjeev Arora, Yingyu Liang. 2017. A Simple but Tough-to-Beat Baseline for Sentence Embeddings.. In *ICLR*.
[24] Michael Schmitz, Robert Bart, Stephen Soderland, Oren Etzioni, et al. 2012. Open language learning for information extraction. In *Proceedings of the 2012 EMNLP*. Association for Computational Linguistics, 523–534.
[25] Dafna Shahaf and Carlos Guestrin. 2012. Connecting Two (or Less) Dots: Discovering Structure in News Articles. *TKDD* 5, 4 (2012), 24:1–24:31.
[26] Vivek Srikumar and Dan Roth. 2013. Modeling Semantic Relations Expressed by Prepositions. *Transactions of the Association for Computational Linguistics* 1 (2013), 231–242.
[27] Zhibiao Wu and Martha Palmer. 1994. Verbs Semantics and Lexical Selection. In *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics (ACL '94)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 133–138.