VerbKB: A Knowledge Base of Verbs for Natural Language Understanding

Derry Tanti Wijaya

CMU-LTI-16-016

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:
Tom Mitchell
William Cohen
Ed Hovy
Martha Palmer

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
In Language and Information Technologies

Keywords: Verbs, Knowledge Base

For Kung and Mbah Tun and for the many early morning walks we used to take together

Abstract

A verb is the organizational core of a sentence. Understanding the meaning of the verb is, therefore, a key to understanding the meaning of the sentence. One of the ways we can formulate natural language understanding is by treating it as a task of mapping natural language text to its meaning representation: entities and relations anchored to the world. Since verbs express relations over their arguments and adjuncts, a lexical resource about verbs can facilitate natural language understanding by mapping verbs to relations over entities expressed by their arguments and adjuncts in the world. In this thesis, we semi-automatically construct a verb resource called **VerbKB** that contains important semantics for natural language understanding. A verb lexical unit in VerbKB consists of a verb lexeme or a verb lexeme and a preposition e.g., "live", "live in", which is typed with a pair of NELL semantic categories that indicates its subject type and its object type e.g., "live in"(*person*, *location*). We present algorithms behind VerbKB that learn two semantic types of mappings for these verb lexical units that will complement existing resources of verbs such as WordNet and VerbNet and existing knowledge bases about entities such as NELL. The two semantic types of mappings are (1) the mappings from verb lexical units to binary relations in knowledge bases (e.g., the mapping from the verb lexical unit "die at"(*person*, *nonNegInteger*) to the binary relation ***personDiedAtAge***) and (2) the mappings from verb lexical units to *changes* in binary relations in knowledge bases (e.g., the mapping from the verb lexical unit "divorce"(*person*, *person*) to the *termination* of the relation ***hasSpouse***). The mappings from verb lexical units to binary relations in knowledge bases such as NELL, YAGO, or Freebase can provide a direct link between the text and the background knowledge about the world contained in these knowledge bases, enabling inferences over the world knowledge to better understand the text. The mappings from verb lexical units to *changes* in binary relations in knowledge bases can facilitate automatic updates of relations and temporal scoping of relations in the knowledge bases.

Acknowledgments

Thank you to Momo and Gendis for being so strangely and delightfully understanding always. Thank you to my advisor, Tom, who is the coolest, wisest, and nicest advisor anyone can hope for. Thank you to my friends especially Hye Ju, Fatima, Meghana, Bhavana, Sunayana, and Yanbo for being the most wonderful company throughout our PhD journey together. Thank you to NELL team especially William, Partha, and Ndapa for your invaluable advice, insights, and friendships. Thank you to Fanny and Annie, for helping to raise Gendis; it's true that it takes a village to raise a child and do PhD at the same time ;) Thank you to my committee members, Ed, William, and Martha, for your insightful comments and advice. Words are not enough to express my gratitude.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

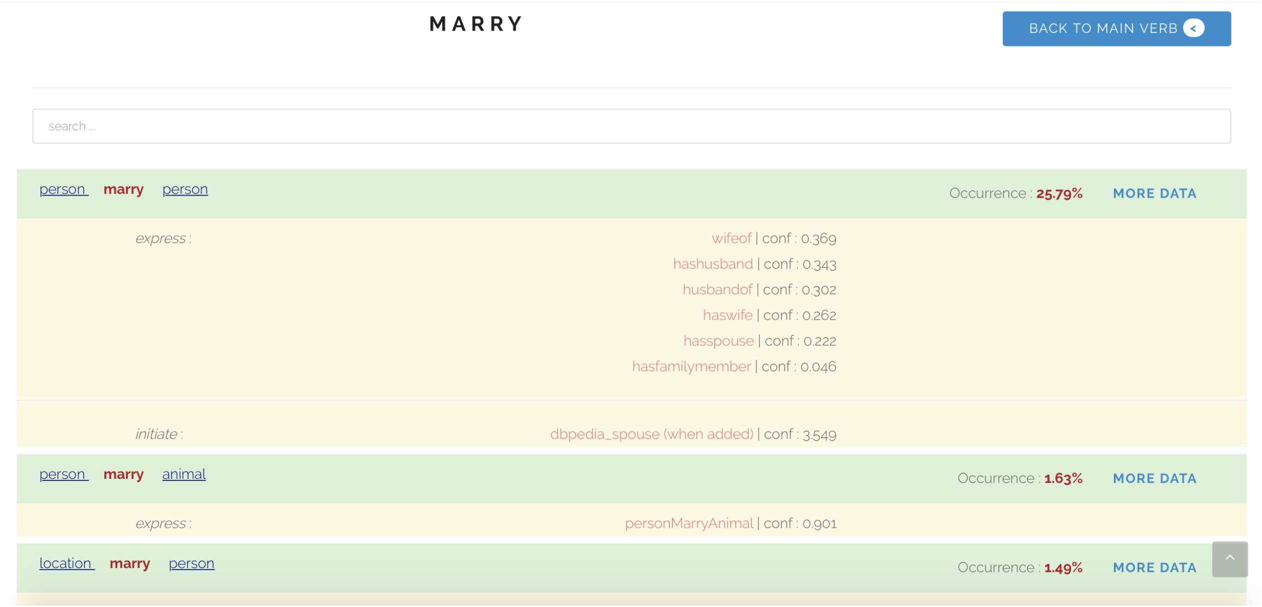
Verbs are the organizational core of sentences; they are predicates over entities in sentences. Some verbs express relationships among these entities in the world, some express events that *change* these relationships, and some do both. Understanding the meaning of a verb in the sentence is, therefore. a key to understanding the meaning of the sentence.

One of the ways we can formulate natural language understanding is by treating it as the problem of mapping natural language text to its meaning representation: entities and their relationships in the world. A lexical resource about verbs can facilitate natural language understanding by mapping verbs in text to relationships or *change* of relationships among the entities that are their arguments or adjuncts. The mappings from verbs to relations or changes in relations among entities in knowledge bases such as NELL [Carlson et al., 2010], YAGO [Hoffart et al., 2013], DBPedia [Auer et al., 2007], or Freebase [Bollacker et al., 2008] can provide a link between natural language text and the knowledge about the world that is contained in these knowledge bases; enabling inferences over the world knowledge to better understand the text.

In this thesis, we construct such a verb resource called **VerbKB** that contains two semantic types of mapping for verbs that we believe can assist natural language understanding and that will complement existing resources of verbs such as WordNet [Miller, 1995] and VerbNet [Kipper et al., 2000], and existing knowledge bases of noun phrases such as NELL.

We define a verb lexical unit in VerbKB to consist of a verb lexeme or a verb lexeme and a preposition – that is, a lemmatized verb phrase that matches the part-of-speech based regular expression: V | VP where V is a verb lexeme and P is a preposition – which is typed with a pair of NELL semantic categories that indicates its subject type and its object type. Examples of verb lexical units in VerbKB are “eat” (*person, food*), “eat with” (*person, tableItem*), “live in” (*person*, *location*), etc.

Complementary to knowledge bases such as NELL, which contains the mappings from noun phrases to entities e.g., “Barack Obama” to *Obama*, the *categories* of the entities e.g., *Obama* is of category *person*, and the *binary relations* – i.e., relations with two arguments – between entities e.g., ***hasSpouse***(*Obama*, *Michelle Obama*); VerbKB contains (1) the mappings from its verb lexical units to *binary relations* that the verb lexical units can express in the knowledge bases, (2) the mappings from its verb lexical units to events that *change the binary relations* among the entities in the knowledge bases. We formulate *changes in relations* as initiations (addition of new values) or terminations (deletion of existing values) of the relations in the knowledge bases. For example, the verb lexical unit “marry”(*person*, *person*) can express the initiation of the ***hasSpouse***(*person*, *person*) binary relation in the NELL knowledge base (see Figure 1.1 of our VerbKB website[1]), while the verb lexical unit “divorce”(*person*, *person*) can express a **divorce** event that terminates the ***hasSpouse*** relation.



MARRY

person marry person Occurrence : 25.79% MORE DATA

express :
wifeof | conf : 0.369
hashusband | conf : 0.343
husbandof | conf : 0.302
haswife | conf : 0.262
hasspouse | conf : 0.222
hasfamilymember | conf : 0.046

initiate : dbpedia_spouse (when added) | conf : 3.549

person marry animal Occurrence : 1.63% MORE DATA

express : personMarryAnimal | conf : 0.901

location marry person Occurrence : 1.49% MORE DATA

Figure 1.1: The VerbKB entry for the verb pattern “marry” showing the mapping of the verb pattern to the relations in NELL that it expresses. It also shows the mapping from the verb pattern to the *initiation* of the ***spouse*** relation in DBPedia. The typed verbs are ranked based on their frequencies of occurrences in the SVO triples.

The overarching goal of constructing this knowledge base of verb lexical units is to provide a link between the surface form verbs and real-world relations in knowledge bases (Figure 1.2).

Since not all verb lexical units in VerbKB have mappings to the existing NELL relations, VerbKB extends the vocabulary of relations in NELL by automatically clustering these verb lexical units into synonym sets (synsets) and proposes synsets that are not mapped to the existing NELL relations as new relations. Starting from verbs extracted from the subject-verb-object (SVO) triples obtained from the high coverage Web-scale corpora of ClueWeb [Callan et al., 2009a] (the semantic types of the verbs’ subject and object fillers defined by the types in the NELL knowledge base) and leveraging the knowledge about the synonymy and antonymy relations between English verbs in manually constructed resources that are the Moby Thesaurus[2] and WordNet, VerbKB automatically clusters 65,679 unique verb patterns (verb lexical units in VerbKB *without* types) into 215,106 synsets, each synset typed with semantic types in NELL and organized into a subsumption taxonomy based on types. Each synset is then either mapped

[1] http://www.dwijaya.org/dvkb.html#DKVB
[2] Moby Thesaurus: http://moby-thesaurus.org

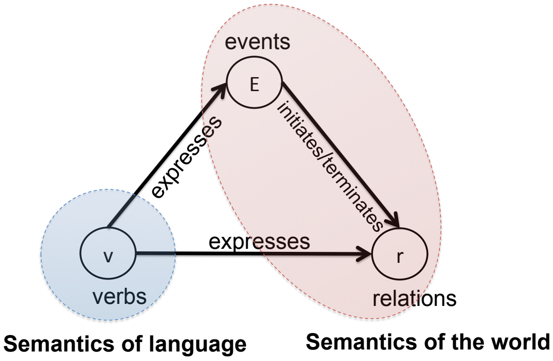


Figure 1.2: The mappings between the semantics of the world and the semantics of the language that refers to this world.

to a pre-existing NELL relation or added as a new relation in NELL to extend the vocabulary of relations that NELL should read.

VerbKB is made available publicly[3] as an effort to facilitate natural language understanding systems by providing a high coverage link between the unstructured surface form of verbs and the structured relations that they express in the knowledge base.

1.1 Overall Description of the General Approach to Building VerbKB

This thesis is about VerbKB that contains clusters of verb lexical units. Each cluster is a placeholder for the binary relation between real-world entities that are the subject and object of the verb lexical units in the cluster. To build VerbKB, we first extract verb phrases that are parts of the subject-verb-object triples obtained from ClueWeb. We lemmatize the verb phrases and select the lemmatized phrases that match this part-of-speech based regular expression: V | VP, where V is a verb lexeme and P is a preposition. We refer to these lemmatized verb phrases (that match the part-of-speech based regular expression: V | VP) as *verb patterns*.

Then, the underlying approach is that the verb patterns are mapped to binary relations or to changes in binary relations between the entities that are the verb patterns' subjects and objects. Since a verb pattern often has multiple senses, e.g., "play", we add to the verb pattern, its type signatures – i.e., NELL types (and supertypes) of its subject and object fillers – to differentiate the individual senses e.g., "play"(*person*, *musicalInstrument*), "play"(*person*, *person*), "play"(*person*, *game*), etc. The use of NELL types means that the sense granularity is constrained by the semantic granularity of NELL types. These verb patterns with type signatures are the lexical units in VerbKB and we refer to them as *typed verbs*.

In Chapter 3, we link these typed verbs to their equivalent existing NELL relations if such exist. We show the value of having the links from typed verbs to knowledge base relations in terms of relation extraction.

[3] http://www.dwijaya.org/dvkb.html#DKVB

In Chapter 4, we link the *addition* or *removal* of the verb patterns into/from Wikipedia pages to the changes: *initiation* or *termination* of Wikipedia infobox relations (in other words, DBPedia relations) if such exist. We show the value of having the links from verbs to *changes* in knowledge base relations in terms of temporal scoping.

In Chapter 5, we add the missing typed verbs-relations into NELL. Here, to avoid unnecessary duplication, we first cluster semantically equivalent typed verbs in VerbKB. Although the clustered typed verbs may have different verb patterns, the fact that their semantics are similar enough – measured by overlaps of their subject and object fillers – and their two type signatures are identical is enough to merge them. For example, we group "marry in"(*person*, *location*) and "wed in"(*person*, *location*) in the same cluster then add the cluster ***personMarryAtLocation*** into NELL as a new binary relation in NELL. VerbKB clusters are mapped to the existing NELL relations or are added as new relations to extend the vocabulary of relations that NELL should read.

The work we presented about VerbKB shows that we can semi-automatically construct such a verb resource that goes beyond existing resources with regards to coverage that is, the number of verb patterns that it contains and the links to knowledge base relations that it provides. We show how to achieve this by leveraging a combination of high coverage text corpora from which we obtain the subject-verb-object triples. We also leverage a knowledge base (NELL) with a rich type system over entities to get the type signatures of the verb patterns. We leverage other pre-existing linguistic resources such as a thesaurus and WordNet for their synonymy and antonymy relations to produce a more precise clustering of the typed verbs.

Regarding coverage, the verb patterns in VerbKB cover subject-verb-object triples that occur a total of over 2 billion times in ClueWeb. Regarding precision, the verb clusters in our VerbKB align better with manually constructed verb clusters compared to the verb clusters in other pre-existing automatically constructed resources. To the best of our knowledge, VerbKB is the largest publicly available knowledge base of English verbs to date that contains mappings from verbs to knowledge base relations and changes in these relations. Most importantly, VerbKB extends the vocabulary of relations that NELL should read for higher coverage representation of arbitrary text.

However, the approach we presented in this thesis suffers from several shortcomings that we plan to address in future work, including:

- We work only with verb patterns that are part of the subject-verb-object (SVO) triples construction. We ignore relations expressed by intransitive verbs e.g., "he runs", the copula whose type signature is not distinct enough to disambiguate the relation e.g., "he is a friend" – the type signature here is (*person*, *person*) and there are adjectives and adverbs e.g., "the happy man", and nouns e.g., "the wind speed is 9 mph". Our sample of DBPedia and Freebase binary relations show that about 35.5% of the relations cannot be expressed by a combination of verbs and prepositions alone; future work can address how the missing types can express these relations.
- We consider only a small subset of interrelationships between typed verbs, of the kind *begin-relation*(X) → *end-relation*(Y), as in *begin-spouse*("marry"(*person*, *person*)) → *end-spouse*("divorce"(*person*, *person*)), *begin-yearsActive*("be born in"(*person*, *year*)) → *end-yearsActive*("die in"(*person*, *year*)). Future work can address more general interrelation-

ships and inferences such as “hire”(*company, person*) → “work for”(*person, company*) → “be appointed as”(*person, ceo*) → “lead”(*person, company*).

- We only consider its subject and object fillers and their types for each typed verb. These correspond roughly to the *Agent* and *Patient*/*Theme* of the verb that may not provide enough information to adequately differentiate the verb. For example, in VerbKB the relation ***personEatFood*** has verbs that should not all become one cluster: “eat”, “finish”, “consume”, “digest”, “ingest”, “devour”, “gobble”, “wolf”, “chew with”, “pig on”, etc. Future work can bring into consideration other semantic roles like *Instrument*: “eat with fork”, “chew with teeth” and *Manner*: “gobble fast”, “ingest slowly” that can better differentiate the verbs in the cluster.
- We only consider verb lexemes for classification. Thus we lose the distribution of tenses of the verb patterns, which can be useful for classifying verb patterns with respect to the internal temporal structure of the events they denote. For example, if the simple present tense can be used by a verb pattern to refer to the actual present *and* if the verb pattern cannot appear in the progressive, then the event being described is a state e.g., we can say “Kim knows the answer” (interpreted as referring to what Kim knows now; a state) but we cannot say “*Kim is knowing the answer”. Future work can bring into consideration the distribution of temporal modifiers and of tenses based on the work of the aspectual classification of verbs by Vendler [1957] to, for example, better identify change-of-state or nonstative verb patterns and to link them to changes in the knowledge base relations.

  We also observed in Wikipedia edit histories that a change in the verb tense is often a good indicator of a change in the infobox relation. For example, in *Duterte*’s (the president of the Philippines) Wikipedia page, the sentence “Duterte *is due to take office on* June 30, 2016” is changed to “Duterte *took office on* June 30, 2016” on the day he was sworn into office. Similarly, the sentence “He *will be* the first Mindanaoan president of the country” is changed to “He *is* the first Mindanaoan president of the country” on that day. Correspondingly, the word “elect” is removed from his ***office*** infobox as the phrase “6th President of the Fifth Republic” is added. Future work can take tenses into consideration as features for learning the mapping from verbs to changes in relations.
- In our typed verbs, we do not differentiate between the preposition that is verb-specific – i.e., the preposition that signals the *argument* of the verb e.g., the preposition “against” in the sentence “she played *against* Kim in the final” – and the preposition that can also work in a similar role with other verbs – i.e., the preposition that signals the *adjunct* of the verb e.g., the preposition “since” in the sentence “she has played *since* 1999”, which signals a more *verb-general* notion of temporal scope. Future work can consider distinguishing the two uses of the preposition and generalizing over prepositions that signal adjuncts in a more general construct e.g., “*V* since”(*t, date*) where *V* is any durative verbs and *t* is any type.

Thesis Terminology In this thesis, whenever we mention *typed verb*(*s*), we are referring to the verb lexical unit(s) in VerbKB e.g., “divorce”(*person, person*), “live in”(*person, location*), “die at"(*person*, *nonNegInteger*), "die on"(*person*, *date*), "play since"(*person*, *year*), etc.

On the other hand, when we mention *verb pattern*(*s*) or *verbs* (in plural form), unless otherwise specified, we are referring to the verb lexical unit(s) in VerbKB *without* types e.g., "divorce", "live in", "die at", "die on", "play since", etc.

Furthermore, when we mention *relation*(*s*) in the context of knowledge bases, we are referring to binary relations – i.e., relations with two arguments.

1.2 Motivating Assumptions and Hypothesis

Modern theories of grammar see sentences as consisting of predicates and their arguments, where predicates are functions over the arguments. Since verbs are also functions over their arguments and adjuncts in sentences, this representation suggests that the verbs and their auxiliaries in sentences are predicates and the noun phrases that they appear with are their arguments or adjuncts. Similar to the computational linguistics' definition of the term "meaning" as a mapping between a linguistic sign and some non-linguistic entity to which the sign refers: a concept in the mind or an entity in the world [Ovchinnikova, 2012], natural language understanding (NLU) is this task of mapping natural language text to its meaning representation: the entities and their relations that are anchored in the world. In this thesis, since verbs are predicates in sentences, we believe that verbs are an important link between the *surface* form of a sentence (i.e., the predicate-argument linguistic construction) and the *meaning* of the sentence (i.e., the non-linguistic entities in the world and the relations among them).

Constructing a *high* coverage verb resource is important because such a resource can provide the predicates necessary to define *most* relations in the world to represent the meaning of sentences. Even when the verbs in sentences are implicit e.g., in noun compounds, possessives, or prepositional phrases, these can be rephrased to include verbs. For example, noun compounds such as "sleeping pill" v.s. "headache pill" can be rephrased to include verbs [Wijaya and Gianfortoni, 2011]: "pill that *causes* sleep" v.s. "pill that *alleviates* a headache". Possessives such as "Shakespeare's tragedy" v.s. "Shakespeare's house" can be rephrased to "the tragedy that is *written* by Shakespeare" v.s. "the house where Shakespeare *lives*". Prepositional phrases such as "John in the house" v.s. "John in anger" can be rephrased to "John is *located* in the house" v.s. "John is *feeling* anger".

Assuming that the vast majority of relations in the world needed to represent the meaning of arbitrary text can be defined as verb predicates of the form $g(a_1, ..., a_N)$ where a_i is the semantic type for the i^{th} argument of the predicate and that g is a group of semantically similar verbs, our first motivating assumption for building VerbKB is that *a high coverage verb resource that contains the mappings from typed verbs to these predicates can provide a high coverage vocabulary of relations to represent the meaning of arbitrary text.*

Our analysis of relations in the large-scale DBPedia, a large-scale knowledge base containing relations from Wikipedia infoboxes [Lehmann et al., 2015] and Freebase [Bollacker et al., 2008] knowledge base appears to validate this assumption.

The analysis of samples: 10% of (and most frequent) relations from DBPedia and 10% of (and randomly sampled) relations from Freebase shows that indeed the vast majority (98%) of relations in DBPedia and Freebase can be described by typed verbs[4]. Given that a vast majority of these knowledge base relations are describable by typed verbs, a high coverage resource that maps typed verbs to the predicates that define these relations can provide the high coverage of predicates needed to represent the meaning of an arbitrary text.

Given these high coverage verb predicates, our second motivating assumption is: *that in order to further facilitate natural language understanding, which is the task of mapping natural language text to its meaning representation – entities and relations in the world, we need to link these verb predicates to knowledge base relations*.

The benefits of linking to the knowledge base are many. In order to represent the meaning of a natural language text, it is often not enough to know only the lexical meaning of the words. Knowledge and reasoning about the entities and the relations in the world to which the text refers may be needed to understand the text. Problems in understanding the text such as reference resolution, interpretation of noun compounds, resolution of syntactic ambiguity, are some of the problems that may require world knowledge for their resolutions; and some of these world knowledge may already exist in the knowledge bases or can be inferred from them. Knowledge bases such as NELL, YAGO, Freebase, or ConceptNet [Liu and Singh, 2004] provide ontologies that capture world knowledge. The structured knowledge in these knowledge bases also enables inference engines to reason and form inference rules about the knowledge [Lao et al., 2011, Gardner et al., 2014, Haarslev and Möller, 2003, Liu and Singh, 2004].

Some typical natural language phenomena that require world knowledge and reasoning to accurately represent their meanings include:

(a) syntactic ambiguity e.g., “Jen looked at the *man* with a telescope” v.s. “Jen looked at the *beach* with a telescope” [Nakashole and Mitchell, 2015]. Here, it is not immediately clear just from the syntax of the sentences that the first sentence is ambiguous while the second is not, since knowledge that a man can carry a telescope but cannot a beach is world knowledge. Knowledge and inferences in knowledge bases over the categories of entities that a *person* can ***carry*** can help derive the meaning representations of these sentences. For example, that the first sentence is ambiguous and can either be represented by this set of predicates: $\{$*look_at*$_1$(Jen, man), *carry*$_2$(man, telescope)$\}$or this set of predicates: $\{$*look_at*$_1$(Jen, man), *look_with*$_1$(Jen, telescope)$\}$ while the second sentence can be represented by this set of predicates: $\{$*look_at*$_1$(Jen, beach), *look_with*$_1$(Jen, telescope)$\}$[5]

(b) anaphoric resolution e.g., “Jen gave the bananas to the monkeys because they were *hungry”* v.s. “Jen gave the bananas to the monkeys because they were *ripe”*. Here, to resolve that *they* in the first sentence refers to the monkeys while *they* in the second sentence refers to the bananas, we need knowledge of typical categories of entities that can be hungry and typical categories of entities that can be ripe. Knowledge in knowledge bases about *animal* i.e., that it can ***feel*** hungry; and knowledge about *fruit* i.e., that it can ***be*** ripe can help derive the meaning rep-

[4] The relations that cannot be described by typed verbs are relations that are specific to DBPedia and Freebase system design such as ***careerStation*** in DBPedia, which is a placeholder for holding information related to an athlete’s various relations (***matches***, ***goals***, etc) during a specific time span. On top of that, there are relations with TRUE/FALSE values such as ***coating?*** (i.e., whether a drug has a coating or not) or relations whose values are positions in a sequence such as ***presidentNumber*** e.g., ***presidentNumber***(Abraham Lincoln,16) – Abraham Lincoln is the 16th president of the United States

[5] the subscript on the predicate in these examples is the Wordnet synset ID

resentations of these sentences. For example, that the first sentence can be represented by this set of predicates: $\{$*give*$_3$(Jen, bananas), *give_to*$_3$(Jen, monkeys), *feel*$_4$(monkeys, hungry)$\}$ while the second sentence can be represented by this set of predicates: $\{$*give*$_3$(Jen, bananas), *give_to*$_3$(Jen, monkeys), *be*$_1$(bananas, ripe)$\}$.

In these two examples, knowledge and inference about entities and relations to which the text refer are crucial for its interpretation. One way to access such knowledge is to enrich the predicates used to represent the meaning of the text with world knowledge and inference capabilities of some knowledge bases; by linking these *verb predicates* to *relations* and their *arguments* or *adjuncts* to *entities* in some knowledge bases. Such links will allow users of the verb resource to tap into the knowledge, inference rules, and inference/reasoning capabilities that come with the knowledge base. The knowledge base, on the other hand, can also benefit from the verb resource. The knowledge base can use the verbs to extract more instances of its relations or update their values, as well as to extend its ontology of relations with the high coverage vocabulary of relations provided by the verb predicates. For the knowledge base, extracting more relations and instances can mean denser knowledge graph that can lead to better inferences [Gardner et al., 2013].

Unfortunately, existing lexical semantic resource about verbs are limited in their mappings from verbs to knowledge bases. Some existing resources classify verb lexemes into semantic classes manually (e.g. WordNet) or classify verbs automatically (e.g. DIRT [Lin and Pantel, 2001a]). However, they have no direct links from these verb classes to relations in knowledge bases. Other resources provide a basis for defining semantic relations between verb lexemes and their arguments in terms of semantic roles (e.g. PropBank Kingsbury and Palmer [2002], VerbNet [Kipper et al., 2006], FrameNet [Baker et al., 1998]) where verb lexemes express frames, with a separate set of roles defined for each frame. However, they have no direct links from the verb classes or frames to relations in knowledge bases. There is also no direct mapping from the semantic roles, which are labels in natural language form e.g., *Agent, Patient, Theme,* to entities/categories in knowledge bases.

Furthermore, existing verb resources that are semantically richest: WordNet, VerbNet or FrameNet are manually constructed, making scalability a challenge. Knowledge bases have thousands of real-world relations that are expected to grow over time. Manually annotating verbs to map to the growing vocabularies of real-world relations is expensive. A verb resource that maps verbs to relations in knowledge bases should be automatically constructed and grow in coverage with the knowledge bases, leveraging on corpus statistics from large corpora such as the Web to learn high coverage mappings from verbs to relations.

Our hypothesis is, therefore, that: *we can semi-automatically construct a verb resource that goes beyond current resources in terms of coverage and links to knowledge bases, by leveraging a combination of high coverage text corpora, a knowledge base with a rich type system over entities, and other pre-existing linguistic resources such as a thesaurus and WordNet.*

In order to construct a precise and high coverage verb resource that maps verbs in the corpora to relations in the knowledge base, we leverage (1) an existing linguistic resource in the form of a large-scale English thesaurus[6][7] that contains over 2.5 million synonyms of English words, (2)

[6] Moby Thesaurus: http://moby-thesaurus.org

[7] Project Gutenberg: http://www.gutenberg.org/ebooks/28900

a large knowledge base of entities and relations connected to the real-world such as DBPedia and NELL (whose English knowledge base contains over 2.7 million instances of 1.1k different categories and binary relations), and (3) corpus statistics of over 650 million unique subject-verb-object (SVO) triples occurring a total of over 2.1 billion times in a large corpus of over 1 billion web pages in ClueWeb[8] [Callan et al., 2009b].

1.3 Road Map and Summary of Key Results

In Chapter 2 of the thesis, we discuss existing lexical semantic resources and point out where information relevant for natural language understanding is still missing. We will also discuss the importance of mapping lexical resources to knowledge bases and compare our approach and verb resource to existing verb resources.

In Chapter 3 of the thesis, we present an algorithm to learn the mappings from typed verbs to the NELL knowledge base relations e.g., to map the typed verb "marry (*person*, *person*)" to the NELL relation ***hasSpouse*** (*person*, *person*). The algorithm learns the mappings by aligning the typed verbs and knowledge base relations with corpus statistics acting as an *interlingua* that links the typed verbs and the relations. To demonstrate that the algorithm we propose can be straightforwardly applied to compute the mappings from typed verbs in languages other than English to relations from other knowledge bases, we use the same algorithm to map typed Portuguese verbs to the Portuguese NELL relations [Duarte and Hruschka, 2014] using Portuguese SVO triples as an interlingua. Since Portuguese NELL is much smaller than that of English NELL i.e., English NELL has extracted more relation instances and semantically typed more entities than Portuguese NELL; we also explore a method that adds relation instances from English NELL to Portuguese NELL to improve the coverage of the typed Portuguese verbs to relations mappings.

We show in the experiments, the effectiveness of the mappings from typed verbs to knowledge base relations to extract more instances for the knowledge base relations [Wijaya and Mitchell, 2016] in English NELL and Portuguese NELL. We also find that adding knowledge i.e., relation instances from English NELL to Portuguese NELL helps in improving the coverage of the typed Portuguese verbs to the Portuguese NELL relations mappings.

In Chapter 4 of the thesis, we present an algorithm to learn the mappings from verbs to *changes* in knowledge base relations (initiations or terminations of relations) that relate to *person* entities in DBPedia e.g., to map verbs: "divorce" and "separate from" to the termination of the DBPedia relation ***spouse***. The algorithm learns the mappings from correlated Wikipedia article text and its infobox updates. To the best of our knowledge, there is no pre-existing resource for verbs that automatically maps verbs to changes in knowledge base relations. We also show in the experiments, that these mappings from verbs to changes in relations in the infobox are effective for predicting Wikipedia infobox updates when the verbs are added or deleted from the corresponding Wikipedia article text [Wijaya et al., 2015].

In Chapter 5 of the thesis, we present an algorithm to extend the vocabulary of relations in the knowledge base. Since not all typed verbs have mappings to the existing NELL relations, the

[8] http://lemurproject.org/clueweb09.php/

algorithm extends the vocabulary of relations in NELL by automatically clustering typed verbs into synonym sets (synsets) and proposing synsets that are not mapped to the existing NELL relations as new relations. Although trivially we can extend the vocabulary of relations in NELL by adding every typed verb as a new relation in NELL as in the OpenIE fashion [Fader et al., 2011], it has been shown that there is value in using clusters of semantically similar surface forms rather than just individual words for improving performance; in tasks such as knowledge base inference [Gardner et al., 2015] or word embedding for dependency parsing [Ammar et al., 2016].

To create clusters of typed verbs, we group typed verbs (that are extracted from ClueWeb's SVO triples and typed with NELL's semantic types) into similarly typed and semantically similar clusters based on (1) the types of the subjects and objects of the verbs and the verbs' selectional preference, (2) their similarities based on their shared subject-object pairs in the SVO triples, (3) their synonymy and antonymy constraints from Moby thesaurus and WordNet[9]. Then, each cluster is either mapped to a pre-existing NELL relation (based on the overlap of the typed verbs learned for the relations in Chapter 3 and the typed verbs in the cluster) or added as a new relation in NELL.

In terms of alignment with the coarse-grained WordNet senses induced by the Oxford Dictionary of English (ODE) inventory [Navigli et al., 2007], our verb clusters produce the best alignment to these manually constructed verb clusters, compared to the verb clusters in other automatically constructed large-scale resources such as PATTY [Nakashole et al., 2012] or PPDB [Cocos and Callison-Burch, 2016]. In terms of running time, it took only a total of 7 hours to generate the clusters for VerbKB. This in contrast to some of the automatically constructed resource such as [Kawahara et al., 2014] that took up to three days to construct.

We integrate all the semantics our algorithms have learned about verbs in Chapter 3, 4, and 5 into VerbKB that we publicly release[10]. VerbKB contains 65k unique verb patterns mapped into 200k binary relations, each typed with semantic categories in NELL. A typed verb in VerbKB can be mapped to more than one relation, each expressing a particular verb sense and the verb's subject and object types. To the best of our knowledge, this is the largest knowledge base of English verbs to date. As a comparison in Table 1.1, WordNet has 11k verb lexemes and 13k verb synonym sets. VerbNet has 6k verb lexemes that are categorized into 280 classes according to the syntactic frames they can appear in. Automatically constructed polysemy-aware verb classes introduced in [Kawahara et al., 2014] have 1.6k verb lexemes categorized into over 840 classes according to their syntactic frames. PATTY [Nakashole et al., 2012], which is a large-scale semantically-typed resource of relational patterns mined from Wikipedia, contains 12k unique verb patterns over 187k binary relations, each typed with semantic categories in YAGO.

1.4 General Design Choices

We make several design choices in building VerbKB. The first is that the VerbKB's lexical units contain verb patterns, which are lemmatized verb phrases that match this part-of-speech-based regular expression: V | VP where V is a verb lexeme and P is a preposition. This limits the verb

[9] We use only synonymy and antonymy relation from WordNet, not the synsets.

[10] http://www.dwijaya.org/dvkb.html#DKVB

	Size of English Lexicon	**Number of Verb Groups**
VerbKB	65k unique verb patterns	215k
PATTY	12k unique verb patterns	187k
WordNet	11k unique verb lexemes	13k
VerbNet	6k unique verb lexemes	280†
Kawahara's	1.6k unique verb lexemes	840†

Table 1.1: Size comparison between VerbKB and other resources, some resources (†) categorize its lexicon into groups according to syntactic frames

pattern in our VerbKB to be either a verb (e.g., "move") or a verb followed by a preposition (e.g., "move to"). We also require that the verb pattern appears between its subject and object (the noun phrases/NPs) in the sentence.

Secondly, we extract and lemmatize verb patterns that match this regular expression from ClueWeb SVO triples and Portuguese SVO triples. We use the Stanford CoreNLP lemmatizer to lemmatize ClueWeb (English) SVO triples. To reduce some of the Portuguese verb pattern inflections and generalize over the verbs and the prepositions, we use the LemPORT [Rodrigues et al., 2014] lemmatizer to lemmatize Portuguese SVO triples and expand contracted prepositions e.g., "nas" to "em as" or "pelos" to "por os".

For example, given the SVO triple "John ate noodles with chopsticks" we extract the verb pattern "eat" with its subject "John" and its object "noodles", *and* the verb pattern "eat with" with its subject "John" and its object "chopsticks". We indicate when the verb pattern is in a passive voice by a special indicator: *(passive)*. For example, given the SVO triple "the enemy was defeated by our troops", we extract the verb pattern "(passive) defeat by", which means "to be defeated by". In tasks where ClueWeb SVO triples are not used e.g., in the task of mapping verbs to changes in DBPedia relations where Wikipedia articles are used as a source corpus instead of ClueWeb (Chapter 4), we use the Stanford CoreNLP dependency parser to parse Wikipedia articles and extract verb patterns (lemmatized verb phrases that match our part-of-speech-based regular expression: V | VP) and their subjects and objects.

Thirdly, we derive the type signatures of the verb patterns by labeling the subjects and objects of these verb patterns with NELL's categories for the noun phrases. We use a list of NELL's category labels for millions of noun phrases, which is available for download from http://rtw.ml.cmu.edu/rtw/nps. We also use NELL's category labels for noun phrases that are in NELL's knowledge base, which is available for download from http://rtw.ml.cmu.edu/rtw/resources.

NELL currently has 290 semantic categories that are organized into a subsumption tree with a height of 7. At the root of the tree is the category *everyPromotedThing* whose children include the categories: *abstractThing*, *agent*, *geoLocatableThing*, *item*, *location*, and *visualizableThing*. At the leaf of the taxonomy tree, an example of leaf categories that is deepest in the tree is the category *politicianUs* with a subsumption hierarchy: *politicianUs* → *personUs* → *personNorthAmerica* → *personByLocation* → *person* → *humanAgent* → *agent* → *everyPromotedThing*. NELL's semantic categories and the entities that NELL has learned to belong to these categories is available for download and can be browsed at NELL's website:

http://rtw.ml.cmu.edu/rtw/kbbrowser/

Dealing with Noise Since the NELL knowledge base is automatically constructed, it is noisy and NELL's category labels for noun phrases may contain erroneous labels. For example, NELL labels the noun phrase "underground coal miners" with category *sportsTeam* with a 0.52 confidence. To reduce noise from erroneous labels, we make a design decision to select only labels with confidence of at least 0.9.

Secondly, SVO triples extracted from ClueWeb may contain noise: unrepeatable triples that are erroneous due to segmentation/parsing errors e.g., "smells notice decline" or outliers: triples that are unrepeatable because they are too specific e.g., "Harry ran into the kitchen". To reduce noisy/outlier triples, we make a design decision to select only SVO triples that occur at least twice in ClueWeb.

Thirdly, for the task of mapping typed verbs to knowledge base relations, we focus only on mapping typed verbs that are *informative* for a relation. We define informative typed verbs for a relation based on how often the typed verbs co-occur with entity pairs that match the relation's type (domain and range) in the corpus. For example, to find informative typed verbs for the relation ***hasSpouse***, we look among typed verbs that co-occur frequently with entity pairs that match ***hasSpouse*** type, which is (*person*, *person*).

To adjust for the fact that some verbs appear more frequently in general (e.g., the verbs "make", "be"), instead of using raw co-occurrence counts to sort the informativeness of typed verbs for a relation, we compute *tf-idf* scores of each verb pattern *v* that matches our part-of-speech-based regular expression, for each type signature *t* = (t_1, t_2). For example, we compute the *tf-idf* score of the verb pattern "drive" for the type signature (*person*, *vehicle*).

The term frequency (*tf*) is the relative frequency of the verb pattern *v* for the type signature *t* – computed by dividing the total number of times *v* occurs with entity pairs of type *t* by the total number of times *any* verb pattern occurs with entity pairs of type *t*. Inverse document frequency (*idf*) is the log of the inverse fraction of type signatures that the verb pattern has – computed by dividing the number (not the frequency) of distinct type signatures in the corpus by the number (not the frequency) of distinct type signatures the verb pattern has.

tf-idf is a product of *tf* and *idf*. The *tf-idf* score of a verb pattern for a type signature increases proportionally to the number of times the verb pattern occurs with entity pairs of that type, but is offset by how often the verb pattern occurs generally – with entity pairs of *any* type – in the corpus. This helps to adjust for the fact that some verbs appear more frequently in general. For example, the verb pattern "make" occurs frequently across diverse types: (*person*, *product*), (*organization*, *product*), (*person*, *organization*), (*organization*, *organization*), (*person*, *creativeWork*), etc. while the verb pattern "die of" occurs frequently only for a specific type: (*person*, *physiologicalCondition*).

The *tf-idf* scores are used to sort the informativeness of typed verbs for a relation. To scale our mapping algorithm to a large text corpus, we put a threshold on the number of typed verbs we consider as potentially being mapped to the relation from this sorted list. The specific design choices are detailed in the corresponding chapter of the mapping algorithm (Chapter 3).

Chapter 2

Related Work

In this section, we discuss existing lexical semantic resources and point out where information relevant for natural language understanding is still missing. We would also discuss the importance of mapping lexical resources, which contain semantic knowledge about words, and knowledge bases, which contain conceptual knowledge about entities and relations anchored to the world.

2.1 Lexical Semantic Resources

In discussing lexical semantic resources which contain meaning representation for words, it is useful to talk about meaning representation itself to understand what representations are suitable for expressing linguistic meaning, which information should be included, and how it can be constructed. Meaning representation in linguistic theories can be discussed in terms of these three frameworks [Ovchinnikova, 2012]: *formal semantics*, *lexical semantics*, and *distributional semantics*. As a running example, we will use the verb lexeme "lend" and show how each framework represents knowledge about this lexeme when available.

2.1.1 Formal Semantics

Formal semantics mainly focus on the logical properties of natural language – rules that allow translating the syntax (surface structures) to semantics (logical forms) in a compositional way. For example, the sentences "The person lent the student money" is assigned to a first order logical representation: $\exists p, s, m, e(person(p) \wedge student(s) \wedge money(m) \wedge lend(e, p, m, s))$, where the first argument of every verb predicate such as "lend" is an event variable, the second argument is a prototypical agent, the third argument is a prototypical theme/patient, and the fourth argument is a prototypical goal/recipient.

However, this representation only concentrates on the logical features expressed by function words (e.g., *and*, *if*) while the meaning of the non-logical features expressed by content words (e.g., *person*, *lend*, *student*, and *money*) are mapped to atomic predicate names. The criticism towards this approach is that many natural language phenomena require more knowledge for their resolution than only logical structure. For example, the sentences "a person plays a guitar” and ‘a person plays a football game” is mapped to the same logical structure $\exists x, y, e(P(x) \wedge Q(y) \wedge R(e, x, y))$. This structure as it is does not indicate that the verb lexeme “play” indicates two different relations when the verb lexeme in the two sentences are mapped to the same atomic predicate ***play***. The mapping to atomic predicate names offers little in terms of generalization. For example, that the relation ***play*** in the first sentence is semantically similar to the relation ***strum*** in the sentence “a musician strums a guitar”. Formal semantics can easily be extended so that the first sentence’s predicate is mapped to the relation ***play1*** while the second sentence’s predicate is mapped to the relation ***play2***. The relations can then be expanded into additional predicates and ***play1*** can cluster with the relation ***strum***. However, this extension requires knowledge beyond logical features.

2.1.2 Lexical Semantics

Most existing lexical semantic resources can be discussed in terms of the *lexical semantics* meaning representation. Lexical semantics mainly focus on the organization of lexicons into groups (word senses or verb classes or frames) and the semantic links between these groups (hyponymy, meronymy, antonymy, causation, inheritance, temporal precedence, etc.). The main paradigms underlining the lexical semantics meaning representation are a definition-based model of lexical meaning, a prototype-based model of lexical meaning, and the aspectual approaches to lexical semantic representation.

Definition Model of Lexical Semantics. The definition-based model involves decomposing lexical meaning into *semantic markers* – atomic units of meaning and conceptualization. A handcrafted verb resource that is based on this decomposition approach is VerbNet [Kipper et al., 2000]. VerbNet lists over 6k verb lexemes that are categorized into 280 classes according to the syntactic frames they can appear in. All verb lexemes in the same class appear in the same set of syntactic frames. The classification into these classes is based on Levin’s classification of verb lexemes [Levin, 1993], which is motivated by the notion that verb lexemes that are semantically similar also have similar syntactic realizations (the Semantic Consistency Hypothesis that there is some set of semantic features such that verb lexemes that share the same syntactic behavior can be identical along those features).

For example, the verb lexeme “lend” in VerbNet belongs to the class `give-13.1`, which contains 7 verb lexemes and 4 syntactic frames. The entry for one frame is shown below:

Syntactic Frame NP V NP PP.Recipient
Example “They lent a bicycle to me”
Syntax Agent V Theme {TO} Recipient
Semantics HAS_POSSESSION(START(E), Agent, Theme)
HAS_POSSESSION(END(E), Recipient, Theme)
TRANSFER(DURING(E), Theme)
CAUSE(Agent, E)

As seen in this frame, the semantics are expressed through a conjunction of the semantic predicates: HAS_POSSESSION, TRANSFER, CAUSE. The semantics listed here are not just for the verb lexeme, *lend* but also apply to all verb lexemes from `give-13.1` (e.g., deal, loan, refund, etc) whenever they appear in that syntactic frame. However, at the moment semantic predicates in VerbNet are just labels – they are not axiomatized or linked to any formal theory [Ovchinnikova, 2012]. A complementary resource to VerbNet that links verbs to knowledge base relations can be one step towards filling this gap. For example, one can envision a method for linking the semantic predicates in VerbNet to relations in the knowledge base, by linking the verb lexemes to the relations in the knowledge base and the arguments of the verb lexemes to entities in the knowledge base. The semantic predicates that the arguments participate in can then be mapped to relations that the entities participate in the knowledge base.

Such a mapping can be beneficial for both VerbNet and the knowledge base. The mappings from arguments to entities and predicates to relations can be used to verify and validate the semantic predicates encoded in VerbNet in the same spirit as the VerbCorner project [Hartshorne et al., 2014]. In addition to the crowd-sourcing effort of VerbCorner, one can envision using semantic entailment approaches to verify the semantic entailments of verb lexemes using knowledge about the verb lexemes' arguments in the knowledge base as features e.g, what relations the arguments have in the knowledge base, whether the arguments have changed state in the knowledge base; combined with features of the text from which the knowledge base relations are extracted, such as its syntactic parse, dependency path features, etc.

Conversely, the mappings between the semantic predicates in VerbNet and relations in the knowledge base can be useful for knowledge base updates particularly for change of state verb lexemes whose predicates can inform the knowledge base of what relations to update with what values. For example, the verb lexeme "lend" when appearing in the above syntactic frame can be used as a trigger in the knowledge base to terminate (among which) the ***hasPossession***(*person*, *thing*) relation between entities that are the Agent and the Theme and to initiate the ***hasPossession***(*person*, *thing*) relation between entities that are the Recipient and the Theme.

Prototype Model of Lexical Semantics. The prototype-based model of lexical meaning involves representing meaning via a prototype – a structure of concepts underlying lexical meaning, an example of which is Frame Semantics [Fillmore, 1967] that considers lexical meanings to be related to prototypical situations captured by *frames* – structures of related concepts. A hand-crafted lexical semantic resource that is based on this prototype approach and supported by corpus evidence is FrameNet [Ruppenhofer et al., 2006].

The lexical meaning in FrameNet is expressed in terms of frames, which are supposed to describe prototypical situations spoken about in natural language. Every frame contains a set of roles corresponding to the participants of the described situation.

For example, the verb lexeme "lend" is part of the LENDING frame and has *Borrower*, *Lender* and *Theme* as core roles. These roles are more specific and often referring to concrete scenarios. For example, for the verb lexeme "lend" FrameNet assigns *Lender* instead of *Agent* in VerbNet and *Borrower* instead of *Patient* in VerbNet. Also, in contrast to VerbNet, FrameNet assigns these roles not only to verb lexemes but also nouns, adjectives, adverbs, and prepositions. For example, the noun "loan" is also part of the LENDING frame. In addition, FrameNet introduces semantic relations between frames e.g., the BEING_BORN and DEATH frames are connected by the "precede" relation. This feature opens a range of new reasoning options and can also be useful for paraphrasing.

Similar to VerbNet, FrameNet also gives syntactic realization patterns of frame elements based on corpus evidence e.g., the role *Borrower* in the frame LENDING is most frequently filled by a noun phrase in the indirect object position. Similar to VerbNet, FrameNet does not yet have links that connect the frame and its roles to the relation (or the set of relations) and categories/entities in knowledge bases. A resource that links verbs to relations a knowledge base can be one step in this direction.

Lexical Semantic Relations. As an alternative to having to define a set of labels or roles (i.e., semantic primitives) to define meaning, this approach represents meaning as a network of relationships between word senses. A hand-crafted lexical semantic resource that is based on this approach is WordNet [Miller et al., 1990]. In WordNet, lexical-semantic knowledge is represented in a network-like structure. Nouns, verb lexemes, adjectives, and adverbs and their synonyms are grouped into synonym sets (synsets) which express word senses.

For example, the verb lexeme "lend" is in several verb synsets ({lend, impart, bestow contribute, add, bring}, {lend, loan} etc.), each referring to to the different senses of "lend": the first sense here refers to "lend" in the sense of "to bestow a quality on", while the second sense refers to "lend" in the sense of "to give temporarily" or "to let have for a limited time".

Semantic relations such as hypernymy, hyponymy, meronymy, antonymy, etc. are defined among synsets and words. For example, the direct hypernym of the second sense of "lend" above is the verb synset {give} as in "to transfer possession of something concrete or abstract to somebody".

Synsets in WordNet, however, are often criticized for being too fine-grained to enable automatic word sense classification [Agirre and De Lacalle, 2003]. Similar to VerbNet and FrameNet, WordNet also does not have links from its word senses to relations in knowledge bases.

Aspectual Approaches to Lexical Semantic Representation. As verbs denote events that take place in time, verbs can be differentiated according to how the events they denote take place in time – i.e., their aspectual notions. The first of these aspectual distinctions is stativity vs. dynamicity – i.e., the distinction of verbs into those that express *states* (events that do not involve change e.g., "hate", "know", "be yellow") vs. those that express *events* (events that involve change e.g., "run", "reach", "break", "hit").

Events or nonstates can further be divided into several subclasses based on their temporal feature: durativity vs. punctuality – i.e., the *durative* vs. *punctual* events.

Durative events can be subdivided based on their telicity – i.e., whether or not they have a culmination, telos, or endpoint – into *activities* (events that take time but have no inherent temporal endpoint e.g., "run", "wipe", "pour") or *accomplishments* (events that take time *and* have an inherent endpoint or *telos* at which a result state comes about e.g., "fill", "clean", "draw").

Punctual events can be subdivided into *semelfactives* (events that take no more than a moment in time [Engelberg, 1999] but like *activities*, do not have a result state that follows e.g., "hit", "hop", "wink") or *achievements* (events that take no more than a moment in time and at which, like *accomplishments*, there is a transition to a result state e.g., "explode", "break").

Thus the major aspectual classes of verbs are: *states*, *activities*, *accomplishments*, *achievements*, and *semelfactives*. The various aspectual classification systems make the same distinctions, collapsing some or subdividing others [Vendler, 1957, Dowty, 1979] .

To determine which aspectual class a verb belongs to, a variety of diagnostics have been proposed. For example, the test for aspectual verb categories proposed by Dowty [1979] includes the test for the kinds of preposition that the verb can take. For example, *achievement* verbs cannot occur with the preposition “for” when it indicates the duration of the events e.g., “*John *breaks* the window for an hour”. The tense of the verb can also be an indication of its aspectual class. For example, in *stativity* tests [Lakoff, 1966], one of the indicator of stative verbs is that they cannot appear in the progressive e.g., “*I am knowing the answer”.

In this thesis, we try to learn verbs that belong to the *events* (or nonstates) aspectual class and to link them to changes in the knowledge base relations. To the best of our knowledge, there is not yet a verb resource that learns this linking automatically. However, because we lemmatize verbs before linking them, we lose some of their aspectual features such as their tenses that can be useful for disambiguation. Future work can explore whether the existing aspectual diagnostics – such as those proposed by Dowty [1979] or Lakoff [1966] – can be used, for example, as constraints, to learn better linking between verbs and changes in the knowledge base relations.

2.1.3 Distributional Semantics

Distributional semantics’ representation of meaning is based on the quotation: ”You shall know a word by the company it keeps” [Firth, 1961] – where lexical meaning is obtained from the distributional properties of words: ”words which are similar in meaning occur in similar contexts” [Rubenstein and Goodenough, 1965]. A lot of automatically constructed lexical resources are developed out of this idea, for example, DIRT [Lin and Pantel, 2001b], PPDB [Ganitkevitch et al., 2013], Polysemy-Aware Verb Classes [Kawahara et al., 2014], PATTY [Nakashole et al., 2012] and ReVerb [Fader et al., 2011].

The DIRT (Discovery of Inference Rules from Text) [Lin and Pantel, 2001b] is a collection of paraphrases automatically learned from corpora. The approach is motivated by the hypothesis that a phrase (a path between two nouns extracted from a dependency parse tree) expresses a binary relationship between the nouns. If two phrases tend to link the same sets of nouns then the meanings of the corresponding phrases are similar. For example, the phrases *X wrote Y* and *X is the author of Y* are similar, with some measure of similarity. DIRT contains around 231k unique phrases and their pairwise similarities. However, a range of different types of phrases can have similar distributional properties. Therefore, although DIRT extracts good paraphrases such as ‘imprisoned” or “jailed” for the phrase “thrown into jail” for example, DIRT also extracts phrases that are temporally or causally related like “began the trial of” or “interrogated” [Ganitkevitch et al., 2013].

Instead of extracting paraphrases from monolingual corpora like DIRT, the Paraphrase Database (PPDB) [Ganitkevitch et al., 2013] extracts paraphrases from large bilingual corpora. The idea behind the acquisition of the paraphrases is that two words in one language that align in a parallel text to the same word in a different language should be synonymous. For example, the English phrases “thrown into jail” and “imprisoned” are aligned to the same German phrase “festgenommen” therefore they should be synonymous. They found that this bilingual pivoting rarely extracts the non-paraphrases that DIRT extracts. One rationale behind this can be that bilingual pivoting can help disambiguate the semantic roles or selectional preference of the source word in its particular sense [Mechura, 2008]. For example, "jail" and "imprison" have the same core roles in FrameNet: *Authorities*, *Prison*, and *Prisoner* while "interrogate" has a very different set of core roles: *Speaker*, *Message*, *Addressee*, and *Topic*. The German phrase "festgenommen" may have disambiguated what the relevant roles or selectional preference is for the phrase "thrown in jail".

The result is a semantic lexicon containing more than 220 million ranked paraphrase pairs of English (8 millions of these are lexical: single word to single word). The paraphrase pairs are ranked using their monolingual distributional similarity scores [Chan et al., 2011] based on contexts extracted from the Google *n*-grams [Brants and Franz, 2006, Lin et al., 2010] and Annotated Gigaword corpus [Napoles et al., 2012].

The most recent release of PPDB [Cocos and Callison-Burch, 2016] includes the clustering of each phrase' paraphrases into separate sense clusters (synsets). The clustering is based on the paraphrase scores that quantify the goodness of a pair of paraphrases [Bannard and Callison-Burch, 2005], the paraphrases' similarities in terms of the foreign words that they align to in the bilingual corpora, and their distributional similarity measures based on WORD2VEC [Mikolov et al., 2013] trained on a monolingual corpus (Google News dataset).

Kawahara et al. [2014] produce a collection of polysemy-aware verb classes from verb lexeme uses in GigaWord (LDC2011T07; English Gigaword Fifth Edition) and web corpora. They do this in two clustering steps: they first cluster verb lemexe uses into verb-specific semantic frames and then cluster these semantic frames for multiple verb lexemes into verb classes. They use the Chinese Restaurant Process to cluster in both steps using a combination of dependency slots and words co-occurring with the verb lexemes as features (e.g., "dobj:bird" or "nsubj:child") in the first clustering step and slot-only features (e.g., "dobj" or "nsubj") in the second step. Clustering each lexeme's uses into its verb-specific semantic frames in the first step helps deal with verb polysemy in the second step as each verb lexeme now has multiple data points (i.e., frames) to cluster in the second step. They differ from previous works in that their clustering steps also produce verb-specific semantic frames like PropBank [Kingsbury and Palmer, 2002]. A semantic frame discovered for the verb lexeme "inform" for example, consists of *nsubj* slot (with instance words such as "i", "he" and "we"), *dobj* slot (with instance words such as "me", "you", and "us"), and *prep_of* slot (with instance words such as "decision", "this" and "situation"). However, because they do not consider lexical similarities in their second clustering step, they discover that their clusters often consist of non-paraphrases with mixed meanings. For example, the verb lexemes "need" and "say" are in the same cluster because they have a high syntactic similarity of constituting slot distributions, which are comprised of a subject and a sentential complement. The constructed lexical resource has 1.6k verb lexemes over 840 classes and takes up to three days to construct.

PATTY [Nakashole et al., 2012] is another large-scale collection of synonym sets of relational patterns harvested from text corpora. To extract the relational patterns, PATTY traverses the dependency graph of each sentence in the corpora to extract the shortest path (expanded to include adverbial and adjectival modifiers) that connects two entities in the sentence. PATTY differs from DIRT, PPDB, and Kawahara et al. [2014] in that each of its patterns has a type signature for the entities that they connect (e.g., for the pattern "first performed in", it has the type signature [person × country]. The type signatures are derived through the use of a dictionary of entity-category pairs provided by knowledge bases like YAGO, Freebase, or DBPedia.

The type signatures are useful for several other things. Firstly, typed patterns can be grouped into pattern synonym sets (synsets) and ordered into a subsumption hierarchy based not only on the overlap of entity pairs that occur with the patterns but also on the compatibility of the patterns' type signatures as an additional cue. For example, the pattern of type signature *actor* × *award* can be grouped with the pattern of type *musician* × *award*. Based on the entity pairs overlap, the pattern "covered" is subsumed by "performed" for the same type signature *musician* × *song*. Secondly, type signatures can also distinguish different "senses" of the verb pattern, which naturally helps when dealing with verb polysemy. For example, the verb pattern "covered" with the type signature *musician* × *song* is in a *different* synset than the verb pattern "covered" with the type signature *journalist* × *event*. PATTY uses frequent sequence mining to extract these syntactic-ontologic-lexical patterns efficiently on large text corpora like Wikipedia. The resulting is a collection of about 350k pattern synsets with an average accuracy of around 85% – they compute accuracy by randomly sampling the synsets and asking human judges whether the pattern synset indicates a valid relation or not.

The following work, HARPY [Grycner and Weikum, 2014], aligns PATTY synsets to WordNet synsets with a Mean Reciprocal Rank (MRR) alignment score of 0.7. A later work, RELLY [Grycner et al., 2015], organizes PATTY synsets into a higher coverage and precision subsumption graph, using Probabilistic Soft Logic modeling framework [Kimmig et al., 2012, Bach et al., 2015] to integrate the high precision knowledge about the type (and word) hierarchy in resources such as the YAGO knowledge base (and WordNet) with the noisy subsumption information in PATTY and the noisy mappings in HARPY. A manual evaluation of hypernymy links inferred by these systems shows that in comparison to HARPY and PATTY, RELLY has higher precision for both precision at the top 100 hypernymy links and at the randomly sampled 100 hypernymy links. Precision in RELLY is comparable to PATTY, but RELLY has more than four times as many hypernym links.

As PATTY's patterns are extracted from dependency paths connecting entities from knowledge bases, PATTY differs from all the other existing lexical resource in that it has direct links to knowledge bases, from its pattern synsets to knowledge base relations. However, we find that PATTY has low coverage in terms of verb patterns. A lot of its clusters contain patterns of the same typed verb, just with different (adjectival, adverbial, modifier) expansion. For example, one synset contains "is known [[num]] [[prp]] role as", "well known for [[prp]] role in", "best known for [[prp]] role on", "is known for [[prp]] role in", etc. When we consider only the typed verbs in the clusters, the average size of PATTY clusters is 1.19, which means that a lot of its clusters consist of variations of a single typed verb. We believe therefore that in terms of verb patterns particularly, PATTY clusters do not provide a lot of generalization.

Although PATTY clusters are small in terms of the verb patterns that they contain, its verb clusters are not singletons as that of ReVerb [Fader et al., 2011]. ReVerb is a lexical semantic resource that is purely textual, where each pattern is a cluster/relation by itself. For example, a textual pattern "made a deal with" is itself a relation in ReVerb.

ReVerb operates in an Open IE paradigm; it makes a pass over its corpus – requiring no manual tagging of relations nor any pre-specified relations, identifies relation phrases that satisfy the syntactic and lexical constraints and then finds a pair of NP arguments for each identified relation phrase. The resulting extractions are then assigned a confidence score using a logistic regression classifier.

The syntactic constraint specifies that every multi-word relation phrase must appear between its two arguments in the sentence and must begin with a verb, end with a preposition, and be a contiguous sequence of words. The syntactic constraint limits relation patterns to be either a verb (e.g., “invented”), a verb followed immediately by a preposition (e.g, “located in”), or a verb followed by nouns, adjectives, or adverbs ending in a preposition (e.g., “has atomic weight of”). The constraint eliminates incoherent relation phrases (e.g., “was central torpedo”) and uninformative extractions by capturing relation phrases expressed by a verb-noun combination (e.g., given the sentence “Faust made a deal with the Devil”, instead of extracting the uninformative “made”, Reverb extracts “made a deal with”).

To avoid overly-specific extracted relation phrases, the lexical constraint specifies that a binary relation phrase ought to appear with at least a minimal number of distinct argument pairs (they set it to be 20 in their experiment) in a large corpus (they use a corpus of 500 million Web sentences in their experiment).

Like other Open IE systems, due to its open-domain and open-relation nature, ReVerb is purely textual and is unable to relate the surface forms to an ontology of a knowledge base, if known in advance [Soderland et al., 2010].

2.2 Lexical vs. World Knowledge

We have already discussed how knowledge about the world is crucial for natural language understanding. To discuss possible differences between lexical and world knowledge, we will use the illustration from [Ovchinnikova, 2012] that points to the different levels of knowledge relevant for NLU.

(1) If *NP* is a noun phrase and *V* is an intransitive verb, then the concatenation *NP V* is a clause.
(2) The phrase *x wrote y* corresponds to the proposition *write(x,y)*.
(3) The proposition *write(x,y)* refers to a “text creation” event, such that *x* plays the role of *author* and *y* plays a role of *text* in this event.
(4) If something is a tragedy then it is a play.
(5) The main function of a playwright is writing plays.
(6) “Romeo and Juliet” was written by Shakespeare.

Example (1) represents a typical syntactic rule and is included in the grammar of the English language. In example (2), a surface realization of the predicate *write* is mapped to its logical form – for example, in the output of a semantic parser. In example (3), the predicate and its arguments can be mapped to the “text creation” frame and its thematic roles in a lexical resource such as VerbNet or FrameNet or can be mapped to the relation ***write***(*author*, *text*) that has argument types *author* and *text* in a knowledge base such as NELL. Example (4) is an example of a type-of or is-a relation that can be included in a lexical resource such as WordNet or in a knowledge base such as NELL, while example (5) is common sense knowledge about playwrights. This can be part of a definition of a category “playwrights” in a knowledge base ontology or learned automatically by an inference engine such as PRA [Lao et al., 2011] over the knowledge base. Example (6) is a specific fact about the world that can be part of a factual ontology containing knowledge about the ***write*** relation instances (e.g. YAGO).

It is straightforward to see the examples (1) and (2) are language dependent and belong to lexical knowledge while example (6) is not. Example (6) is a part of the knowledge about the world. Everything in between (3) and (5) is more difficult to classify, they are both language dependent and anchored to the world. For a better NLU, it is clear that we need both lexical and world knowledge. Thus, there is a need to bridge the knowledge from (2) to (3) and use inference over the network of world knowledge which can be found in knowledge bases i.e., example (3) to (6), to better interpret natural language expression.

Strictly speaking, lexical semantic resources provide information about words and not about the world. Although the generalization which these resources give (VerbNet with its verb classes, FrameNet with its frames, WordNet with its synsets, DIRT and PPDB with its paraphrases) can be seen as referring to conceptual entities, it is not clear how much inference we can do over them. Reasoning over lexical semantic resources alone has a significant shortcoming in that they imply too little structure [Ovchinnikova, 2012]. In most cases, semantic relations defined in these lexical resources are just two-place predicates: *relation_name*$(word_1, word_2)$ (where *relation_name* is, for example, semantic relations such as hyponymy, antonymy, causation, inheritance, etc.) that are difficult to use for defining complex relationships like the fact that a PERSONAFRICA (a NELL's category) is a person who has citizenship of a country that is located in Africa. Lexical semantic relations seem to be not enough for representing detailed world knowledge. A purely textual resource such as ReVerb that lacks generalization is even more limited in the reasoning that can be done over it. For example, given a sentence from a document: *Shakespeare's Romeo and Juliet is a tragedy*, reasoning over examples (4) - (6) enables us to infer that *Shakespeare* is, therefore, a *playwright*. This additional information can be useful for understanding other parts of the document, which reasoning over text alone may not allow us to do. Although methods like Universal Schema [Riedel et al., 2013] can be used to conduct inference over purely textual patterns, they have also benefited from structured data in knowledge bases as part of their schema, for example in improving the generalization of surface patterns in their schema.

In contrast to the lexical semantic resources, knowledge bases are designed for conceptualizing the world: its entities and the more complex relationships. However, being built up with lexemes makes lexical-semantic resources more applicable in NLP, while knowledge bases require an additional lexical interface to be mapped to linguistic structures [Ovchinnikova, 2012]. There is a need, therefore, to map the lexical semantic resources to knowledge bases containing a conceptual representation of the world to facilitate deep inference over the data for better natural language understanding [Soderland et al., 2010].

2.3 Summary

In terms of the meaning representation, formal semantics focus on logical features of a language, focusing mainly on functional words and compositionality while representing content words as atomic predicate names. Lexical semantics on the other hand mostly ignore the logical aspects of language or compositionality, focusing mainly on the specification of the meaning of content words. Distributional similarity represents the meaning of words via their distribution and computes compositionality from the distributional similarity of its components.

In this thesis, we use insights from lexical and distributional semantics to build a lexical

	Similarities for Synsets	**Corpora**	**Contains**	**Outputs Synsets**	**Deals with Polysemy**	**Type Signatures**	**Links to KB relations**
DIRT	lexical	mono-	rel. phrase	no	–	no	no
PPDB	lexical	bi-	paraphrase	yes	yes	no	no
Kawahara’s	syntactic	mono-	v	yes + frames	yes	no	no
PATTY	lexical + type	mono-	rel. phrase	yes	yes	yes	yes
ReVerb	–	mono-	v+n+p	no	–	no	no
VerbKB	lexical + type + constraints	mono-	v+p	yes	yes	yes + selectional preference	yes + △ relations

Table 2.1: Comparison between VerbKB and other automatically constructed, large-scale resources

resource for verbs that specifies the meaning of typed verbs by mapping the verbs to relations in knowledge bases with compatible type signatures.

To create a resource for verbs with higher coverage than any other pre-existing resources, we leverage a very large web corpus (ClueWeb) and extract verbs from over 650 million subject-verb-object (SVO) triples occurring a total over 2.1 billion times in ClueWeb.

From pre-existing resources, we learn that distributional similarities alone are not enough to identify similar verbs. For example, DIRT extracts non-paraphrases that are not synonyms but temporally or causally related because they have similar distributional similarities. Analysis of clusters in PPDB sense clusters [Cocos and Callison-Burch, 2016], where paraphrases are clustered based on monolingual and bilingual distributional similarities, reveals that verb clusters score significantly lower in terms of F-score (a harmonic mean of precision and recall of the clusters) and V-measure (a harmonic mean of homogeneity and completeness of the clusters) than clusters of other parts of speech. Similar evidence is found in word embeddings. State-of-the-art word embeddings such as WORD2VEC that are trained on bag-of-words contexts are shown to produce *verb* embeddings with qualities that are substantially lower than that for *nouns* for the task of word similarity prediction [Schwartz et al., 2016]. They found that using symmetric pattern contexts (Hearst patterns [Hearst, 1992] such as “X or Y” where X and Y are verb lexemes e.g., “run or walk”) improves verb lexeme similarity performance by up to 15%. The rationale behind the usefulness of this context type is that two verb lexemes that co-occur in a symmetric pattern tend to take the same semantic roles in the sentence, and are thus likely to be similar in meaning.

Since semantic role is important for discovering similarities between verbs, like PATTY we leverage not only the distributional similarities of the verbs based on their contexts (i.e., subject-object pairs) in the SVO triples but also their type signatures as manifestations of their semantic roles [Zapirain et al., 2013]. Different from other automatically constructed resources, however, we also leverage pre-existing hand-crafted lexical resources that are Moby thesaurus and WordNet and use relational information from these lexicons as *constraints* to encourage synonym (and respectively discourage antonym) verbs to be in the same clusters. As an additional contribution that is different from other pre-existing lexical resource for verbs, we also map verbs to *changes* in relations in knowledge bases. These comparisons between VerbKB and other existing, automatically constructed resources are summarized in Table 2.1.

Chapter 3

Mapping Verbs to Knowledge Base Relations

3.1 Introduction

As we have discussed in our motivating assumption, the mappings from verbs to knowledge base relations that enable the anchoring from text to conceptual knowledge can be useful for natural language understanding systems. However, despite recent progress in a knowledge base construction, there is not yet a verb resource that maps to these knowledge bases: one that contains verb predicates that identify knowledge base relations.

Aside from aiding natural language understanding systems, a resource that maps verbs in different languages to knowledge base relations can also be useful to the knowledge base. For example, for extracting facts from text into the knowledge bases, or to aid alignment and integration of knowledge across different knowledge bases and languages. Such a multi-lingual verb resource would also be useful for tasks such as machine translation and machine reading.

In this chapter, which is based on our previously published papers [Wijaya et al., 2013, Wijaya and Mitchell, 2016], we present an algorithm that learns the first semantics about verbs that we include in our VerbKB, namely the mappings from typed verbs to the NELL knowledge base relations e.g., the mapping from the typed verb “marry (*person*, *person*)” to the NELL relation ***hasSpouse*** (*person*, *person*).

The algorithm learns the mappings by aligning the typed verbs and knowledge base relations with subject-verb-object (SVO) triples extracted from a corpus acting as an *interlingua* that links the typed verbs and the relations. We use a scalable Expectation Maximization (EM) approach using SVO triples extracted from the very large ClueWeb text corpus [Wijaya and Mitchell, 2016]. Given a text corpus in any language and any knowledge base, the method can produce mappings from that language’s typed verbs to the knowledge base relations.

Experiments with the English NELL knowledge base and ClueWeb corpus show that the learned English typed verb-to-relation mappings are effective for extracting relation instances from English text. When applied to the Portuguese NELL knowledge base [Duarte and Hruschka, 2014] and a Portuguese text corpus, the same method automatically constructs a verb resource in Portuguese that is effective for extracting relation instances from Portuguese text.

Since English NELL has extracted more relation instances and has typed many more entities than Portuguese NELL, we explore a method that adds more relation instances to Portuguese NELL from the instances in English NELL. We use a multilingual knowledge base (DBPedia) to link the entities in English NELL to their corresponding Portuguese noun phrases. We find that adding this knowledge from English NELL improves the coverage of the typed Portuguese verbs mappings and correspondingly the recall of the relation extraction.

3.2 Motivating Study

Our work to map typed verbs to relations in the knowledge bases using web text as a kind of *interlingua* started from our earlier work on knowledge base ontology alignment. In that work, we align ontologies of two knowledge bases that share few or no data entries in common [Wijaya et al., 2013] by using corpus statistics based on the Web as an *interlingua* to link the two. For example, using corpus statistics of 650 million Subject-Verb-Object (SVO) triples from the entire ClueWeb [Callan et al., 2009a] corpus of about 230 billion tokens, we can create a graph linking the two knowledge bases (Figure 3.1).

Our approach, called PIDGIN, then performs inference over this graph to determine that KB$_1$:***bornIn*** is equivalent to KB$_2$:***personBornInCity***. PIDGIN first associates the relation nodes from KB$_1$ with seed labels i.e., self-injection. Starting with this seed information, a graph based semi-supervised learning (SSL) algorithm [Talukdar and Crammer, 2009] is used to propagate these relation labels and classify the rest of the nodes in the graph. One of the advantages of this approach for alignment is that it takes the graph structure (specified by the ontologies of resources to be aligned) and transitivity into account.

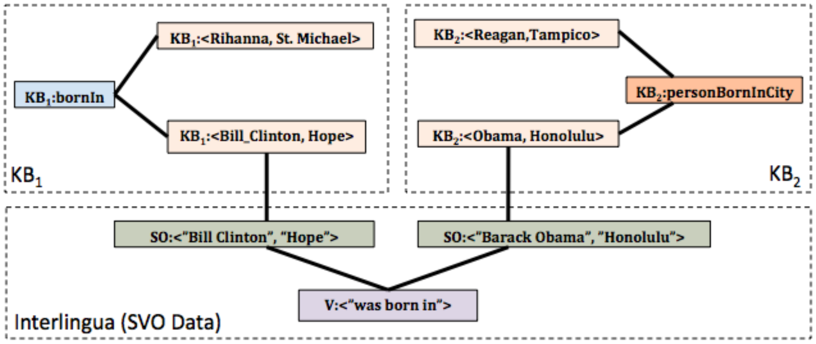


Figure 3.1: Graph construction using SVO triples as an interlingua to link the knowledge bases to be aligned.

As a by-product of label propagation on the graph, each verb pattern (lemmatized verb phrase that matches our part-of-speech-based regular expression: V | VP, see section 1.4) and NP-pair node in the graph (i.e., the *V:* and *SO:* nodes in Figure 3.1) will be assigned scores for each relation label. Exploiting these scores, we can estimate the probability that a verb pattern v represents a relation r as $P(v|r) \approx \frac{\hat{Y}(v,r)}{\sum_{v'} \hat{Y}(v',r)}$, where $\hat{Y}(v,r)$ is the score of label r assigned to verb pattern node v. Since a verb pattern may represent different relations depending on

Knowledge Base	Relation	Learned Verbs
Freebase	/sports/sports_team/ arena_stadium	played at, played in, defeated at, will host at, beaten at
	/medicine/medical_ treatment/side_effects	may promote, can cause, may produce, is worsened, exacerbate
NELL	drugPossiblyTreats PhysiologicalCondition	treat, relieve, reduce, help with, can help alleviate
	politicianHoldsOffice	serves as, run for, became, was elected
Yago2	actedIn	played in, starred in, starred, played, portrayed in
	isMarriedTo	married, met, date, wed, divorce

Table 3.1: Examples of relation-verb pattern pairs automatically learned by PIDGIN. Although we stem verbs in experiments, for better readability, we present the original non-stemmed forms of the same verbs above.

the category types of the NP-pair with which it co-occurs e.g., the verb pattern *enter* has a different meaning when it appears with an NP-pair ⟨*Paul*, *room*⟩ from when it appears with an NP-pair ⟨*John*, *American Idol*⟩; when estimating $P(v|r)$ we also take into account the scores of r on the NP-pair nodes $\langle NP_1, NP_2 \rangle$ with which verb pattern v co-occurs. We now measure, $P(v|r) \approx \frac{Y(v,r)}{\sum_{v'} Y(v',r)}$ where $Y(v,r) = \sum_{T_v} \hat{Y}(T_v, r)$, where T_v is an SVO triple $\langle np_1$, v, $np_2 \rangle$, and where $\hat{Y}(T_v, r) = \hat{Y}(\langle np_1, np_2 \rangle, r) * \hat{Y}(v, r)$. We multiply this estimate with the tf-idf score of the verb pattern, which is proportional to the number of times a verb pattern appears for a relation but is offset by the total number of times the verb pattern appears with all the relations. This helps to reduce the effect of common verbs such as *is*, *become* that represent many relations.

Using this scoring, for each relation, we can return a ranked list of verbs that represent the relation. Some of the verbs returned are shown in Table 3.1. As we can see in Table 3.1, the system is able to distinguish verbs representing the relation ***/medicine/medical_ treatment/side_effects***: “exacerbate”, “can cause” from the verbs representing the antonym relation ***drugPossiblyTreatsPhysiologicalCondition***: “relieve”, “can help alleviate” even when the two relations have the same domain (*drug*) and range (*physiological condition*). The system is also able to recognize the directionality of the relation. For example, for the relation ***_acquired***, which represents the inverse of the relation ***acquired*** (as in company X acquiring company Y); the system is able to return the correct verbs: “(passive) buy” and “(passive) purchase”, which are the inverse forms of the verbs “buy” and “purchase” (as in “be bought by” and “be purchased by”). Of practical importance is the fact that PIDGIN can learn verbs representing relations in knowledge bases whose instances are created manually or extracted via carefully constructed regular-expression matching (e.g., Freebase and YAGO). We can, for example, use these verbs to then automate an extraction process for these knowledge bases.

3.3 Overview of Method

Motivated by these encouraging initial mapping results that are the by-product of ontology alignment, we formalize the problem further and use the idea of web text as an interlingua to map verbs to knowledge base relations. Given a knowledge base such as NELL Carlson et al. [2010] which consists of:

(1) an ontology that defines a set of categories (e.g., *SportsTeam, City*),
(2) another part of the ontology that defines relations with these categories as their domain and range types (e.g., ***teamPlaysInCity****(SportsTeam, City)*),
(3) constraint specifications (e.g., mutual exclusion, subset) among knowledge base categories and relations,
(4) knowledge base entities which instantiate these categories (e.g., *Steelers* ∈ *SportsTeam*),
(5) knowledge base entity pairs which instantiate these relations (e.g., (*Steelers*, *Pittsburgh*) ∈ ***teamPlaysInCity***),

we map verbs to knowledge base (KB) relations using a very large ClueWeb corpus as a kind of interlingua. Our approach first grounds each KB relation instance (e.g., ***teamPlaysInCity****(Steelers, Pittsburgh)*) in mentions of its argument pair in this text, then represents the relation in terms of the verbs that connect these paired mentions (see Fig. 3.2).

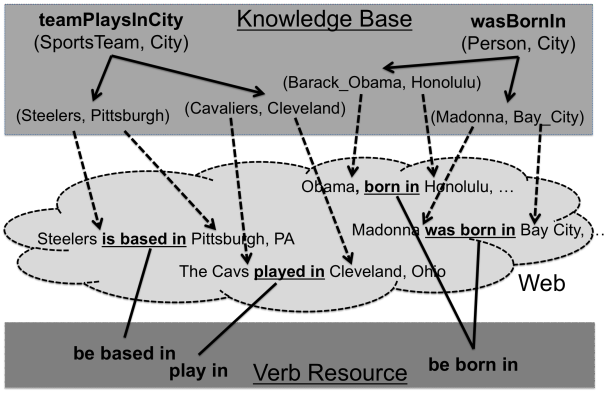


Figure 3.2: Mapping verbs to relations in KB through web text as an interlingua. Each relation instance is grounded by its mentions in the web text. The verbs that co-occur with mentions of the relation's instances are mapped to that relation.

For high coverage mappings, we train on both labeled and unlabeled data in our web text using Expectation Maximization (EM). We introduce type checking during the EM process to ensure only typed verbs whose subject and object types match the relation's domain and range types are mapped to the relation. We also incorporate constraints defined in the KB ontology to find typed verbs to relations mappings that are consistent with these constraints. Since the method is both KB- and language-independent, we can use the same method for constructing an English verb resource to automatically construct a Portuguese verb resource.

3.4 Terminology

We define the NELL knowledge base to be a 6-tuple $(C, I_C, R, I_R,$ *Subset*, *Mutex*$)$. C is the set of categories e.g., *SportsTeam* i.e., $c_j \in C = \{c_1, ..., c_{|C|}\}$. I_C is the set of category instances which are entity-category pairs e.g., (*Cleveland*, *City*) i.e., $I_C = \{(e_m, c_j) \mid e_m \in c_j,\ c_j \in C\}$.

R is the set of relations e.g., ***teamPlaysInCity*** i.e., $r_i \in R = \{r_1, ..., r_{|R|}\}$. We also define f_{type} to be a function that when applied to a relation r_i returns the type signature of the relation $f_{type}(r_i) = (c_j, c_k)$ for some $c_j, c_k \in C$ e.g., f_{type}(***teamPlaysInCity***) = (*SportsTeam*, *City*).

I_R is the set of relation instances which are entity-relation-entity triples e.g., *(Cavaliers,* ***teamPlaysInCity****, Cleveland)* i.e., $I_R = \{(e_m, r_i, e_n) \mid (e_m, e_n) \in r_i,\ r_i \in R,\ e_m \in c_j,\ e_n \in c_k,\ f_{type}(r_i) = (c_j, c_k)\}$; $I_R = I_{r_1} \cup I_{r_2} \cup ... I_{r_{|R|}}$; where (e_m, e_n) is an entity pair e.g., (*Cavaliers*, *Cleveland*).

Subset is the set of all subset constraints among relations in R i.e., *Subset* $= \{(i, k) : I_{r_i} \subseteq I_{r_k}\}$. For example $\{$*(person,* ***ceoOf****, company)*$\} \subseteq \{$*(person,* ***worksFor****, company)*$\}$.

Mutex is the set of all mutual exclusion constraints among relations in R i.e., *Mutex* $= \{(i, k) : I_{r_i} \cap I_{r_k} = \phi\}$. For example $\{$*(drug,* ***hasSideEffect****, physiologicalCondition)*$\} \cap \{$*(drug,* ***possiblyTreats****, physiologicalCondition)*$\} = \phi$.

Each KB entity e_m can be referred to by one or more noun phrases (NPs). For example, the entity *Cavaliers* can be referred to in text using either the NP *"Cleveland Cavaliers"* or the NP *"The Cavs"*[1]. We define $N_{en}(e_m)$ to be the set of English NPs corresponding to entity e_m.

We define SVO to be the English Subject-Verb-Object (SVO triples) interlingua[2] consisting of tuples of the form (np_s, v_p, np_o, w), where np_s and np_o are noun phrases (NP) corresponding to subject and object, respectively, v_p is a verb pattern that connects them, and w is the count of the tuple.

3.5 Data Construction

We construct a dataset D for mapping English verbs to relations in NELL KB. First, we convert each tuple in SVO to its equivalent entity pair tuple(s) in SVO' $= \{(e_m, v_p, e_n, w) \mid np_s \in N_{en}(e_m),\ np_o \in N_{en}(e_n),\ (np_s,\ v_p,\ np_o,\ w) \in SVO\}$. For example, we convert the tuple ("Pitt", "marry", "Jolie", 9302) in SVO to the entity pair tuple (*Brad Pitt*, "marry", *Angelina Jolie*, 9302) in SVO', where *Brad Pitt* and *Angelina Jolie* are entities in the NELL knowledge base.

Then, we construct D from SVO' as a collection of labeled and unlabeled instances.

[1] defined by the ***canReferTo*** relation in NELL KB

[2] We use 600 million SVO triples collected from the entire ClueWeb Callan et al. [2009a] of about 230 billion tokens with some filtering described in Section 3.8.1.

The set of labeled instances is $D^\ell = \{(\mathbf{y}_{(e_m,e_n)}, \mathbf{v}_{(e_m,e_n)})\}$ where $\mathbf{y}_{(e_m,e_n)} \in \{0,1\}^{|R|}$ is a bit vector of label assignment, each bit representing whether the instance belongs to a particular relation i.e., $y^i_{(e_m,e_n)} = 1 \iff (e_m, e_n) \in r_i$ and 0 otherwise. $\mathbf{v}_{(e_m,e_n)} \in \mathbb{R}^{|V|}$ is a $|V|$-dimensional vector of verb pattern counts that connect e_m and e_n in SVO' (V is the set of all verb patterns) i.e., $v^p_{(e_m,e_n)}$ is the number of times the verb pattern v_p connects e_m and e_n in SVO'.

The collection of unlabeled instances is constructed from entity pairs in SVO' whose label assignment $\mathbf{y}$ is unknown (its bits are all zero) i.e.,
$D^u = \{(\mathbf{y}_{(e_m,e_n)}, \mathbf{v}_{(e_m,e_n)}) \mid (e_m, *, e_n, *) \in SVO', (e_m, *, e_n) \notin I_R\}$.

An instance in our dataset $d_{(e_m,e_n)} \in D$ is therefore either a labeled or unlabeled tuple i.e., $d_{(e_m,e_n)} = (\mathbf{y}_{(e_m,e_n)}, \mathbf{v}_{(e_m,e_n)})$.

We let $f_{type}(d_{(e_m,e_n)})$ return the argument type of the instance i.e., $f_{type}(d_{(e_m,e_n)}) = (c_j, c_k)$ where (e_m, c_j) and $(e_n, c_k) \in I_C$.

We let $f_{verb}(d_{(e_m,e_n)})$ return the set of all verb patterns that co-occur with the instance in SVO' i.e., $f_{verb}(d_{(e_m,e_n)}) = \{v_p \mid (e_m, v_p, e_n, *) \in SVO'\}$.

When applied to a relation r_i, we let $f_{verb}(r_i)$ return the set of all verb patterns that co-occur with instances in D whose types match that of the relation i.e., $f_{verb}(r_i) = \{v_p \mid \exists\ d_{(e_m,e_n)} \in D,\ v_p \in f_{verb}(d_{(e_m,e_n)}),\ f_{type}(d_{(e_m,e_n)}) = f_{type}(r_i)\}$.

3.6 Model

We train a Naive Bayes classifier on our dataset. Given as input a collection D^ℓ of labeled instances and D^u of unlabeled instances, it outputs a classifier, $\hat{\theta}$, that takes an unlabeled instance and predicts its label assignment i.e., for each unlabeled instance $d_{(e_m,e_n)} \in D^u$ the classifier predicts the label assignment $\mathbf{y}_{(e_m,e_n)}$ using $\mathbf{v}_{(e_m,e_n)}$ as features:

$$\begin{aligned} &P(y^i_{(e_m,e_n)} = 1 \mid d_{(e_m,e_n)};\ \hat{\theta}) \\ &\qquad = \frac{P(r_i|\hat{\theta}) P(d_{(e_m,e_n)}|\ r_i; \hat{\theta})}{P(d_{(e_m,e_n)}|\hat{\theta})} \\ &\qquad = \frac{P(r_i|\hat{\theta}) \prod_{p=1}^{|V|} P(v_p|r_i;\hat{\theta})^{v^p_{(e_m,e_n)}}}{\sum_{k=1}^{|R|} P(r_k|\hat{\theta}) \prod_{p=1}^{|V|} P(v_p|r_k;\hat{\theta})^{v^p_{(e_m,e_n)}}} \end{aligned} \tag{3.1}$$

If the task is to classify the unlabeled instance into a single relation, only the bit of the relation with the highest posterior probability is set i.e, $y^k_{(e_m,e_n)} = 1$ where $k = \arg\max_i P(y^i_{(e_m,e_n)} = 1 \mid d_{(e_m,e_n)}; \hat{\theta})$.

3.6.1 Parameter Estimation

To estimate model parameters (the relation prior probabilities $\hat{\theta}_{r_i} \equiv P(r_i|\hat{\theta})$ and probabilities of a verb pattern given a relation $\hat{\theta}_{v_p|r_i} \equiv P(v_p|r_i;\hat{\theta})$) from both labeled and unlabeled data, we use a *hard* Expectation Maximization (EM) algorithm Nigam et al. [2006]. The estimates are computed by calculating a maximum a posteriori estimate of θ, i.e. $\hat{\theta} = \arg\max_{\theta} \; \mathcal{L}(\theta|D) = \arg\max_{\theta} \; log(P(D \mid \theta)P(\theta))$.

The first term, $P(D \mid \theta)$ is calculated by the product of all the instance likelihoods:

$$\begin{aligned} & P\left(D \mid \theta\right) \\ & \quad = \prod_{d_{(e_m,e_n)} \in D^u} \sum_{i=1}^{|R|} P(r_i|\theta)P(d_{(e_m,e_n)}|r_i;\theta) \\ & \quad \times \prod_{d_{(e_m,e_n)} \in D^{\ell}} \sum_{\{i|y^i_{(e_m,e_n)}=1\}} P(r_i|\theta)P(d_{(e_m,e_n)}|r_i;\theta) \end{aligned} \tag{3.2}$$

The second term, $P(\theta)$, the prior distribution over parameters is represented by Dirichlet priors: $P(\theta) \propto \prod_{i=1}^{|R|}((\theta_{r_i})^{\alpha_1 - 1} \prod_{p=1}^{|V|}(\theta_{v_p|r_i})^{\alpha_2 - 1})$ where α_1 and α_2 are parameters that effect the strength of the priors. We set $\alpha_1 = 2$ and $\alpha_2 = 1 + \sigma(P^e(v_p|r_i))$, where $P^e(v_p|r_i)$ is the initial bias of the mapping from the verb pattern v_p to the relation r_i. Thus, we define $P(\theta)$ as:

$$P(\theta) = \prod_{i=1}^{|R|}(P(r_i|\theta) \prod_{p=1}^{|V|}(P(v_p|r_i;\theta)^{\sigma(P^e(v_p|r_i))}) \tag{3.3}$$

We can see from this that $\sigma(P^e(v_p|r_i))$ is a conjugate prior on $P(v_p|r_i;\theta)$ with σ as the confidence parameter. This conjugate prior allows incorporation of any existing knowledge (Section 3.6.2) we may have about the verbs to relations mappings.

From Equation 3.2, we see that $log\ P(D|\theta)$ contains a log of sums, which makes a maximization by partial derivatives computationally intractable. Using EM, we instead maximize the *expected* log likelihood of the data with respect to the posterior distribution of the y labels given by: $\arg\max_{\theta} E_{(\mathbf{y}|D;\theta)}[log\ P(D|\theta)]$.

In the E-step, we use the current estimates of the parameters $\hat{\theta}^t$ to compute $\hat{\mathbf{y}}^t = E[\mathbf{y}|D;\hat{\theta}^t]$ the expected label assignments according to the current model. In practice, it corresponds to calculating the posterior distribution over the y labels for unlabeled instances $P(y^i_{(e_m,e_n)} = 1 \mid d_{(e_m,e_n)};\hat{\theta}^t)$ (Equation 3.1) and using the estimates to compute its expected label assignment $\hat{\mathbf{y}}^t_{(e_m,e_n)}$.

In the M-step, we calculate a new maximum a posteriori estimate for $\hat{\theta}^{(t+1)}$ which maximizes the expected log likelihood of the complete data, $\mathcal{L}_c(\theta|D;\hat{\mathbf{y}}^t) = log(P(\theta^t)) + \ \hat{\mathbf{y}}^t\ [log\ P(D|\theta^t)]$:

$$\begin{aligned} & \mathcal{L}_c(\theta|D;\hat{\mathbf{y}}^{\mathbf{t}}) = log(P(\theta^t)) \\ & \quad + \sum_{d_{(e_m,e_n)} \in D} \sum_{i=1}^{|R|} y^{ti}_{(e_m,e_n)}\ log\ P(r_i|\theta)P(d_{(e_m,e_n)}|r_i;\theta) \end{aligned} \tag{3.4}$$

$\mathcal{L}_c(\theta|D;\mathbf{y})$ bounds $\mathcal{L}(\theta|D)$ from below (by application of Jensen's inequality $E[log(X)] \leq log(EX)$). The EM algorithm produces parameter estimates $\hat{\theta}$ that correspond to a local maximum of $\mathcal{L}_c(\theta|D;\mathbf{y})$. The relation prior probabilities are thus estimated using current label assignments as:

$$P(r_i|\hat{\theta}^{(t+1)}) = \frac{1 + \sum\limits_{d_{(e_m,e_n)} \in D} y^{ti}_{(e_m,e_n)}}{|R| + |D|} \tag{3.5}$$

The verbs to relations mapping probabilities are estimated in the same manner:

$$P(v_p \mid r_i;\ \hat{\theta}^{(t+1)}) = \frac{\sigma_i^{(t+1)}\ (P^e(v_p \mid r_i))\ + \sum\limits_{d_{(e_m,e_n)} \in D} v^p_{(e_m,e_n)}\ y^{ti}_{(e_m,e_n)}}{\sigma_i^{(t+1)}\ + \sum\limits_{s=1}^{|V|} \sum\limits_{d_{(e_m,e_n)} \in D} v^s_{(e_m,e_n)}\ y^{ti}_{(e_m,e_n)}} \tag{3.6}$$

We start with $\sigma = |V|$ and gradually reduce the impact of prior by decaying σ with a decay parameter of 0.8 at each iteration in the manner of Lu and Zhai [2008]). This will allow the EM to gradually pick up more verbs from the data to map to relations.

EM iteratively computes parameters θ^1, ..., θ^t using the above E-step and M-step update rule at each iteration t, halting when there is no further improvement in the value of $\mathcal{L}_c(\theta|D;\mathbf{y})$.

3.6.2 Prior Knowledge

In our prior $P(\theta)$, we incorporate knowledge about verbs to relations mappings from the text patterns learned by NELL to extract relations. This is our way of *aligning* our verbs to relations mappings with NELL's current extractions. Coupled Pattern Learner (CPL) Carlson et al. [2010] is a component in NELL that learns these contextual patterns for extracting instances of relations and categories. Examples of CPL's extraction patterns for the relation ***hasSpouse*** are "a divorce from", "along with wife", 'is married to", etc. We consider only CPL's extraction patterns that when lemmatized contain verb phrases that match our part-of-speech-based regular expression: V | VP (see section 1.4). We extract the verbs and the prepositions that occurs in these lemmatized patterns. For example, among this set of examples of CPL patterns, we extract the pattern "(passive) marry to" (meaning "to be married to").

Given a set E_{r_i} of CPL's extraction patterns for a relation r_i, and E_{r_i,v_p} as the set of extraction patterns in E_{r_i} that contain the verb pattern v_p, we compute $P^e(v_p \mid r_i) = \frac{\mid E_{r_i,v_p} \mid}{\mid E_{r_i} \mid}$ and use them as priors in our classifier (Equation 3.3).[3]

3.6.3 Type Checking

Although some verbs are ambiguous (e.g., the verb pattern "play" may express several relations: ***musicianPlaysMusicalInstrument***, ***athletePlaysSport***, ***actorPlaysMovie***, etc), knowing

[3] We manually add a few verb patterns for relations whose E_r is an empty set when possible, to set the EM process on these relations with good initial guesses of the parameters. On average, each relation has about 6 verb patterns in total as priors.

the types of the verbs' subjects and objects can help disambiguate the verbs (e.g., the typed verb "play" that takes a *musicalInstrument* type as an object is more likely to express the ***musicianPlaysMusicalInstrument*** relation). Therefore, we incorporate *type checking* in our EM process to ensure that it maps typed verbs to relations whose domain and range types match the verbs' subject and object types:

- In the E-Step, for each unlabeled instance (entity pair), we only consider existing NELL relations whose domain and range types match the entity pair's types *and* that share some verbs with the entity pair as potentially being the labels of the instance. In other words, in the E-step we only compute $P(y^i_{(e_m,e_n)} = 1 \mid d_{(e_m,e_n)})$ if $f_{type}(r_i) = f_{type}(d_{(e_m,e_n)})$ *and* $\big(f_{verb}(r_i) \cup \{v_p | E_{r_i,v_p} \neq \emptyset\}\big) \cap f_{verb}(d_{(e_m,e_n)}) \neq \emptyset$.
- In the M-step, for each typed verb, we only consider existing NELL relations whose domain and range types match the types of at least one of the entity pairs that co-occur with the verb in the SVO' as potentially being mapped to the verb. In other words, in the M-step we only compute $P(v_p \mid r_i)$ if $v_p \in f_{verb}(r_i)$ or $E_{r_i,v_p} \neq \emptyset$.

3.6.4 Incorporating Constraints

In the E-step, for each unlabeled instance, given the probabilities over relation labels $P(y^i_{(e_m,e_n)} = 1 \mid d_{(e_m,e_n)}; \hat{\theta}^t)$, and *Subset* and *Mutex* constraints[4], similar to Dalvi et al. [2015], we use a Mixed-Integer Program (MIP) to produce its bit vector of label assignment as output: $\hat{\mathbf{y}}^t_{(e_m,e_n)}$.

The constraints among relations are incorporated as constraints on bits in this bit vector. For example, if for an unlabeled instance (*Jeff Bezos*, *Amazon*), a bit corresponding to the relation ***ceoOf*** is set then the bit corresponding to the relation ***worksFor*** should also be set due to the subset constraint: ***ceoOf*** $\subseteq$ ***worksFor***. For the same instance, the bit corresponding to ***competesWith*** should not be set due to the mutual exclusion constraint ***ceoOf*** $\cap$ ***competesWith*** = ϕ. The MIP formulation for each unlabeled instance thus tries to maximize the sum of probabilities of selected relation labels after penalizing for violation of constraints (Equation 3.7), where ζ_{ik} are slack variables for *Subset* constraints and δ_{ik} are slack variables for *Mutex* constraints:

$$\begin{aligned}
\underset{\mathbf{y}_{(e_m,e_n)},\zeta_{ik},\delta_{ik}}{\text{maximize}} & \Bigg(\sum_{i=1}^{|R|} y^i_{(e_m,e_n)} \times \; P(y^i_{(e_m,e_n)} = 1 | d_{(e_m,e_n)}; \hat{\theta}^t) \\
& - \sum_{(i,k)\in Subset} \zeta_{ik} \; - \sum_{(i,k)\in Mutex} \delta_{ik} \Bigg) \\
& \text{subject to,} \\
& y^i_{(e_m,e_n)} \leq \; y^k_{(e_m,e_n)} + \zeta_{ik}, \forall (i,k) \in Subset \\
& y^i_{(e_m,e_n)} + y^k_{(e_m,e_n)} \leq 1 + \delta_{ik}, \forall (i,k) \in Mutex \\
& \zeta_{ik}, \delta_{ik} \geq 0, y^i_{(e_m,e_n)} \in \{0,1\}, \forall i,k
\end{aligned} \tag{3.7}$$

[4] The *Subset* and *Mutex* constraints are obtained as part of the NELL knowledge base ontology, which is publicly available at the NELL Read The Web project website: http://rtw.ml.cmu.edu/resources/.

Our EM algorithm that includes type checking and constraints is summarized in Algorithm 1.

Algorithm 1 The EM Algorithm for Mapping Typed Verbs To Relations

```
Input: $D = D^\ell \cup D^u$ and an initial naive Bayes classifier $\theta^1$ from labeled documents $D^\ell$ only
    (using Equations 3.5 and 3.6)
Output: $\theta^T$ that includes typed verbs to relations mappings given by $P(v_p|r_i;\theta^T)$
for $t$ = 1 ... $T$ do
    E-Step:
    for $d_{(e_m,e_n)} \in D^u$ do
        Compute $P(y^i_{(e_m,e_n)} = 1|d_{(e_m,e_n)};\theta^t)\ \forall r_i \in R$ that satisfy type checking (Equation 3.1)
        Find a consistent label assignment $\mathbf{y}^t_{(e_m,e_n)}$ by solving MIP (Equation 3.7)
    end for
    M-step: Recompute model parameters $\theta^{t+1}$ based on current label assignments (Equation
        3.5 and 3.6) respecting type checking
    if convergence ($\mathcal{L}_c(\theta^{t+1})$, $\mathcal{L}_c(\theta^t)$) then
        break
    end if
end for
return $\theta^T$
```

3.7 Portuguese Verb Mappings

3.7.1 Mapping Verbs To Relations in Portuguese NELL

Given a text corpus in any language and any knowledge base, the algorithm can produce mappings from that language's typed verbs to the knowledge base relations. To demonstrate this, we map typed Portuguese verbs to relations in Portuguese NELL [Duarte and Hruschka, 2014], which is an automatically and independently constructed KB separate from English NELL.

We use Portuguese NELL and Portuguese text corpus SVO_{pt}[5] and construct a dataset D_{pt}. Given D_{pt}, we follow the same approach as before to find mappings from typed Portuguese verbs to relations. Since Portuguese NELL is newly constructed, it contains fewer facts (category and relation instances) than English NELL, and hence its dataset D^ℓ_{pt} has fewer labeled instances (see Table 3.2).

3.7.2 Adding Labeled Instances to Portuguese NELL and SVO triples by aligning English NELL Entities to Portuguese Noun Phrases

Since English NELL has extracted more relation instances and semantically typed more entities than Portuguese NELL, we use relation instances and entities in English NELL to add

[5] We obtain the Portuguese SVO triples from the NELL-Portuguese team at the Federal University of Sao Carlos.

	English NELL	**Portuguese NELL**	**Portuguese NELL^{+en}**
Number of relations in KB $\|R\|$	317	302	302
Number of relation instances in KB $\|I_R\|$	135,267	5,675	12,444
Number of labeled entity pairs in SVO $\|D^\ell\|$	85,192	2,595	5,412
Number of unlabeled entity pairs in SVO $\|D^u\|$	240,490	595,274	1,186,329

Table 3.2: Statistics of KB and dataset constructed

more knowledge to Portuguese NELL. Specifically, since each category and each relation in the Portuguese NELL ontology has a one-to-one mapping in the English NELL ontology, we can add relation instances to Portuguese NELL from the corresponding relations in English NELL. Adding more relation instances to Portuguese NELL can result in more labeled instances in the dataset D_{pt}, a more productive EM, and better typed verbs-to-relations mappings.

English NELL however, has only English noun phrases (NPs) to refer to entities in its relation instances. To add more labeled instances in D_{pt} using English relation instances, we need to find instantiations of these English relation instances in Portuguese SVO_{pt}, which translates to finding Portuguese NPs that refer to entities in English NELL. For example, the Portuguese NP: "Artria torcica interna" for the English NELL entity: *internal mammary artery*.

To automatically translate entities in English NELL to Portuguese NPs, we use DBPedia Auer et al. [2007] which has structured information about Wikipedia pages in many languages. The idea is to map each English NELL entity e_m to its corresponding English DBPedia page and therefore its Portuguese DBPedia page[6]. We use the structured information of the Portuguese page in DBPedia: its title and label as the set of Portuguese NPs corresponding to the English entity, $N_{pt}(e_m)$.

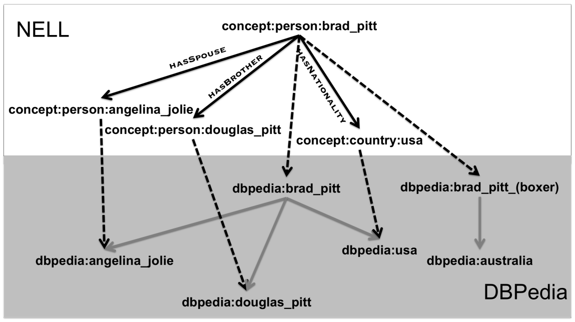


Figure 3.3: The mapping of the NELL entity *Brad Pitt* to DBPedia.

More specifically, for each English NELL entity e_m with English NPs that can refer to it, $N_{en}(e_m)$, we find *candidate* English DBPedia pages that can refer to the entity. We do this by computing Jaccard similarities Jaccard [1912], Chapman [2009] of the entity's NPs with titles and labels of English DBPedia pages. We select pages with Jaccard similarities of more than 0.6 as candidates e.g., for the English NELL entity *Brad Pitt* we find candidate English pages:

[6] Almost every DBPedia English page has a corresponding Portuguese page

http://dbpedia.org/page/Brad_Pitt (*Brad Pitt*, the US actor) and http://dbpedia.org/page/Brad_Pitt_(boxer) (*Brad Pit*, the Australian boxer).

Then, we construct a graph containing nodes that are: (1) the NELL entity that we want to map to DBPedia, (2) its candidate DBPedia pages, (3) other entities that have relations to the entity in NELL KB, and (4) the candidate DBPedia pages of these other entities (see Fig. 3.3 for the NELL entity *Brad Pitt*).

We add as edges to this graph: (1) the can-refer-to edges between entities in NELL and their candidate pages in DBPedia (dashed edges in Fig. 3.3), (2) the relation edges between the entities in NELL KB (black edges), and (3) the hyperlink edges between the pages in DBPedia (gray edges). In this graph, we want to use the knowledge that NELL has already learned about the entity to narrow its candidates down to the page that the entity refers to. The idea is that relatedness among the entities in NELL implies relatedness among the DBPedia pages that refer to the entities. We use Personalized Page Rank Page et al. [1999] to rank candidate DBPedia pages in this graph and pick the top ranked page as the page that can refer to the NELL entity.[7]

For example, to find the DBPedia page that can refer to our NELL entity *Brad Pitt*, we use NELL's knowledge about this entity to rank its candidate pages. As seen in Fig. 3.3, DBPedia page of *Brad Pitt*, the US actor (*dbpedia:brad_pitt*) is highly connected to other pages (*dbpedia:angelina_jolie*, *dbpedia:douglas_pitt*, *dbpedia:usa*) that are in turn connected to the NELL entity *Brad Pitt*. *dbpedia:brad_pitt* is thus ranked highest and picked as the page that can refer to the NELL entity *Brad Pitt*.

Once we have an English DBPedia page that can refer to the NELL entity e_m, we can obtain the corresponding Portuguese page from DBPedia. The title and label of the Portuguese page become the set of Portuguese NPs that can refer to the NELL entity i.e., $N_{pt}(e_m)$ (see Table 3.3 for examples). Using $N_{pt}(e_m)$ we find instantiations of English relation instances in SVO_{pt} to add as labeled instances in D_{pt}. Portuguese NELL that is enriched with English NELL (i.e., Portuguese NELL^{+en}) has more than double the amount of relation instances, labeled and unlabeled instances (Table 3.2) than Portuguese NELL. In the experiments, we observe that this translates to better typed verbs-to-relations mappings.

Mapping NELL to DBPedia is also useful because it can align existing knowledge and add new knowledge to NELL. For example, by mapping to DBPedia, we can resolve abbreviations (e.g., the NELL entity: *COO* as "Chief Operations Officer" in English or "Diretor de Operaç$\tilde{o}$es" in Portuguese), or resolve a person entity (e.g., the NELL entity: *Utamaro* as "Kitagawa Utamaro", the virtual artist).

[7] There are previous works for mapping entities in NELL to pages in DBPedia such as the work of Dalvi et al. [2015], which produces mappings by taking into consideration the hierarchical and mutual exclusion constraints between the categories in NELL. There are also other algorithms that can be used to label pages in DBPedia with entities in NELL such as the Modified Adsorption (MAD) algorithm that can propagate labels on graphs [Talukdar and Crammer, 2009]. In the future, we can explore how these works can complement ours.

English NELL entity	**Portuguese NPs**
Amazonian Brown Brocket	“Veado-Roxo”, “Fuboca”
COO	“Diretor de Operaç$\tilde{o}$es”
Utamaro	“Kitagawa Utamaro”
Notopteridae	“Peixe-faca”
1967 Arab Israeli War	“Guerra dos Seis Dias”, “Guerra de 1967”
Food Products	“Produtos Alimenticios”, “Alimento”, “Comida”, ...

Table 3.3: Example Portuguese NPs learned for NELL entities

3.8 Experiments

3.8.1 Design Choices

For better coverage of verbs, we lemmatize verbs in the English SVO (using the Stanford CoreNLP Manning et al. [2014]). We lemmatize verbs in Portuguese SVO_{pt} (using LemPORT Rodrigues et al. [2014]) and expand contracted prepositions.

For better precision and to make our method scale to a large text corpus, we focus on mapping typed verbs that are informative for a relation based on how often the verbs co-occur with entity pairs whose types match the relation's domain and range types. We use *tf-idf* scores counts to adjust for the fact that some verbs appear more frequently in general (see section 1.4).

For each argument type in the English SVO we consider only the top 50 typed verbs (in terms of *tf-idf* scores) to map. For each of these typed verbs, we also use only the top 50 entity pairs that co-occur with the verb in the SVO (in terms of co-occurrence counts) to construct our dataset D.

For typed Portuguese verbs-to-relations mappings, since SVO_{pt} is much smaller than the English SVO (i.e., it contains only about 22 million entity pair-verb triples compared to the 600 million triples in the English SVO), we use all the Portuguese entity pairs and verbs for the mapping. To adjust for the fact that some verbs appear more frequently in general, we use *tf-idf* scores instead of co-occurrence counts for the values of $\mathbf{v}_{(e_m,e_n)}$ in the M-step (Equation 3.6).

3.8.2 Evaluation

We set aside 10% of D^ℓ (labeled entity pairs in NELL) for testing. Given a test instance $t_{(e_m,e_n)}$ and the trained model, we can predict the label assignment $\mathbf{y}_{(e_m,e_n)}$ using Eq. 3.1. This simulates the task of relation extraction where we predict relation(s) that exist between the entity pair in $t_{(e_m,e_n)}$.

We compare predicted labels of these test instances to their labels in NELL and measure precision, recall and F1 values of the prediction. We evaluate relations in NELL that have more than one labeled instance in D^ℓ (constructed using the method described in section 3.5). For experiments on English NELL, we evaluate 77 relations, with an average of 23 (and a median of 11) *training* instances per relation. For experiments on Portuguese NELL^{+en}, which is Portuguese

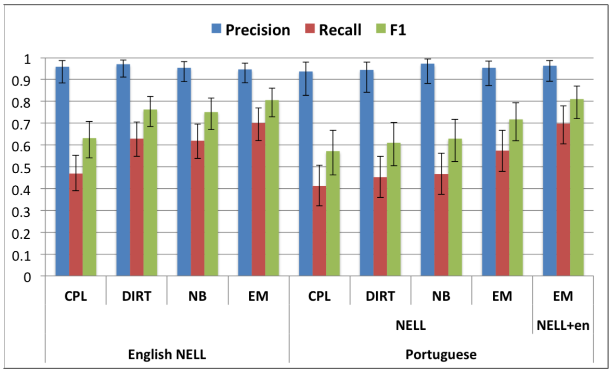


Figure 3.4: Performance on leaf relations without MIP and with error bars indicating 0.9-confidence Wilson score interval [Brown et al., 2001].

NELL enriched with relation instances from English NELL, we evaluate 85 relations, with an average of 31 (and a median of 10) *training* instances per relation. We compare the prediction produced by our approach: **EM** with that of other systems: **CPL**, **DIRT**, and **NB**[8].

In **CPL**, we obtain verbs-to-relations mapping weights from NELL's CPL patterns and hand-labeled verb patterns (see Section 3.6.2). In **DIRT**, we obtain verbs-to-relations mapping weights in an unsupervised manner Lin and Pantel [2001b] based on their mutual information over labeled training instances. In Naive Bayes (**NB**) we learn the verbs-to-relations mapping weights from labeled training instances. In contrast to the other systems, **EM** allows learning from both labeled and unlabeled instances.

To make other systems comparable to our proposed method, for **NB** and **DIRT**, we add **CPL** weights as priors to their verbs-to-relations mapping weights. For all these other systems, we also incorporate type-checking during prediction in that unlabeled instances are only labeled with relations that have the same domain and range types as the instances' argument types.

We show the micro-averaged performance of the systems on *leaf* relations of English NELL and Portuguese NELL (Fig. 3.4), where we do not incorporate constraints and classify each test instance into a single relation. We observe in both the English and Portuguese NELL that the typed verbs-to-relations mappings obtained by **EM** result in predictions that have a statistically significant higher recall and a comparable precision to the predictions made using mappings produced by **CPL**, **DIRT**, and **NB**.

In Figure 3.4, we also observe a gain in performance when we run **EM** on Portuguese NELL^{+en} which is Portuguese NELL enriched with relation instances from English NELL obtained using our DBPedia linking in section 3.7. More labeled instances result in a higher precision and statistically significant higher recall and F1 score. This shows the usefulness of aligning and merging knowledge from many different KBs to improve typed verbs-to-relations mappings and relation extraction in general.

We show the micro-averaged performance of the systems on *all* relations of English NELL and Portuguese NELL (Fig. 3.5). Here, we incorporate hierarchical and mutually exclusive

[8] We do not compare with PATTY[Nakashole et al., 2012] for the task of relation extraction in NELL as the publicly released resource for PATTY only provides mappings between typed verbs and relations in YAGO and DBPedia.

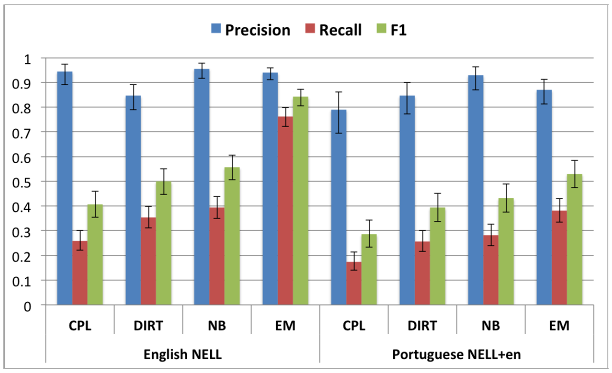


Figure 3.5: Performance on all relations with error bars indicating 0.9-confidence Wilson score interval.

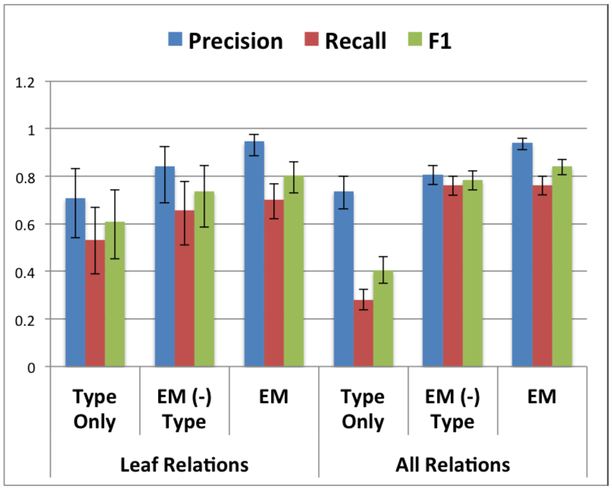


Figure 3.6: Performance on the English NELL relations with and without type-checking with error bars indicating 0.9-confidence Wilson score interval.

constraints between relations in our **EM**, allowing a test instance to be classified into more than one relation while respecting these constraints. Like before, we observe that the typed verbs-to-relations mappings obtained by **EM** result in predictions with statistically significant higher recall and F1 score compared to predictions produced using the mappings of other systems, which do not incorporate constraints between relations.

In the experiments, we also observe that **NB** performs comparably or better than **DIRT**. We hypothesize that it is because **NB** obtains its verbs-to-relations mappings in a supervised manner while **DIRT** obtains its mappings in an unsupervised manner.

We also conduct experiments to investigate how much influence type-checking has on prediction. We show performance over instances whose types alone are not enough to disambiguate their assignments (i.e., when more than one relation shares their argument types) to see the merits of verbs-to-relations mappings on prediction (Fig. 3.6). We observe that verbs learned by **EM** result in better predictions even when used without type-checking (**EM** (-) **Type**) than using

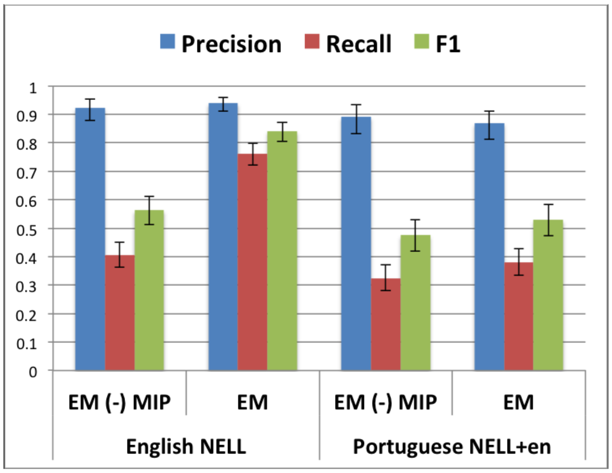


Figure 3.7: Performance on all relations with and without incorporating constraints using MIP with error bars indicating 0.9-confidence Wilson score interval.

type-checking alone (by picking the majority class among relations that have the correct type) (**Type Only**). Adding type checking in **EM** improves performance even further with statistically significant higher precision and F1 score. This shows how verbs learning is complementary to type-checking.

In Figure 3.7, we observe the effect of incorporating constraints between relations using MIP. We observe that the typed verbs learned using constraints (**EM**) result in predictions with a statistically significant higher recall and a comparable precision than the typed verbs learned without constraints (**EM - MIP**). Using *Subset* constraints, we improve the recall of predictions by predicting also the parent labels for the entity pairs. Using *Mutex* constraints, we maintain the precision of the predictions by predicting the labels for the entity pairs that respect the mutual exclusivity constraints between the relations.

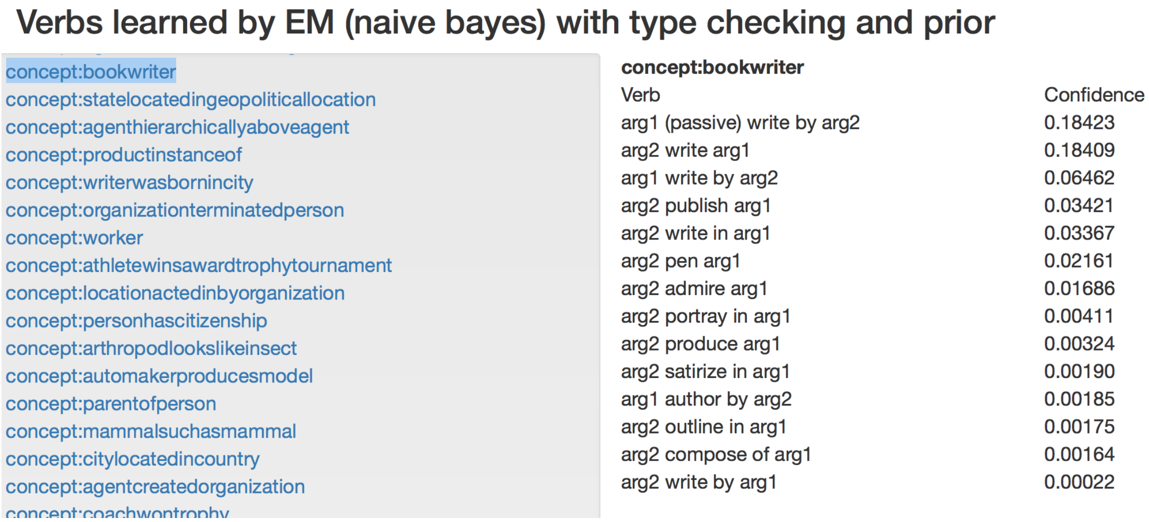
The results of our experiments highlight the merit of learning from a large, though unlabeled corpus to improve the coverage of verbs-to-relations mappings and thus the recall of predictions. We also observe the usefulness of incorporating constraints and for merging knowledge from multiple KBs to significantly improve recall. Another advantage of **EM** is that it produces relation labels for unlabeled data not yet in NELL KB. We show some of these new proposed relation instances as well as some of the typed verbs-to-relations mappings obtained by **EM** (Table 3.4).

EM learns on average 177 typed English verbs and 3310 typed Portuguese verbs per relation, and proposes on average 1695 new instances per relation for English NELL, and 6426 new instances per relation for Portuguese NELL[9]. It learns fewer English verbs than Portuguese due to the filtering of English data (Section 3.8.1) and a high degree of inflection in Portuguese verbs. The smaller size of the Portuguese knowledge base also means more of its proposed instances are new.

[9] The full list of the mappings can be browsed or downloaded from
http://www.cs.cmu.edu/%7Edwijaya/mapping.html
and as part of VerbKB in http://www.dwijaya.org/dvkb.html#DKVB

Relation	Verbs	Proposed New Instances
personHasJobPosition	a_1 become a_2, a_1 work as a_2, a_1 be crowned a_2	(*Hu Jintao*, *president*), (*Ashoka*, *king*), (*Tiger Woods*,*golfer*)
bookWriter	a_1 be written by a_2, a_2 write a_1	(*Dracula*, *Bram Stoker*), (*Divine Comedy*, *Dante*)
cityAlsoKnownAs	a_1 be known as a_2, a_2 be known as a_1, a_2 be renamed a_1,	(*Amman*, *Philadelphia*), (*Chennai*, *Madras*), (*Southport*, *Smithville*)
bacteriaEoAgenteCausador-DeCondicaoFisiologica	a_1 causar a_2, a_1 vírus em/de a_2, a_2 ser causar por a_1, a_1 transmissor de a_2	(*HIV*, *Diabetes Mellitus*), (*Borrelia*, *Lyme Arthritis*), (*P. Falciparum*, *Paludism*), (*Salmonella*, *Meningitis*)
liderDeOrganizacao	a_1 fundador a_2, a_1 ceo de/em a_2	(*Jimmy Wales*, *Wikipedia*), (*Chad Hurley*, *Youtube*)
pessoaAcusadaDoCrime	a_1 ser condenar aa_2, a_1 ser acusar de a_2, a_1 ser prender por a_2	(*Pedrinho Matador*, *Homicidios*), (*Omid Tahvili*, *Trafico de Drogaso*)

Table 3.4: Some relations' verbs and proposed new instances



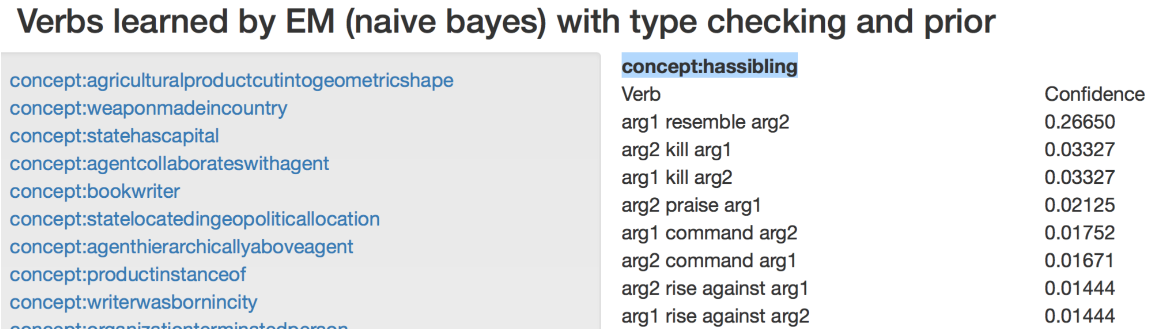
Verbs learned by EM (naive bayes) with type checking and prior

concept:bookwriter
concept:statelocatedingeopoliticallocation
concept:agenthierarchicallyaboveagent
concept:productinstanceof
concept:writerwasbornincity
concept:organizationterminatedperson
concept:worker
concept:athletewinsawardtrophytournament
concept:locationactedinbyorganization
concept:personhascitizenship
concept:arthropodlookslikeinsect
concept:automakerproducesmodel
concept:parentofperson
concept:mammalsuchasmammal
concept:citylocatedincountry
concept:agentcreatedorganization
concept:coachwontrophy

concept:bookwriter

Verb	Confidence
arg1 (passive) write by arg2	0.18423
arg2 write arg1	0.18409
arg1 write by arg2	0.06462
arg2 publish arg1	0.03421
arg2 write in arg1	0.03367
arg2 pen arg1	0.02161
arg2 admire arg1	0.01686
arg2 portray in arg1	0.00411
arg2 produce arg1	0.00324
arg2 satirize in arg1	0.00190
arg1 author by arg2	0.00185
arg2 outline in arg1	0.00175
arg2 compose of arg1	0.00164
arg2 write by arg1	0.00022

Figure 3.8: Our website that shows the mappings from verb patterns to NELL relations. The figure shows some of the verb patterns that map to the ***bookWriter*** relation in NELL sorted by their confidences – that is, the probability of the verb pattern given the relation $P(v|r)$.



Verbs learned by EM (naive bayes) with type checking and prior

concept:agriculturalproductcutintogeometricshape
concept:weaponmadeincountry
concept:statehascapital
concept:agentcollaborateswithagent
concept:bookwriter
concept:statelocatedingeopoliticallocation
concept:agenthierarchicallyaboveagent
concept:productinstanceof
concept:writerwasbornincity
concept:organizationterminatedperson

concept:hassibling

Verb	Confidence
arg1 resemble arg2	0.26650
arg2 kill arg1	0.03327
arg1 kill arg2	0.03327
arg2 praise arg1	0.02125
arg1 command arg2	0.01752
arg2 command arg1	0.01671
arg2 rise against arg1	0.01444
arg1 rise against arg2	0.01444

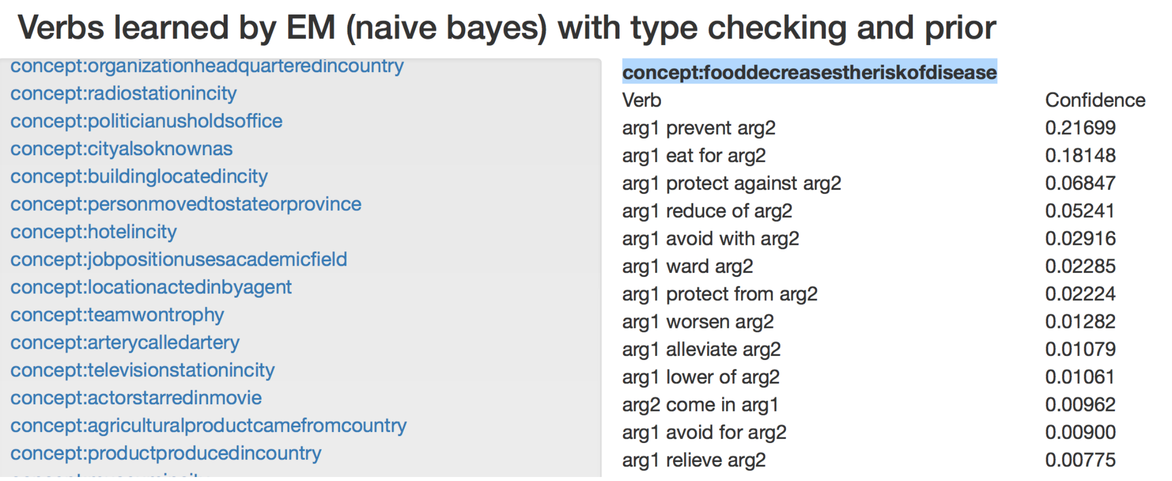
Figure 3.9: Our website showing some of the verb patterns that map to the ***hasSibbling*** relation in NELL sorted by their confidences

In Figure 3.8, we show our website[10] that contains the mappings between typed verbs to relations in NELL. A qualitative analysis of the mappings shows areas where the current approach still needs improvements.

Since we work only with verb patterns that are parts of the subject-verb-object (SVO) triples construction, our extracted verb patterns are mostly agentive. In Figure 3.9 we show an example relation (***hasSibbling***) that does not have any agentive verb that can express it. Unlike relations such as ***hasFather*** or ***hasMother***, which have agentive verbs that can express them like "fathered" or "mothered"; relations such as ***hasBrother***, ***hasSister*** or ***hasSibbling*** do not have the same type of verbs that can express them – i.e., there is no verb like "brothered" or "sistered" or "sibblinged" that can indicate the relations. Our approach instead learns patterns such as "kill" for the ***hasSibbling*** relation. An investigation of the SVO triples that have the verb "kill" and have their subject and object pairs being instances of the ***hasSibbling*** relation shows that "kill" is mapped to ***hasSibbling*** because the verb frequently occurs with the entity pair (*Cain*, *Abel*), which is an instance of the ***hasSibbling*** relation. Since our method is frequency based and does not distinguish the source documents of the SVO triples, we end up with this somewhat incorrect mapping.

In Figure 3.10, we show some of the verbs that map to the relation ***foodDecreasesTheRiskOfDisease*** in NELL. As we can see in the figure, the typed verb "eat for"(*food*, *disease*), which means "to prevent", is one of the highest ranked verbs for this relation. However, not only that this verb should be in a passive voice i.e., "(passive) eat for"(*food*, *disease*); "eat for" is also ambiguous because it can be used to mean "to cause" e.g., in the sentence: "fiber is eaten for weight loss". In this case, the typed verb is "(passive) eat for"(*food*, *nonDiseaseCondition*) since *weight loss* is a ***physiologicalCondition*** that is not a *disease*. However, NELL does not have existing relations with that type that means "to cause". It has only the relation ***foodCanCauseDisease*** with the domain *food* and the range *disease* that means "to cause". This is an example of how typed verbs that cannot be mapped to any existing relation in NELL can be used to extend the vocabulary of relations in NELL. In this case, the typed verb "(passive) eat for"(*food*, *nonDiseaseCondition*) can be proposed as a member of the new relation ***foodCanCauseNonDiseaseCondition*** in NELL.

[10] http://www.cs.cmu.edu/%7Edwijaya/mapping.html



Verbs learned by EM (naive bayes) with type checking and prior

- concept:organizationheadquarteredincountry
- concept:radiostationincity
- concept:politicianusholdsoffice
- concept:cityalsoknownas
- concept:buildinglocatedincity
- concept:personmovedtostateorprovince
- concept:hotelincity
- concept:jobpositionusesacademicfield
- concept:locationactedinbyagent
- concept:teamwontrophy
- concept:arterycalledartery
- concept:televisionstationincity
- concept:actorstarredinmovie
- concept:agriculturalproductcamefromcountry
- concept:productproducedincountry

concept:fooddecreasestheriskofdisease

Verb	Confidence
arg1 prevent arg2	0.21699
arg1 eat for arg2	0.18148
arg1 protect against arg2	0.06847
arg1 reduce of arg2	0.05241
arg1 avoid with arg2	0.02916
arg1 ward arg2	0.02285
arg1 protect from arg2	0.02224
arg1 worsen arg2	0.01282
arg1 alleviate arg2	0.01079
arg1 lower of arg2	0.01061
arg2 come in arg1	0.00962
arg1 avoid for arg2	0.00900
arg1 relieve arg2	0.00775

Figure 3.10: Our website showing some of the verb patterns that map to the ***foodDecreasesTheRiskOfDisease*** relation in NELL sorted by their confidences

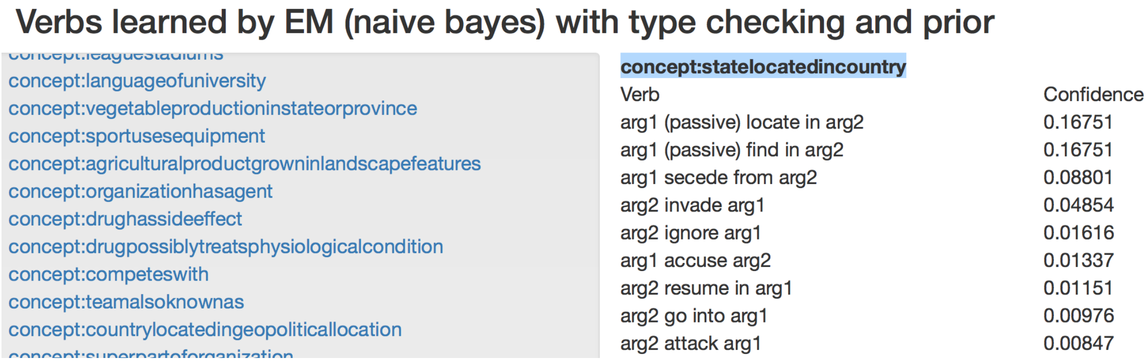
In Figure 3.11, we show some of the verbs that map to the relation ***stateLocatedInCountry*** in NELL. As we can see in the figure, there are typed verbs that are not quite correct for the relation such as "invade"(*country*, *state*). This may be caused by entailments. A *country* invading a *state* may entail the *state* being annexed into and therefore, being located in the *country*. But the verb "invade" itself does not normally express the "***locatedIn***" relation. However, the verb may share a lot of common subject-object pairs with the ***stateLocatedInCountry*** entity pairs that it is being mapped to the relation.

There are also typed verbs like "ignore"(*country*, *state*) or "accuse"(*state*, *country*) that are incorrectly mapped to the ***stateLocatedInCountry*** relation. This may be a problem caused by the use of metonymy – i.e., for the typed verb "ignore"(*country*, *state*) it may be that it is the *government* of the *country* that is ignoring the *state*, not the *country* itself. The use of metonymy may cause these verbs to share a lot of common subject-object pairs with the ***stateLocatedInCountry*** entity pairs that the verbs are being mapped to the relation.

With the entailment and the metonymy problems, the same subject-object pair may be used with different verbs in different frames thus the same subject-object pair does not necessarily have the same relation across frames. In our approach, we have used type checking and the constraints among relations as additional cues for learning the mapping so that we do not depend only on the verbs' subject-object pair overlap. However, more can be done such as adding constraints based on synonymy or antonymy relations between verbs that we explore in Chapter 5. Future work can address how to deal further with these problems.

3.9 Related Work on Mapping Verbs To KB Relations

Existing verb resources are limited in their ability to map to KBs. Some existing resources classify verb lexemes into semantic classes manually (e.g. WordNet Miller et al. [1990]) or classify verbs automatically (e.g. DIRT Lin and Pantel [2001b]). However, these classes are



Verbs learned by EM (naive bayes) with type checking and prior

concept:leaguestadiums
concept:languageofuniversity
concept:vegetableproductioninstateorprovince
concept:sportusesequipment
concept:agriculturalproductgrowninlandscapefeatures
concept:organizationhasagent
concept:drughassideeffect
concept:drugpossiblytreatsphysiologicalcondition
concept:competeswith
concept:teamalsoknownas
concept:countrylocatedingeopoliticallocation
concept:superpartoforganization

concept:statelocatedincountry

Verb	Confidence
arg1 (passive) locate in arg2	0.16751
arg1 (passive) find in arg2	0.16751
arg1 secede from arg2	0.08801
arg2 invade arg1	0.04854
arg2 ignore arg1	0.01616
arg1 accuse arg2	0.01337
arg2 resume in arg1	0.01151
arg2 go into arg1	0.00976
arg2 attack arg1	0.00847

Figure 3.11: Our website showing some of the verb patterns that map to the ***stateLocatedInCountry*** relation in NELL sorted by their confidences

not directly mapped to KB relations. Other resources provide relations between verb lexemes and their arguments in terms of semantic roles (e.g. PropBank Kingsbury and Palmer [2002], VerbNet Kipper et al. [2000], FrameNet Ruppenhofer et al. [2006]). However, it is not directly clear how the verb lexemes map to relations in specific KBs.

Most existing verb resources are also manually constructed and not scalable. A verb resource that maps to KBs should grow in coverage with the KBs, possibly by leveraging large corpora such as the Web for high coverage mappings. Our previous work, Wijaya et al. [2013], leverages Web-text as an interlingua. However, in that work, we used it to map KBs to KBs and obtain verbs-to-relations mappings only indirectly. We also compute heuristic confidences in verbs-to-relations mappings from label propagation scores, which are not probabilities. In contrast, in this work we map typed verbs directly to relations, and obtain $P(v_p|r_i)$ as an integral part of our EM process.

In terms of systems that learn mappings of textual patterns to KB relations, CPL Carlson et al. [2010] is one system that is most similar to our proposed approach in that it also learns text patterns for KB relations in a semi-supervised manner and uses constraints in the KB ontology to couple the learning to produce extractors consistent with these constraints. However, CPL uses a combination of heuristics in its learning, while we use EM. In our experiments, we use CPL patterns that contain verbs as priors and show that our approach outperforms CPL in terms of effectiveness for extracting relation instances.

In terms of the relation extraction, there are distantly-supervised methods that can produce verb pattern groupings as a by-product of relation extraction. One state-of-the-art uses matrix factorization and universal schemas to extract relations Riedel et al. [2013]. In this work, they populate a database of a universal schema (which involves surface form predicates and relations from pre-existing KBs such as Freebase) by using matrix factorization models that learn latent feature vectors for relations and entity tuples. One can envision obtaining a verb pattern grouping for a particular relation by predicting verb pattern surface forms that occur between entity tuples that are instances of the relation. However, unlike our proposed method that learns mappings from typed verbs to relations, they do not incorporate argument types in their learning, preferring to learn latent entity representation from data. Although this improves relation extraction, they observe that it hurts the performance of surface form prediction because a single surface pattern (like “visit”) can have multiple argument types (person-visit-location, person-visit-person, etc). Unlike our method, it is not clear in their method how argument types of surface patterns can be dealt with. Furthermore, it is not clear how useful prior constraints between relations (*subset*, *mutex*, etc.) can be incorporated in their method.

3.10 Analysis and Discussion

We have observed how type signatures can help resolve ambiguities of what relation a verb pattern expresses. For example, the verb pattern “play” can express either ***actorStarredInMovie*** or ***musicianPlaysInstrument*** depending on whether its subject and object type pair is (*actor*, *movie*) or (*musician*, *musicalInstrument*). In the experiments, we have seen how type signatures help improve the performance of a relation extractor that uses verbs as features. However, we believe that not all ambiguities can be resolved using type signatures. When two relations have the same type, for example, ambiguities can come from the verb, the preposition or the arguments.

For example, for the relation ***hasBrother***(*person*, *male*) and ***hasHusband***(*person*, *male*), the typed verb “have”(*person*, *male*) is possible for both i.e, this typed verb is ambiguous. Only when we use the common nouns following the verb can we extract the correct relation: “have a brother” v.s. “have a husband”. However, because we only extract verb patterns (lemmatized verb phrases that match the regular expression V | VP, see section 1.4), we have no way, at least in this current mapping, to disambiguate this typed verb “have”(*person*, *male*).

This example is a shortcoming of our mapping that stems from the way we extract verbs from the corpora in this thesis. Currently, we extract verb patterns that are part of the subject-verb-object construct. This is but one of the potentially many syntactic realizations possible for each verb. The SVO construct limits our coverage of verbs to be agentive, which in turn can restrict the types of relations that the verbs can cover. For example, this explains why our mapping can express relations such as ***fatherOfPerson*** well, using verbs such as “father” as in “X fathered Y” or ***motherOfPerson*** using verbs such as “mother” as in “X mothered Y”; but not relations such as ***hasBrother*** or ***hasSister*** for which there does not exist an equivalent agentive verb such as “X brothered Y” or “X sistered Y”. It will be interesting for the future work to explore if relations such as these can be expressed better if we also extract the nouns following the verb ending with a preposition, perhaps in the manner of how ReVerb [Fader et al., 2011] extracts their relational verb patterns. To extract their verb patterns, ReVerb uses part-of-speech-based regular expression V | VP | VW*P where W = (noun | adjective | adverb | pronoun | determiner) that allows extraction of verb patterns that are either a verb (e.g., “marry”), a verb followed immediately by a preposition (e.g., “live in”), or a verb followed by nouns, adjectives, or adverbs ending in a preposition (e.g., “is the brother of”).

Another example of an ambiguous typed verb is the verb “eat for”(*food*, *physiologicalCondition*) for the relation ***foodCanCausePhysiologicalCondition***(*food*, *physiologicalCondition*) v.s. ***foodDecreasesTheRiskOfPhysiologicalCondition***(*food*, *physiologicalCondition*). The typed verb “eat for”(*food*, *physiologicalCondition*) can express both relations: “eat for weight loss” v.s. “eat for diabetes”. Depending on whether the physiological condition that is the object of the typed verb is desirable (weight loss) or undesirable (diabetes), the preposition “for” in “eat for" can mean either "to cause" or "to prevent". Hence, the preposition is also ambiguous. This case, for example, can be resolved by splitting the category of physiologicalCondition into *disease* and *nonDiseaseCondition*, and splitting the relation ***foodCanCausePhysiologicalCondition*** into ***foodCanCauseDisease*** v.s. ***foodCanCauseNonDiseaseCondition***. In this case the typed verb "eat for"(*food*, *physiologicalCondition*) in "eat for weight loss" can express ***foodCanCauseNonDiseaseCondition*** while the typed verb "eat for"(*food*, *physiologicalCondition*) in "eat for diabetes" can express ***foodDecreasesTheRiskOfDisease***. For future work, it will be interesting to explore if we can learn from the distribution of subjects or objects of such ambiguous typed verbs to decide on whether to split the categories of the subjects or objects into finer categories in order to reduce the ambiguity.

Lastly, this discussion about ambiguous typed verbs and how far type signatures can help point to the deeper question of how verbs express relations. As Gentner [1982] aptly states "it is not perceiving relations but packaging and lexicalizing them that is difficult". In terms of verbs in a particular syntactic frame that express relations, the question that would be interesting to explore is how much is the mapping from verbs (to their underlying relations in the real-world) dictated by the meaning of the verbs and/or the prepositions, and how much is it dictated by the anchoring of the arguments and/or types to the realworld? In other words, if verbs are hooked to languages by their arguments that act as object-reference mappings, how much does the cohesiveness of the argument types of the relations (how concrete they are v.s. how abstract) dictate how easy it is to relate verbs to the underlying relations?

3.11 Conclusion

In this section, we have introduced a scalable EM-based approach with type checking and ontological constraints to automatically map typed verbs to KB relations by using the mentions of the verb patterns with the relation instances in a very large unlabeled text corpus. We demonstrate that our verb resource is effective for extracting KB relation instances while improving recall of both the supervised- and the unsupervised- verbs-to-relations mappings; highlighting the benefit of semi-supervised learning on unlabeled Web-scale text. We also show the flexibility of our method. Being KB-, and language-independent, our method is able to construct a verb resource for any language, given a KB and a text corpus in that language. We illustrate this by building verb resources in Portuguese and in English which are both effective for extracting KB relations. For future work, we want to explore the use of our multi-lingual verb resource for relation extraction by reading natural language text in multiple languages. We also make our mappings from typed verbs to the English and the Portuguese NELL relations publicly available, separately [11] and as part of our knowledge base of verbs [12].

[11] http://www.cs.cmu.edu/%7Edwijaya/mapping.html

[12] http://www.dwijaya.org/dvkb.html#DKVB

Chapter 4

Mapping Verbs to Changes in Knowledge Base Relations

4.1 Introduction

In this chapter, which is based on our previously published paper [Wijaya et al., 2015], we present an algorithm that learns the second semantics about verbs that we include in our VerbKB, namely the mappings from verbs to *changes* in knowledge base relations.

The algorithm learns the mappings from correlated Wikipedia article text and updates to its infobox (that contains DBPedia relations) in Wikipedia edit history. When a state-changing event, such as a marriage or death, happens to an entity, the infobox on the entity's Wikipedia page usually gets updated. At the same time, the article text may be updated with verbs either being added or deleted to reflect the changes made to the infobox. We use Wikipedia edit history to distantly supervise a method for automatically learning verbs and state changes. Additionally, our method uses constraints to effectively map verbs to infobox changes. To the best of our knowledge, there is no pre-existing resource for verbs that automatically maps verbs to changes in knowledge base relations.

We show in the experiments, that the mappings from verbs to changes in relations in the infobox are also effective for predicting Wikipedia infobox updates when verbs are added or deleted from the corresponding Wikipedia article text. A knowledge base of verbs that contains information about these state-changing verbs can also be useful for updating knowledge in knowledge bases, specifically in adding temporal scopes to relation instances (i.e., facts) in the knowledge bases. Temporal scope adds a time dimension i.e., the *begin time* and *end time* to facts in knowledge bases. These time scopes specify respectively the time periods when a given fact was valid in real life, i.e., when it begins to be valid and when it ceases to be valid. Without a temporal scope, many facts are under-specified, reducing the usefulness of the data for upper-level applications such as Question Answering. Although extracting relational facts between entities and storing them in knowledge bases (KBs) has been a topic of active research in recent years, the resulting KBs are generally static and are not updated as the facts change Suchanek et al. [2007], Carlson et al. [2010], Fader et al. [2011], Mitchell et al. [2015]. One possible approach to updating KBs is to extract facts from dynamic Web content such as news

Nakashole and Weikum [2012].

Instead of updating KBs by extracting dynamic facts – labeling entity pairs found in dynamic Web content with relation labels – our algorithm *predicts* state changes (in terms of changes in knowledge base relations) caused by verbs acting on entities in text. Instead of labeling entity pairs with relation labels, we label the *verbs* occurring between the entities with labels that indicate the initiation/termination of KB relations.

Consider, for example, the ***spouse*** relation. Relation extractors such as CPL Carlson et al. [2010] – which extracts instances of relations in NELL – considers both the verb pattern "marry" and "divorce" as good patterns for extracting instances of ***spouse***[1]. In contrast, our algorithm learns that these two verbs, "marry" and "divorce", cause different state changes in the knowledge base: "marry" initiates ***spouse*** relation while "divorce" terminates ***spouse***. The algorithm labels the verb pattern "marry" with the label "*begin-spouse*" (which indicates the initiation of the ***spouse*** relation) and labels the verb pattern "divorce" with the label "*end-spouse*" (which indicates the termination of the ***spouse*** relation). We can use this information to then update the entity's fact *and* its temporal scope Wijaya et al. [2014a]. For example, when an entity pair occurs as a subject and object pair to the verb pattern "marry", we initiate a ***spouse*** instance in the KB between the entities in the pair, and add a *begin time* to the relation instance. On the other hand, when an entity pair occurs as a subject and object pair to the verb pattern "divorce", we terminate the ***spouse*** instance between the entities in the pair by adding an *end time* to the relation instance.

From the experiments we conducted, we observe how learning state-changing verbs (verbs that initiate or terminate knowledge base relations) can be also useful for updating relation instances in the knowledge base [Wijaya et al., 2014a, 2015]. Specifically, we observe in our experiments that when state-changing verbs are added or deleted from an entity's Wikipedia page text, we can predict the entity's infobox updates with 88% precision and 76% recall. Therefore, one compelling application of learning these verbs is to incorporate them as triggers in methods for updating existing knowledge bases, which are currently mostly static. More specifically, once the algorithm is trained i.e., once it learns which verbs can cause which changes in KB relations, when these verbs are added or removed from text – any text beyond Wikipedia such as news text, we can update the corresponding KB relations of the verbs' subjects and objects accordingly.

4.2 Motivating Study

Before we present the algorithm that learns the mappings from verbs to changes in knowledge base relations, we present our motivation for learning the mappings that is based on our previous published papers [Wijaya and Yeniterzi, 2011, Wijaya et al., 2014a]. In these two papers, we explore the idea and implement various methods to automatically identify state changes that happen to an entity based on its *context* (words surrounding the entity). The underlying assumption is that when a state-changing event happens to an entity, it changes state and either the event or this change or both are reflected in the context of the entity i.e., in words that co-occur with the entity.

[1] The full list of extraction patterns that CPL learns for ***hasSpouse*** relation can be browsed in http://rtw.ml.cmu.edu/rtw/kbbrowser/predmeta:hasspouse

The goal of the first paper [Wijaya and Yeniterzi, 2011] was to *identify when* and *what* state changes happen to an entity based on how mention frequencies of words in its context change over time. Assuming that the *when* (time) and the *what* (change that happens to an entity) are given e.g., from knowledge about the entity in a knowledge base such as NELL, the goal of the second paper [Wijaya et al., 2014a] was to *extract* words in the context of the entity at the time that indicate the change or the event that brings about the change.

What we learn from these two works is that it is possible to find words that indicate the change that happens to an entity and/or the event that brings about the change to the entity in the words that surround the entity. This motivates us to come up with the algorithm that maps verbs to changes in an entity's knowledge base relations by correlating the addition/deletion of verbs to/from its context with the simultaneous knowledge base updates to its relations.

4.2.1 Identify Entity Changes

In [Wijaya and Yeniterzi, 2011], we present a method that automatically identifies changes that occur to an entity based on the frequency changes in the *context* words that surround the entity over time.

The method starts by obtaining words that surround the entity over time, with the year granularity, from Google Books NGram dataset [Michel et al., 2011]. This dataset contains 1-gram to 5-gram extracted from the text of around 5 million digitized books and their frequencies (how often these n-grams were used over time): their match counts, page counts and volume counts each year; with the year ranging from as early as the 1500s and as late as 2008. In general, for this method to work, we can use any corpus that contains documents labeled with their date creation times at any granularity e.g., GigaWord [Graff et al., 2003].

The method then clusters words that surround the entity over time and identifies *when* (at which year) changes occur, and also *what* changes occur (i.e., what clusters of words are in transition).

We also find that for the entities we test, the period that our method identifies coincides precisely with events that correspond to the change. For example, for the word "gay", our approach is able to identify the transition of the use of the word as an adjective for happiness, cheerfulness, pleasantness etc., to its use as a noun with the meaning homosexual man. As can be seen in Figure 4.1, the transition occurs on and around the year 1970 which is the year that homosexual movement starts to gain traction.

Another example, for the entity *Iran*, our approach is able to identify the country's transition from a monarchy to an Islamic republic with a new cluster consisting of words such as *republic* and *revolution* emerging around the entity after 1978 (1979 is the year of the Islamic revolution).

Another similar example can be seen with the entity *Kennedy*. Our approach was able to identify the John F. Kennedy the *senator* before the election (one cluster of words surrounding the entity) and the John F. Kennedy the *president* after the election (another cluster of words surrounding the entity). The transition between the two clusters is at 1961, the exact year Kennedy was elected. Similar changes are observed for the entity *Clinton*, from *governor* to *president*. The transition occurs at 1993, the exact year he was elected.

We learn two important insights from this work: (1) that some changes occurring to an entity can be identified from the frequency changes in the words surrounding the entity over time and

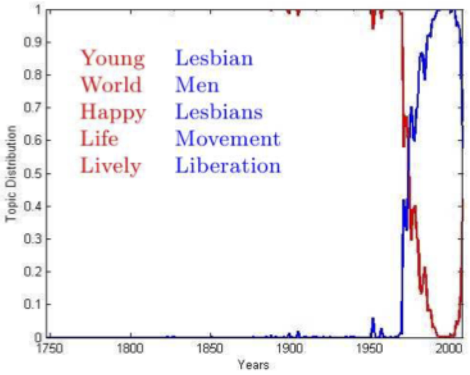


Figure 4.1: Topic-over-Time clustering results showing 2 topics for the word "gay" that transition in the year 1970.

(2) that some changes occurring to an entity can coincide with events happening to the entity.

4.2.2 Extract Change Indicators

In the following work [Wijaya et al., 2014a], we present a method that extracts among words that surround a changed entity, particular words that indicate the change or the event that brings about the change. Further, we explore if these words can be useful for knowledge base updates, specifically for temporal scoping. Specifically, through the change in the frequencies of words that surround the entity, we model the entity's state change brought about by real-world events that happen to the entity (e.g., hired, fired, divorced, etc.). This leads to a new formulation of the temporal scoping problem as a *state change detection problem*. Our experiments show that this formulation of the problem and the resulting solution are highly effective for inferring temporal scope of relation instances.

Starting from a given event of interest and a knowledge base relation that corresponds to the event (e.g., ***hasSpouse*** NELL relation for the *marriage* event), the method takes a dozen or so temporally scoped instances of the relation as seed instances. The method then aggregates (by averaging) the word vectors of the seeds' entities *at the time of* the event (and *after* the event, respectively).

For example, for the *marriage* event, the method takes a dozen or so manually temporally scoped instances of the ***hasSpouse*** relation: e.g., ***hasSpouse***(Kanye_West, Kim_Kardashian, **begin time**: 24 May 2014), ***hasSpouse***(John_Legend, Christine_Teigen, **begin time:** 14 September 2013), etc. Then, the method aggregates the word vectors surrounding the entities: Kanye_West, Kim_Kardashian, John_Legend, Christine_Teigen *at the time of* the marriage event – on 24 May 2014 for Kanye_West and Kim_Kardashian and on 14 September 2013 for John_Legend and Christine_Teigen – and also aggregates the word vectors surrounding these entities *after* the marriage event – on 25 May 2014 for Kanye_West and Kim_Kardashian and on 15 September 2013 for John_Legend and Christine_Teigen.

To filter out the noise from the word vectors of seed entities, the method computes *tf-idf*

Relation	Context Vector	Change Vector
presidentOf(*person*, “USA”)	was elected, took office, became president	vice president (-), by president (+), administration (+), senator (-), governor (-), candidate (-)
wonAward(*movie*, “Academy Award for Best Picture”)	nominated for, to win, won the, was nominated	best picture (+), hour minute (-), academy award (+), oscar (+), nominated (+), won (+), star (-), best actress (+), best actor (+), best supporting (+)

Table 4.1: Example of various contextual units (unigrams and bigrams) in the aggregate context and aggregate change vectors for the relations ***presidentOf*** and ***wonAward***. The *(+)* and *(-)* signs indicate rise and fall in mention frequency, respectively. As we can see here, for the ***wonAward*** relation, the change vector contains mostly new contexts.

statistics for each vector and only retains the top k ranking units in the vector. In the experiments, we used $k = 100$. The method computes *tf-idf* by treating each time unit t as a document containing words that occur in the context of an entity [Wijaya and Yeniterzi, 2011]. To capture more context, instead of using words in the vector, we use unigrams and bigrams of words.

The average of the vectors of seed entities *at the time* of the event is called the aggregate *Context* Vector for the relation. The average of the *difference* between the vectors *on* and *after* the event of interest happens is called the aggregate *Change* Vector for the relation.

We observe, however, that the unigrams and bigrams in the aggregate context and change vectors of each relation reflect the meaningful events and state changes happening to the seed entities (Table 4.1). For example, after ‘becoming president’ and ‘taking office’, US presidents often see a drop in mentions of their previous job titles such as ‘senator’, ‘governor’ or ‘vice president’ as they gain the new ‘president’ job title.

For the task of temporal scoping, once the aggregate *context* and *change* vectors of a relation are computed from the seed instances, given an instance of the relation to temporally scope, we consider every time point t of its entity pair to be a candidate *begin* time. We then compare the context vector and the change vector of every candidate time point t to the aggregate context and change vectors for the relation. We use cosine similarity to measure similarities between the context vector and the aggregate context vector and between the change vector and the aggregate change vector. The highest-ranking candidate time point (most similar to the aggregate context and aggregate change vector) is then considered to be the *begin* time of the relation instance.

We observe in the experiments, that aggregate context and change vectors are effective for detecting *begin* times of relation instances compared to systems that do not take contexts into consideration. One such system is **CoTS** [Talukdar et al., 2012, Wijaya et al., 2012], a state of the art macro-reading system of temporal information that uses temporal profiles of relation instances (i.e., counts of their mentions over large number of documents over time) combined with manually specified constraints about the relations (their functionality, inclusion, exclusion, etc.) to temporally scope relation instances. Our approach gives comparable or better F1 (@k=1) than systems that use only plain temporal profiles, even when these systems are supplemented with many carefully crafted, hand-specified constraints (Figure 4.2). This shows how augmenting temporal profiles with context and change patterns (i.e., Contextual Temporal Profile) can be useful for detecting state change, which is an effective way of identifying *begin* times and updates of relation instances in the knowledge base.

We observe however, that this method works best for relations whose change in contexts is distinctive of the event. For example, the method works best for ***bestPicture*** and ***bestDirector*** (Figure 4.2) because their change vectors contain a lot of new contexts and they are distinctive of the ***wonAward*** event. As we can see, for ***wonAward*** relation that defines ***bestPicture*** or ***bestDirector*** depending on arguments, its change vector contains a large number of new contexts that were not seen before (Table 4.1). In contrast, the method works only comparably for ***president*** and ***secretaryOfState*** relations because for these relations the change in contexts is subtler with respect to the event. Specifically, many *president* and *secretaryOfState* entities are still mentioned a lot in texts as “president X” and “secretary Y” even after they no longer hold those positions. Furthermore, we note that the performance for the ***secretaryOfState*** relation is low in both **CoTS** and in our approach. We found that this was due to few documents mentioning the secretary of state in Google Books Ngram dataset. This leads to weak signals for predicting the temporal scope of secretary of state appointments.

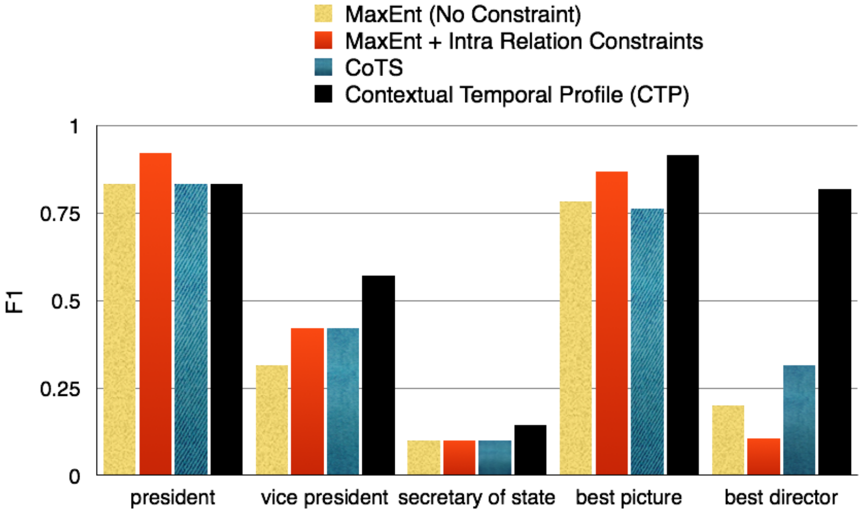
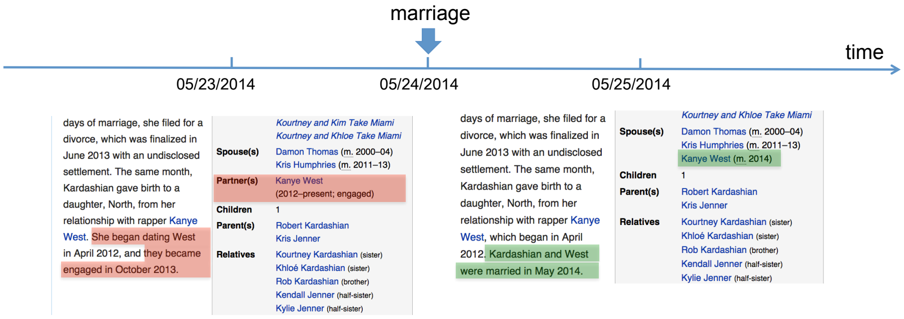


Figure 4.2: Comparison of F1 scores of our approach that uses Contextual Temporal Profile (CTP) with CoTS and other baselines.



days of marriage, she filed for a divorce, which was finalized in June 2013 with an undisclosed settlement. The same month, Kardashian gave birth to a daughter, North, from her relationship with rapper Kanye West. She began dating West in April 2012, and they became engaged in October 2013.

	Kourtney and Kim Take Miami *Kourtney and Khloe Take Miami*
Spouse(s)	Damon Thomas (m. 2000–04) Kris Humphries (m. 2011–13)
Partner(s)	Kanye West (2012–present; engaged)
Children	1
Parent(s)	Robert Kardashian Kris Jenner
Relatives	Kourtney Kardashian (sister) Khloé Kardashian (sister) Rob Kardashian (brother) Kendall Jenner (half-sister) Kylie Jenner (half-sister)

days of marriage, she filed for a divorce, which was finalized in June 2013 with an undisclosed settlement. The same month, Kardashian gave birth to a daughter, North, from her relationship with rapper Kanye West, which began in April 2012. Kardashian and West were married in May 2014.

	Kourtney and Khloe Take Miami
Spouse(s)	Damon Thomas (m. 2000–04) Kris Humphries (m. 2011–13) Kanye West (m. 2014)
Children	1
Parent(s)	Robert Kardashian Kris Jenner
Relatives	Kourtney Kardashian (sister) Khloé Kardashian (sister) Rob Kardashian (brother) Kendall Jenner (half-sister) Kylie Jenner (half-sister)

Figure 4.3: A snapshot of Kim Kardashian’s Wikipedia revision history, highlighting text and infobox changes. In red (and green) are the differences between the page on 05/25/2014 and 05/23/2014: things that are deleted from (and added to) the page. This particular revision history has label *begin-spouse*.

4.3 Overview of Method

Motivated by findings in our previous works that words in the context of changed entities can reflect events and state changes and that these words are effective for temporal scoping, we extend these works specifically in relation to discovering state-changing verbs (verbs that initiate or terminate knowledge base relations) .

Our algorithm learns state-changing verbs from Wikipedia revision history. In particular, we seek to establish a correspondence between infobox edits and verb pattern edits in the same article. The infobox of a Wikipedia article is a structured box that summarizes an entity as a set of facts (attribute-value pairs) . Our assumption is that when a state-changing event happens to an entity e.g., a marriage, its Wikipedia infobox is updated by adding a new SPOUSE value. At approximately the same time, the article text might be updated with verbs that express the event, e.g., “*Jolie is **married** to Pitt* in September ...”. Figure 4.3 is an example of an infobox of an entity changing at the same time as the article’s main text to reflect a marriage event.

Wikipedia revision history of many articles can act as distant supervision data for learning the correspondence between text and infobox changes. However, these revisions are very *noisy*. Many infobox slots can be updated when a single event happens. For example, when a death happens, slots regarding birth relations e.g., *birthdate*, *birthplace*, may also be updated or added if they were missing before. Therefore, our algorithm has to handle these sources of noise. We leverage logical constraints (detailed in section 4.6) to rule out meaningless mappings between infobox and text changes.

4.4 Data Construction and Design Choices

We construct a dataset from Wikipedia edit histories of entities whose facts change between the year 2007 and 2012 (i.e., have at least one fact in YAGO KB Suchanek et al. [2007] with a start or end time in this period). Besides limiting the time period to between 2007 and 2012, we also limit the types of entities we extract from Wikipedia edit histories to only *person* entities. This is a design choice that we made due to the large size of Wikipedia edit histories and the fact that the majority (65%) of Wikipedia pages are of entities of type *person*; and is not related to the algorithm itself. The algorithm can be applied to dataset constructed of entities of *any* category.

We obtain Wikipedia URLs of the set of *person* entities P from YAGO whose facts change between the year 2007 and 2012 and crawl their article's revision history. Given an entity p, its Wikipedia revision history R_p has a set of ordered dates T_p on which revisions are made to its Wikipedia page (we consider date granularity). Each revision $r_{p,t} \in R_p$ is its Wikipedia page at date t where $t \in T_p$.

Each Wikipedia revision $r_{p,t}$ is a set of infobox slots $S_{p,t}$ and textual content $C_{p,t}$. Each infobox slot $s \in S_{p,t}$ is a quadruple, $\langle s_{att}$, s_{value}, s_{start}, $s_{end}\rangle$ containing the attribute name (non-empty), the attribute value, and the start and end time for which this attribute-value pair holds in reality.

A document $d_{p,t}$ in our data set is the *difference*[2] between any two consecutive revisions separated by more than 24 hours i.e., $d_{p,t} = r_{p,t+2} - r_{p,t}$, where $r_{p,t+2}$ is the *first* revision on date $t + 2$ and $r_{p,t}$ is the *last* revision on date t (as a page can be revised many times in a day).

A document $d_{p,t}$ is, therefore, a set of infobox changes $\Delta S_{p,t}$ and textual changes $\Delta C_{p,t}$. Each slot change $\delta s \in \Delta S_{p,t} = \langle s_{att}$, δs_{value}, δs_{start}, $\delta s_{end}\rangle$ is prefixed with $+$ or $-$ to indicate whether they are added or deleted in $r_{p,t+2}$. Similarly, each text change $\delta c \in \Delta C_{p,t}$ is prefixed with $+$ or $-$ to indicate whether they are added or deleted.

For example, in Figure 4.3, a document $d_{kim,\ 05/23/2014} = r_{kim,05/25/2014} - r_{kim,05/23/2014}$ is a set of slot changes: $\langle$SPOUSE, $+$"Kanye West", $+$"2014", " "$\rangle$, $\langle$PARTNER, $-$"Kanye West", $-$"2012-present; engaged", " "$\rangle$ and a set of text changes: $+$"Kardashian and West were married in May 2014", $-$"She began dating West", $-$"they became engaged in October 2013".

For each $d_{p,t}$, we use $\Delta S_{p,t}$ to label the document and $\Delta C_{p,t}$ to extract features for the document. We label $d_{p,t}$ that has a new value or start time added to its infobox: $\langle s_{att}, +\delta s_{value}, *, *\rangle \in \Delta S_{p,t}$ or $\langle s_{att}, *, +\delta s_{start}, *\rangle \in \Delta S_{p,t}$ with the label *begin-*s_{att} and label $d_{p,t}$ that has a new end time added to its infobox: $\langle s_{att}, *, *, +\delta s_{end}\rangle \in \Delta S_{p,t}$ with the label *end-*s_{att}.

The label represents the state change that happens in $d_{p,t}$. For example, in Figure 4.3, $d_{kim,\ 05/23/2014}$ is labeled with *begin-spouse*.

The revision history dataset that we make available[3] for future research consists of all documents $d_{p,t}$, labeled and unlabeled, $\forall t \in T_p,\ t \in [01/01/2007,\ 12/31/2012]$, and $\forall p \in P$; a total of 288,184 documents from revision histories of 16,909 Wikipedia entities. Using our labeling process, we find that out of 288,184 documents, only 41,139 have labels (i.e., have their infobox updated with new values/start/end time). For person entities, the distribution of labels in the dataset is skewed towards birth and death events as these are life events that happen to almost all person entities in Wikipedia. The distribution of labels in the dataset that we release can be seen in Figure 4.4[4]. We show only labels that we evaluate in our task.

For our task of learning state-changing verbs from this revision history dataset, for each labeled $d_{p,t}$, we extract as features, verb pattern $v \in \Delta C_{p,t}$ (lemmatized verb phrase that matches our part-of-speech-based regular expression: V$|$VP, see section 1.4) and whose subject (or object)

[2] a HTML document obtained by "compare selected revisions" functionality in Wikipedia

[3] The dataset can be downloaded from http://www.cs.cmu.edu/%7Edwijaya/postcondition.html

[4] The labels are Wikipedia infobox attribute names. We do not normalize the names, hence there are labels "begin-club" and "begin clubs", which are two distinct Wikipedia infobox attribute names.

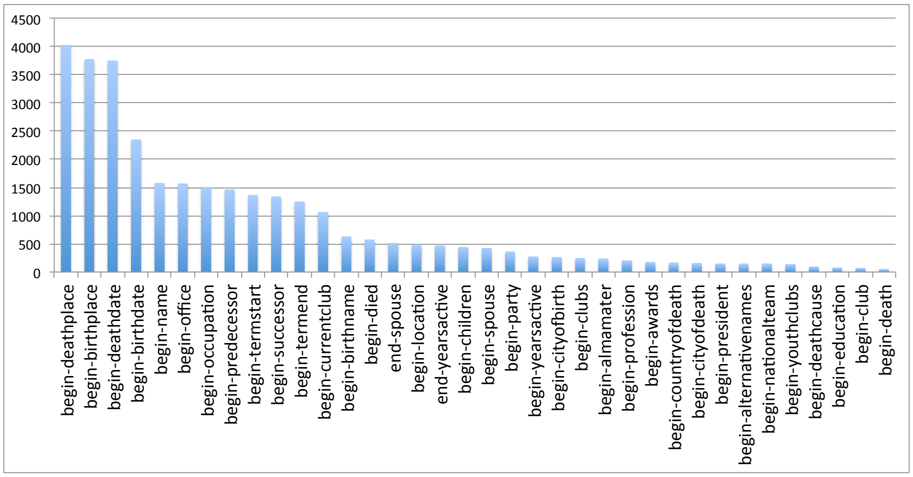


Figure 4.4: Distribution of labels we evaluate in our task in the revision history dataset.

matches the Wikipedia entity p *and* whose object (or subject resp.) matches an infobox value, start or end time: ($v_{subject}$, v_{object}) = ($_{arg1}$, $_{arg2}$) or ($v_{subject}$, v_{object}) = ($_{arg2}$, $_{arg1}$), where $_{arg1}$= p and $\langle s_{att}, _{arg2}, *, * \rangle$ or $\langle s_{att}, *, _{arg2}, * \rangle$ or $\langle s_{att}, *, *, _{arg2} \rangle \in \Delta S_{p,t}$. We use the Stanford CoreNLP Manning et al. [2014] to dependency parse sentences and extract the subjects and objects of verbs. We find that 27,044 out of the 41,139 *labeled* documents contain verb pattern edits, but only 4,735 contain verb pattern edits with a subject and an object, where the subject matches the entity and the object matches the value of the infobox change or vice versa. We use the latter for our task, to improve the chance that the verb pattern edits used as features are related to the infobox change.

4.5 Model

We use a Maximum Entropy (MaxEnt) classifier[5] given a set of training data = {($\mathbf{v}_{d_\ell}$, y)} where $\mathbf{v}_{d_\ell} = (v_1, v_2, ... \, v_{|V|}) \in R^{|V|}$ is the $|V|$-dimensional representation of a labeled document d_ℓ where V is the set of all verbs in our training data, and y is the label of d_ℓ as defined in 4.4.

These training documents are used to estimate a set of weight vectors $\mathbf{w}$ = $\{\mathbf{w}_1, \mathbf{w}_2, ... \, \mathbf{w}_{|Y|}\}$, $\mathbf{w}_y \in R^{|V|}$, one for each label $y \in Y$, the set of all labels in our training data. The classifier can then be applied to classify an unlabeled document d_u using:

$$p(y|\mathbf{v}_{d_u}) = \frac{\exp(\mathbf{w}_y \cdot \mathbf{v}_{d_u})}{\sum_{y'} \exp(\mathbf{w}_{y'} \cdot \mathbf{v}_{d_u})} \tag{4.1}$$

4.6 Feature Selection using Constraints

While feature weights from the MaxEnt model allow us to identify verbs that are good features for predicting a particular state change label, our distantly supervised training data is inherently

[5] We use Mallet implementation of MaxEnt: http://mallet.cs.umass.edu/

noisy. Changes to multiple infoboxes can happen within our revision. We therefore utilize constraints among state changes to select consistent verb pattern features for each type of state change.

We use two types of constraints: (1) mutual exclusion (*Mutex*) which indicate that mutex state changes do not happen at the same time e.g., updates on *birthdate* should not *typically* happen with updates on *deathcause*. Hence, their state-changing verbs should be different. (2) Simultaneous (*Sim*) constraints which indicate that simultaneous state changes should *typically* happen at the same time e.g., update on *birthdate* should *typically* happen with other birth-related updates such as *birthplace*, *birthname*, etc. We manually specified these two types of constraints to all infobox pairs where they apply. We have 10 mutex constraints and 23 simultaneously updated constraints. The full list of constraints can be found in our website[6].

Given a set of constraints, a set of labels Y, and a set of base verbs[7] B in our training data, we solve a Mixed-Integer Program (MIP) for each base verb b in B to estimate whether b should be a feature for state change $y \in Y$.

We obtain label membership probabilities $\{P(y|b) = count(y,b) / \sum_{y'} count(y',b)\}$ from our training data. The MIP takes the scores $P(y|b)$ and constraints as input and produces a bit vector of labels $\mathbf{a}_b$ as output, each bit $a_b^y \in \{0,\ 1\}$ represents whether or not b should be a feature for y.

The MIP formulation for a base verb b is presented by Equation 4.2. For each b, this method tries to maximize the sum of scores of selected labels, after penalizing for violation of label constraints. Let $\zeta_{y,y'}$ be slack variables for *Sim* constraints, and $\xi_{y,y'}$ be slack variables for *Mutex* constraints.

Solving MIP per base verb is fast since we conduct it over the output of MaxEnt. We only consider a label y to be a candidate for b if there exists in the MaxEnt model, a verb pattern with base form b that has a positive weight for the label i.e., if $\exists\ v_i \in V$ s.t. $w_y^i > 0$ and b = base form of v_i.

After we output $\mathbf{a}_b$ for each b, we select features for each label. We only select a verb pattern v_i to be a feature for y if the learned weight $w_y^i > 0$ and $a_b^y = 1$, where b = the base form of v_i. Essentially for each label, we select verb patterns that have positive weights and are consistent with the label as features.

$$\begin{aligned}
\underset{\mathbf{a}_b,\ \zeta_{y,y'},\ \xi_{y,y'}}{\text{maximize}} \quad & \Bigg(\sum_y a_b^y * P(y|b) - \sum_{\langle y,y'\rangle \in Sim} \zeta_{y,y'} - \\
& \sum_{\langle y,y'\rangle \in Mutex} \xi_{y,y'} \Bigg) \\
\text{subject to} \quad & \big(a_b^y - a_b^{y'}\big)^2 \le \zeta_{y,y'}, \quad \forall \langle y,y'\rangle \in Sim \\
& a_b^y + a_b^{y'} \le 1 + \xi_{y,y'}, \quad \forall \langle y,y'\rangle \in Mutex \\
& \zeta_{y,y'},\ \xi_{y,y'} \ge 0,\ a_b^y \in \{0,1\}, \quad \forall y, y'
\end{aligned} \tag{4.2}$$

[6] http://www.cs.cmu.edu/%7Edwijaya/postcondition.html

[7] The verb root or base form of a verb (after removing preposition)

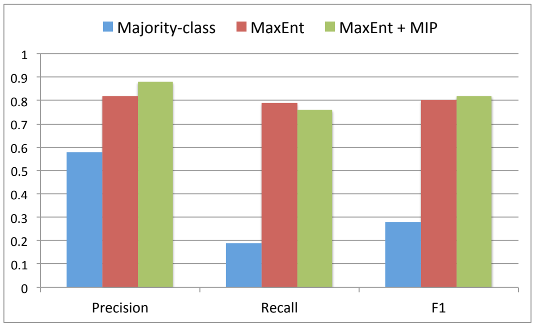


Figure 4.5: Results of predicting state change labels (infobox types) using verb pattern edits as features.

4.7 Experiments

We use 90% of our labeled documents that have verb pattern edits as features (section 4.4) as training data and test on the remaining 10%. Since revision history data is noisy, we manually go through our test data to discard documents that have incorrect infobox labels by looking at the text that changed. The task is to predict for each document (revision), the label (infobox slot change) of the document given its verb pattern features. We compute precision, recall, and F1 values of our predictions and compare the values before and after feature selection (Fig. 4.5).

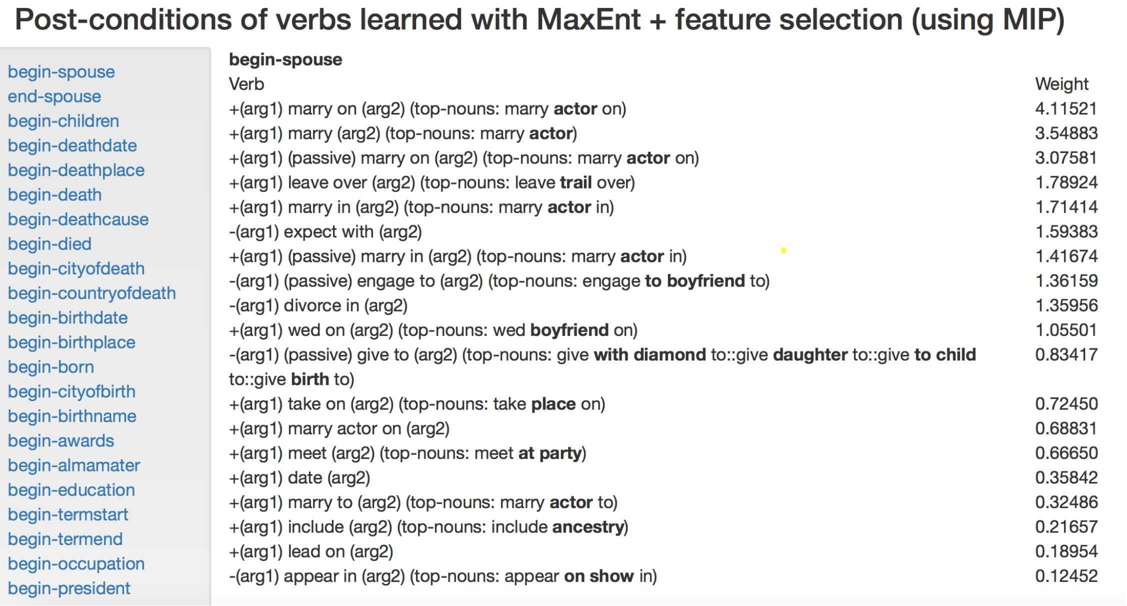
To the best of our knowledge, the task to learn state-changing verbs in terms of states defined in existing knowledge bases and learning it from Wikipedia edit histories is novel. There is no previous approach that can be used as a baseline; therefore we have compared our structured prediction using MIP and MAXENT with a majority class baseline that always predicts "begin-deathplace", which is the majority class label[8]. Both our approaches (MAXENT and MAXENT + MIP) perform better than the majority class baseline (Figure 4.5).

We observe the value of doing feature selection by asserting constraints in an MIP formulation. Feature selection improves precision; resulting in a better F1. By asserting constraints, some of the inconsistent verb pattern features for the labels were removed. For example, before feature selection, the verbs: "marry", and "be married to" were high-weighted features for both *begin-spouse* and *end-spouse*. After asserting constraints that *begin-spouse* is mutex with *end-spouse*, these verbs (whose base form is "marry") are filtered out from the features of *end-spouse*. We show some of the learned verb pattern features (after feature selection) for some of the labels in (Table 4.2). On average, we have about 18 verbs per infobox state change in our state changing verb resource that we make available for future research separately[9] (see Figure 4.6) and as part of our knowledge base of verbs[10].

[8] In the future, we can also compare to a MAXENT baseline that uses *all* word edits as features. Here we focus on verbs as we have observed from our previous works that in a lot of the contexts that surround an entity that changes state, these contexts contain verbs that express the state changing events.

[9] http://www.cs.cmu.edu/%7Edwijaya/postcondition.html

[10] http://www.dwijaya.org/dvkb.html#DKVB



Post-conditions of verbs learned with MaxEnt + feature selection (using MIP)

begin-spouse
end-spouse
begin-children
begin-deathdate
begin-deathplace
begin-death
begin-deathcause
begin-died
begin-cityofdeath
begin-countryofdeath
begin-birthdate
begin-birthplace
begin-born
begin-cityofbirth
begin-birthname
begin-awards
begin-almamater
begin-education
begin-termstart
begin-termend
begin-occupation
begin-president

begin-spouse

Verb	Weight
+(arg1) marry on (arg2) (top-nouns: marry **actor** on)	4.11521
+(arg1) marry (arg2) (top-nouns: marry **actor**)	3.54883
+(arg1) (passive) marry on (arg2) (top-nouns: marry **actor** on)	3.07581
+(arg1) leave over (arg2) (top-nouns: leave **trail** over)	1.78924
+(arg1) marry in (arg2) (top-nouns: marry **actor** in)	1.71414
-(arg1) expect with (arg2)	1.59383
+(arg1) (passive) marry in (arg2) (top-nouns: marry **actor** in)	1.41674
-(arg1) (passive) engage to (arg2) (top-nouns: engage **to boyfriend** to)	1.36159
-(arg1) divorce in (arg2)	1.35956
+(arg1) wed on (arg2) (top-nouns: wed **boyfriend** on)	1.05501
-(arg1) (passive) give to (arg2) (top-nouns: give **with diamond** to::give **daughter** to::give **to child** to::give **birth** to)	0.83417
+(arg1) take on (arg2) (top-nouns: take **place** on)	0.72450
+(arg1) marry actor on (arg2)	0.68831
+(arg1) meet (arg2) (top-nouns: meet **at party**)	0.66650
+(arg1) date (arg2)	0.35842
+(arg1) marry to (arg2) (top-nouns: marry **actor** to)	0.32486
+(arg1) include (arg2) (top-nouns: include **ancestry**)	0.21657
+(arg1) lead on (arg2)	0.18954
-(arg1) appear in (arg2) (top-nouns: appear **on show** in)	0.12452

Figure 4.6: Our website that shows the mappings from verb patterns to changes in DBPedia relations. The figure shows some of the verb patterns that predict the *initiation* of the ***spouse*** relation in DBPedia sorted by their confidences – learned MAXENT weights.

4.8 Related Work on Learning State-Changing Verbs

Learning from Wikipedia Revision History. Wikipedia edit history has been exploited in a number of problems. A popular task in this regard is that of Wikipedia edit history categorization Daxenberger and Gurevych [2013]. This task involves characterizing a given edit instance as one of many possible categories such as spelling error correction, paraphrasing, vandalism, and textual entailment Nelken and Yamangil [2008], Cahill et al. [2013], Zanzotto and Pennacchiotti [2010], Recasens et al. [2013]. Prior methods target various tasks different from ours.

Learning State-Changing Verbs. Very few works have studied the problem of learning state-changing verbs. Hosseini et al. [2014] learned state-changing verbs in the context of solving arithmetic word problems. They learned the effect of words such as *add*, *subtract* on the current state. The VerbOcean resource was automatically generated from the Web Chklovski and Pantel [2004]. The authors studied the problem of fine-grained semantic relationships between verbs. They learn relations such as if someone has bought an item, they may sell it at a later time. This then involves capturing empirical regularities such as "X buys Y" happens before "X sells Y". Unlike the work we present here, the methods of Chklovski and Pantel [2004], Hosseini et al. [2014] do not make a connection to KB relations such as Wikipedia infoboxes. Our vision paper, Wijaya et al. [2014b] gave high-level descriptions of a number of possible methods for learning state changing methods but did not implement any of them.

Label	Verbs
begin-deathdate	+(arg1) die on (arg2), +(arg1) die (arg2), +(arg1) pass on (arg2)
begin-deathplace	+(arg1) die in (arg2), +(arg1) die at (arg2), +(arg1) move to (arg2)
begin-birthplace	+(arg1) be born in (arg2), +(arg1) bear in (arg2), +(arg1) be born at (arg2)
begin-predecessor	+(arg1) succeed (arg2), +(arg1) replace (arg2), +(arg1) join cabinet as (arg2), +(arg1) join as (arg2)
begin-successor	+(arg1) lose **seat** to (arg2), +(arg1) resign on (arg2), +(arg1) resign from post on (arg2)
begin-termstart	+(arg1) be appointed on (arg2), +(arg1) serve from (arg2), +(arg1) be elected on (arg2)
begin-termend	+(arg1) resign on (arg2), +(arg1) step down in (arg2), +(arg1) flee in (arg2)
begin-spouse	+(arg1) marry on (arg2), +(arg1) marry (arg2), +(arg1) be married on (arg2), – (arg1) be engaged to (arg2)
end-spouse	+(arg1) file **for divorce** in (arg2), +(arg1) die on (arg2), +(arg1) divorce in (arg2) +(arg1) announce **separation** on (arg2)
begin-children	+(arg1) have **child** (arg2), +(arg1) raise daughter (arg2), +(arg1) raise (arg2)
begin-almamater	+(arg1) graduate from (arg2), +(arg1) attend (arg2), +(arg1) be educated at (arg2)
begin-awards	+(arg1) be awarded (arg2), +(arg1) be named on (arg2), +(arg1) receive (arg2)
begin-youthclubs	+(arg1) start career with (arg2), +(arg1) begin **career** with (arg2), +(arg1) start with (arg2)
begin-clubs	+(arg1) play for (arg2), +(arg1) play during career with (arg2), +(arg1) sign with (arg2), +(arg1) complete **move** to (arg2)

Table 4.2: Verbs learned for various infobox relation changes. The texts in bold are (preposition+) common noun that occur most frequently with the ⟨verb pattern, relation change⟩ pair in the training data.

4.9 Analysis and Discussion

In this thesis, we apply our algorithm to learn the mapping of verbs for *person* entities in Wikipedia. However, the algorithm is not specific to person entities; it can be applied to other types of entities as long as they have Wikipedia pages and corresponding infoboxes. For example, using the same approach, we can learn for a company entity (e.g., *Google*) that when it changes the ***key-people*** in its infobox, typed verbs such as “be promoted as”(*person*, *ceo*) is added to its Wikipedia page – this example is obtained from actual edits of *Google* Wikipedia page on 10 August 2015.

One compelling application of learning the mapping from verbs to changes in the knowledge base relations is to incorporate them as triggers in methods for updating existing knowledge bases, which are currently mostly static. For example, if a verb is being added or removed from the context of an entity in text, we can update knowledge about the entity in the knowledge base. However, the mapping that we learn here is based on Wikipedia text that has a specific nature: well structured, follows a chronological order, focuses on facts and events. Our conjecture is that the mapping that we learned here will be applicable only to texts that have a similar nature to Wikipedia, e.g., news texts that are well structured, follows a chronological order, and that focuses on facts and events. Future work can address how the mapping can be learned and applied to more general texts.

Furthermore, we realize that Wikipedia edit history alone may not be enough for learning state-changing verbs as they are restricted by relations that are in the infobox. To learn high coverage state-changing verbs, we need to learn from Web-scale time stamped corpora such as news text, GigaWord, and Google Books N-gram. One possible way to do that is to start from the mappings from typed verbs to knowledge base relations that we learn in Chapter 3. The task is then to determine for each verb pattern that maps to a relation, whether the verb pattern initiates the relation, terminates the relation, or neither (i.e., it just expresses the relation). The signals for determining this can be, for example, the changes in mention counts of the verb pattern and its subject and object in time-stamped documents, using method such as [Das Sarma et al., 2011] that discovers dynamic relationships between entities from changes in their mention counts in documents over time.

We also observe that a change in verb tenses in Wikipedia edits often signals a change in knowledge base relations. In this thesis, we lemmatize the verbs. Therefore, we lose information about verb tenses that may be useful for learning the mapping between verbs and changes in knowledge base relations (see our previous discussions about Wikipedia tense edits in section 1.1 and aspectual approaches to verb classification in section 2.1.2). Future work can consider changes in verb tenses as additional features for learning the mapping.

To learn state-changing verbs from time stamped text corpora, we may also need to utilize signals from several corpora of different nature such that the sparsity of changes in one may be compensated by the redundancy of another. For example, changes that are the effect of the verb pattern “marry” (e.g., the appearance of the adjective “married”, the nouns “spouse” or “husband”, or the verb patterns “have spouse” or “have husband” in the context of the entity) may not be mentioned in news document the day after the marriage event happened. But they may be mentioned for several months after. The sparsity of documents with the day granularity (GigaWord) can be overcome by the redundancy of documents with the year granularity (Google Books N-gram). Furthermore, in this thesis we only consider day granularity between Wikipedia edits. Future work can consider longer intervals between edits to capture state-changing events whose durations are longer than a day e.g., mergers and acquisitions.

We may also need to generalize changes that are the effects of state-changing verbs to their categories/types. For example, the verb pattern “elect” causes different changes when applied to different entities. We can cluster changes affected by a verb pattern to their category types and therefore relation values. For example, the nouns “president”, “vice president”, “governor”, “senator”, etc. that are the effect of the verb pattern “elect” can be clustered to the category *jobPosition*, which is the range of the relation ***hasJobPosition***. We can then generalize that the verb pattern “elect” causes the change in the relation ***hasJobPosition***.

We can also utilize signals from pre-existing linguistic resources such as WordNet and VerbNet to learn state-changing verbs. In WordNet, dictionary definitions of the verb lexemes may contain changes affected by the lexemes. For example, from the WordNet definition of the verb lexeme *alkalify* as “turn basic and less acidic”, we can infer that the effects of this verb lexeme are adjectives such as “basic” and “less acidic”. WordNet also contains antonymy relations between verbs that can be useful as constraints for learning the mapping. For example, given that “die” is the antonym of “live”, if we map “die” to the termination of the ***yearsActive*** relation, then we should not map “live” to this same change in the relation. Future work can consider adding antonymy and synonymy relations – which are useful for improving the coverage of the mapping to more verbs – as additional constraints for learning the mapping.

Lexical resource for verbs such as VerbNet also contains useful diagnostics for detecting changes affected by the verbs from their semantic predicates. For example, the verb lexeme *deport* that appears in the syntactic frame "Agent *deport* Theme *to* Destination" has this set of semantic predicates in VerbNet: CAUSE(Agent, Event), LOCATION(START(Event), Theme, ?Source), LOCATION(END(Event), Theme, Destination). We can use it to infer that *deport* initiates LOCATION changes. We can map these semantic predicates to knowledge base relations to get the changes in KB relation affected by the verb lexeme.

An interesting research direction will be to integrate signals from all these different sources to come up with the overall changes caused by the verbs on knowledge base relations.

In terms of the approach, we have presented here a discriminative approach for learning the mapping from verbs to *changes* in relations. However, the semantic that we ultimately want to learn in here – in line to what we have learned in Chapter 3, which is the probability of a verb pattern v given a relation r or $P(v \mid r)$ e.g., $P($"marry" | ***hasSpouse***$)$ – is to learn the probability of an *addition* or a *removal* of the verb pattern v to/from an arbitrary text given a *change* in the relation r or $P(\Delta\ v|\Delta\ r)$ e.g., $P($+"divorce" | end-***hasSpouse***$)$, which means the probability of an *addition* of the verb pattern "divorce" to the context of an entity in an arbitrary text given a *termination* of the entity's ***hasSpouse*** relation in the real-world. In the future, we can think of a generative approach that can better reflect this semantic that we want to ultimately learn.

4.10 Conclusion

In this section, we have presented an algorithm that uses Wikipedia edit histories as distantly labeled data to learn which verbs result in which state changes to entities, and experimentally demonstrate its success. We first constructed and curated a novel dataset from Wikipedia revision history that is tailored to our task. We showed that this dataset is useful for learning verb pattern features that are effective for predicting state changes in the knowledge base (KB), where we considered the KB to be infoboxes and their values. We have made available this set of distantly labeled training data on our website[11]. We also make available our learned mappings from verbs to state changes, as a resource for other researchers on the same website and as part of our knowledge base of verbs[12].

As future work, we wish to explore the usefulness of our verb resource to other KBs to improve KB freshness. This is important because existing KBs are mostly static.

We wish to also explore the application of the learned verb resource to domains other than Wikipedia infobox and text e.g., for predicting state changes in the knowledge base from news text. Specifically, given the learned verb resource that contains the mappings from verbs to changes in KB relations, when these verbs are added/deleted from text – *any* text beyond Wikipedia such as news text, whether we can update the corresponding KB relations of the verbs' subjects and objects effectively.

Additionally, most Wikipedia revisions only have text changes without the associated infobox change. Another line of future work is to also learn from these unlabeled documents.

[11] http://www.cs.cmu.edu/ dwijaya/postcondition.html

[12] http://www.dwijaya.org/dvkb.html#DKVB

Lastly, in our data construction, we make a design choice to extract only Wikipedia edit histories of *person* entities. Although the largest number (more than 65%) of Wikipedia pages are of *person* entities, it will be interesting to construct a dataset containing the rest of the entity types: *organization*, *location*, *film*, etc.; to apply the algorithm and release the learned verb resource for this dataset.

Chapter 5

Extending Knowledge Base Relations

5.1 Introduction

From our work of mapping typed verbs to relations in Chapter 3, we find that many typed verbs in the subject-verb-object triples (SVO triples) extracted from ClueWeb do not yet have mappings to any relation in NELL KB. This is because NELL has only a couple hundreds of relations and hence a limited coverage of the typed verbs in the SVO triples. For example, NELL does not have a relation for the typed verb "support" with type signature (*product*, *programmingLanguage*) even though this typed verb occurs very frequently (932k times) in the SVO triples.

In this chapter, we present an algorithm to extend the vocabulary of relations in NELL and hence its coverage of the SVO triples. The algorithm clusters semantically similar and similarly typed verbs in the SVO triples and propose the newly discovered clusters as new relations in NELL.

The benefits of extending the vocabulary of relations in NELL are many; for example, having more relations in the knowledge base can mean denser knowledge graph which has been shown to lead to better inferences [Gardner et al., 2013].

To extend the vocabulary of relations in the knowledge base, one can argue that it is sufficient to add every typed verb as a new relation in the knowledge base as in the OpenIE fashion [Fader et al., 2011]. For example, a typed verb "marry"(*person*, *person*) can be the relation ***personMarryPerson*** in the knowledge base while another typed verb "wed"(*person*, *person*) can be another relation ***personWedPerson*** in the knowledge base. However, besides the lack of generalization from having each individual typed verb be a separate relation in the knowledge base, it has also been shown that there are values in using clusters of semantically similar surface forms rather than just individual words for improving performance in tasks such as knowledge base inference [Gardner et al., 2015] or word embedding for dependency parsing [Ammar et al., 2016]. Going back to our hypothesis, we believe that it is possible to semi-automatically construct a verb resource that goes beyond current resources in terms of coverage and links to knowledge bases, by leveraging a combination of high coverage text corpora, a knowledge base with a rich type system over entities, and other pre-existing linguistic resources such as thesaurus and WordNet. Instead of adding every single typed verb as a new relation, our verb resource adds clusters of typed verbs as new relations in the knowledge base.

To create clusters of typed verbs, the algorithm groups typed verbs in the SVO triples into similarly typed and semantically similar clusters based on (1) the verbs' subject and object types or the verbs' selectional preferences, (2) their similarities based on their arguments (subject-object pairs in the SVO triples), (3) synonymy and antonymy constraints from Moby thesaurus[1] and WordNet[2]. Each cluster is then either mapped to an existing NELL relation or added as a new relation in NELL. The result is VerbKB, a knowledge base of English verbs[3] that contains 65,679 unique verb patterns mapped into 215,106 binary relations[4], each typed with semantic categories in NELL and organized into a subsumption taxonomy based on types. To the best of our knowledge, this is the largest knowledge base of English verbs to date, with the verbs occurring a total of over 2 billion times in ClueWeb or 98% of all subject-verb-object occurrences in ClueWeb.

A typed verb in VerbKB can be mapped to multiple relations; each relation expresses a particular verb sense and the verb's subject and object types. In terms of alignment with the coarse-grained verb senses induced from WordNet senses by the Oxford Dictionary of English (ODE) inventory [Navigli et al., 2007], the relations in VerbKB have the best alignment to these manually constructed synsets, compared to verb synsets in other automatically constructed large-scale resources such as PATTY [Nakashole et al., 2012] or PPDB [Cocos and Callison-Burch, 2016].

5.2 Overview of Method

VerbKB is constructed semi-automatically, leveraging (1) a high coverage text corpus (ClueWeb), (2) a rich type system and entity population in a large knowledge base (NELL), and (3) knowledge about verbs, specifically their synonymy and antonymy relations in pre-existing, manually constructed linguistic resources (Moby thesaurus[5] and WordNet). However, VerbKB goes well beyond these existing resources in terms of coverage and links to knowledge bases. It specifies meanings of semantically typed verbs by mapping the typed verbs to clusters that are relations in knowledge bases.

To create a resource for verbs with higher coverage than any other resources, we leverage a very large ClueWeb corpus and extract typed verbs from over 650 million subject-verb-object (SVO) triples that occur a total over 2.1 billion times in ClueWeb (see section 1.4 for our design choices with regards to the SVO triples).

We have also learned from the analysis of other automatically constructed pre-existing resources that distributional similarities based on lexical contexts alone are not enough to identify similar verbs (DIRT, PPDB, WORD2VEC). Particularly for verbs, what roles they take seem to matter in terms of computing their semantic similarities [Schwartz et al., 2016]. Since verbs' semantic roles are important for discovering verb similarities, in addition to using distributional similarities of typed verbs based on their subject and object pairs [Lin and Pantel, 2001b], we

[1] Moby Thesaurus: http://moby-thesaurus.org

[2] We use only synonymy and antonymy relation from WordNet, not the synsets.

[3] http://www.dwijaya.org/dvkb.html#DKVB

[4] On average, a relation in VerbKB has 3.7 verb patterns as members and they have the same type signature, which is also the type of the relation.

[5] Moby Thesaurus: http://moby-thesaurus.org

also use as additional cues, the similarities of verbs' type signatures or their selectional preferences [Resnik, 1997, Light and Greiff, 2002]. Selectional preferences centered on both verb lexemes and prepositions have been shown to help classify verb lexemes' semantic roles, especially when there is limited syntactic information, which is certainly true in our case as we have limited syntactic information in our SVO triples [Mechura, 2008, Zapirain et al., 2013].

The motivation behind using type signatures as additional cues for clustering typed verbs is also the belief that when verbs have more than one alternative selectional preference, the selectional preference can help distinguish separate verb senses. For example, while the verb pattern "eat" has only one *typical* preference for an object of type *food*, the verb pattern "follow" has several alternative preferences for object types; including *path* (*path*, *route*, *trail*, *track*), instruction (*instructions*, *guidelines*, *recommendation*) and event (*publication*, *resignation*, *arrest*) [Mechura, 2008]. The type signatures or selectional preferences of the verbs can help distinguish the separate verb senses. For example, the verb pattern "follow" has the sense "happen after" when its object is of type *event* and "move along" when its object is of type *path*. Indeed, type signatures of verbs or their selectional preferences (i.e., their tendency to co-occur with subjects and objects from certain semantic classes), have been shown to improve the performance of automatic verb classification [Sun and Korhonen, 2009].

To type the subjects and objects of verbs in our SVO triples, we harness the rich type system and entity population in a large NELL knowledge base to map the noun phrases subjects/objects to their semantic types (see section 1.4 for our design choices with regards to typing the subjects and objects in the SVO triples).

In addition, we differ from other automatically constructed resources in that we also leverage pre-existing hand-crafted lexical resources that are the Moby thesaurus and WordNet and use relational information (synonymy, antonymy between verbs) from these semantic lexicons as constraints to encourage synonym (and respectively discourage antonym) verbs to be in the same clusters.

5.3 Related Work

There have been a number of automatically constructed lexical resources which contain verbs and organize them into semantically meaningful groups. For example, DIRT [Lin and Pantel, 2001b], PPDB [Ganitkevitch et al., 2013], Polysemy-Aware Verb Classes [Kawahara et al., 2014], PATTY [Nakashole et al., 2012] and ReVerb [Fader et al., 2011]. We have discussed the details of these lexical resources in Chapter 2. Here, we present summaries of these resources and the comparisons to VerbKB.

DIRT (Discovery of Inference Rules from Text) [Lin and Pantel, 2001b] is a collection of paraphrases automatically learned from corpora. The approach is motivated by the hypothesis that if two phrases tend to link the same sets of nouns then the meanings of the corresponding phrases are similar. The algorithm behind our verb resource, VerbKB, computes these DIRT-style pairwise similarities for typed verbs based on the sets of subject and object nouns that the typed verbs occur with in the SVO triples. We compute similarities for typed verbs with the same type signatures, which we call *local* similarities. We also compute similarities for ***all*** verbs *without* their types, which we call *global* similarities. However, our algorithm differs from DIRT in that it also uses the verbs' synonymy and antonymy relations to constrain these computed similarities.

Instead of extracting paraphrases from monolingual corpora like DIRT, the Paraphrase Database (PPDB) [Ganitkevitch et al., 2013] extracts paraphrases from large bilingual corpora. The most recent release of PPDB [Cocos and Callison-Burch, 2016] includes the grouping of each phrase' paraphrases into separate sense clusters (synsets). Unlike PPDB synsets, however, we do not have access to similarities computed from bilingual corpora. Instead, we leverage pre-existing linguistic resources to obtain synonymy and antonymy relations between verbs. We show in the experiments, that the resulting verb synsets in VerbKB align better than PPDB synsets to manually constructed synsets.

Kawahara et al. [2014] produce a collection of polysemy-aware verb classes from verb lexeme uses in GigaWord (LDC2011T07; English Gigaword Fifth Edition) and web corpora. They deal with verb polysemy by first clustering each verb lexeme's uses into its verb-specific semantic frames and then clustering over these frames. The constructed lexical resource has 1.6k verb lexemes over 840 classes and takes up to three days to construct. In contrast, our algorithm took only a total of 7 hours without parallel processing to construct clusters in VerbKB. Also, polysemy is dealt naturally in VerbKB by allowing each verb pattern to have several type signatures that can help to distinguish the different verb senses (e.g., the verb pattern "play" can have types (*musician, musicInstrument*), (*athlete, sport*), etc.)

PATTY [Nakashole et al., 2012] is a large-scale collection of synonym sets of relational patterns harvested from text corpora. PATTY differs from DIRT, PPDB, and Kawahara et al. [2014] in that each of its patterns has a type signature for the entities that they connect (e.g., for the pattern "first performed in", it has the type signature [person × country]. Learning from PATTY and the work of Sun and Korhonen [2009] that shows that type signatures improve verb classification, in building VerbKB we cluster typed verbs.

PATTY, however, has low coverage in terms of typed verbs. The average size of the clusters is only 1.19, which means a lot of its clusters consist of variations of a single typed verb. We believe that in terms of typed verbs, PATTY clusters do not provide a lot of generalization. In contrast, VerbKB has on average 3.7 typed verbs per cluster.

Although PATTY clusters are small in terms of typed verbs, its verb clusters are not singletons as that of ReVerb [Fader et al., 2011]. ReVerb is a lexical semantic resource that is purely textual, where each pattern is a cluster/relation by itself. Like other Open IE systems and unlike VerbKB, due to its open-domain and open-relation nature, ReVerb is purely textual and is unable to relate the surface forms to an ontology of a knowledge base, if known in advance [Soderland et al., 2010].

5.4 Data Construction

We start from over 650 million subject-verb-object (SVO) triples extracted from ClueWeb. We use triples that occur more than once in ClueWeb and extract the verb pattern that is, the lemmatized verb phrase that occurs between the subject and object in each triple and that matches our part-of-speech based regular expression: V | VP (section 1.4).

A verb pattern may have different senses depending on its type signature i.e., the types of its subject and object. For example, the verb pattern "play" with the type signature (*musician*, *musicInstrument*) has a different sense than the same verb pattern "play" with a different type signature (*athlete*, *sport*). We obtain type signatures for the verbs and cluster the typed verbs instead.

As described in section 1.4, we use a list of NELL's category labels for millions of noun phrases and NELL's category labels for noun phrases that are in NELL KB to type the subject and object noun phrases in the SVO triples.

Different from the mapping from typed verbs to relations in Chapter 3 where we use *only* SVO triples whose subjects and objects are entities in NELL, to cluster as many typed verbs as possible in the SVO triples here, we use *all* SVO triples whose subjects and objects can be labeled with categories in NELL. This design decision results in more typing errors than what we encounter in Chapter 3 because on top of the errors that arise from a subject/object being typed with a wrong category by NELL (see how we deal with that in section 1.4), the typing of subjects and objects here is done out of context i.e., independent of the verbs that occur with the subjects or the objects. As a result, a subject/object may be typed with more than one of the NELL categories, not all of the categories are correct with regards to the verb pattern. For example, the noun phrase "apple" is labeled with the NELL categories *fruit* and *company* which are both correct. In the triple "John ate an apple" however, labeling the object "apple" with the category *company* will result in an *incorrect* type signature for "eat"[6].

To deal with this second source of typing errors, we compute *acceptable* type signatures for each verb pattern[7] from this large-scale, erroneous types by conducting these steps:

1. We filter out type signatures that are uninformative for the verb pattern based on their frequencies of co-occurrence with the verb pattern in the SVO triples (section 5.4.1).
2. From the remaining type signatures, we recursively expand the types with their parent types – based on the hierarchy of types in NELL – to generate all possible candidate signatures for the verb pattern (section 5.4.2) .
3. We compute the verb pattern's selectional preference – its tendency to co-occur with subjects and objects from certain semantic classes – to rank and to automatically threshold candidate signatures for the verb pattern based on the selectional association scores they have with the verb pattern (section 5.4.3).

We detail each of these steps in the following sections.

5.4.1 Filtering out *uninformative* type signatures

For each verb pattern, given the category labels of its subject and object in SVO triples, we filter out subject/object labels that are uninformative for the verb pattern based on their frequencies of co-occurrence with the verb pattern.

First, labels with low entropy[8] are filtered out. The entropy is computed for each label and each verb pattern in terms of the noun phrases (NPs) the label co-occurs with in the verb pattern's SVO triples.

[6] "eat a company" may be true in some metaphorical use of the verb pattern "eat".

[7] "acceptable" as in typical use of the verb pattern

[8] $<= 0.5$

For example, the SVO triple: "Jesus *died on* the cross" is a frequent triple for the verb pattern "die on". However, NELL labels the NP: "the cross" to be of type *location* or *physicalCharacteristic*. Obviously *physicalCharacteristic* is not a good object type for the verb pattern "die on" i.e., people do not typically "die on" a *physicalCharacteristic*. Furthermore, the label *physicalCharacteristic* only occurs with the verb pattern "die on" when its object is the NP: "the cross". Therefore, *physicalCharacteristic* as an object for the verb pattern "die on" has a low entropy and can be filtered out from the list of typical objects of the verb pattern.

Secondly, we filter out category labels which are mutually inclusive[9] with other labels that occur more frequently with the verb pattern.

For example, the verb pattern "die on" can occur with object NPs of type *date* e.g., "Monday", "Tuesday", "March 13", etc. However, some of the *date* NPs such as "Monday", "Tuesday", and "Friday" are also labeled by NELL with the category *televisionShow*. Since the category *televisionShow* is mutually inclusive with the category *date* – they share a lot of NPs in common as objects of the verb pattern "die on" – and since the category *date* occurs more frequently than *televisionShow* with the verb pattern "die on", the category *televisionShow* is filtered out.

Thirdly, we filter out categories in NELL that we deem are too general or uninformative for verb patterns such as the category *everyPromotedThing*, etc. These are categories that are at the root of NELL's ontology of types.

5.4.2 Generating *all* candidate type signatures

After we filter out labels that are uninformative for the verb pattern's subject or object, we generate all possible type signatures for the verb pattern from its remaining subject and object labels.

Since the NELL categories are organized into a hierarchy, depending on which level in the hierarchy we consider, we may obtain different type signatures for the same verb pattern. For example, for the verb pattern "die on" with type signature (*personEurope*, *dayOfTheWeek*), we can use the combination of parent categories of *personEurope*[10] and *dayOfTheWeek*[11] to recursively generate more type signatures for the verb pattern: (*personByLocation*, *dayOfTheWeek*), (*person*, *dayOfTheWeek*), and (*personEurope*, *date*), (*personByLocation*, *date*), and (*person*, *date*).

Using the NELL hierarchy of types, we thus generate all possible type signatures for the verb pattern and compute their frequencies of occurrence with the verb pattern in the SVO triples. We use LASH [Beedkar and Gemulla, 2015], which is a scalable algorithm for mining patterns in the presence of hierarchies, to mine type signatures of verb patterns in the SVO triples and compute their frequencies of occurrences in the SVO triples. LASH can generalize each category in a mined signature to its higher category in the NELL hierarchy and generate their combinations. For example, given a mined signature (*personByLocation*, *dayOfTheWeek*) for the verb pattern "die on", LASH will generate type signatures (*person*, *dayOfTheWeek*), (*personByLocation*, *date*) and (*person*, *date*) for the verb pattern. Among the generated signatures, we select those that occurs at least twice with the verb pattern in the SVO as candidate signatures for the verb pattern.

[9] Have high *Jaccard* similarity ($>$ 0.6) based on noun phrases they co-occur with in the verb pattern's SVO triples

[10] *personEurope* → *personByLocation* → *person*

[11] *dayOfTheWeek* → *date*

5.4.3 Computing selectional preference

Using frequencies computed by LASH, we compute Resnik's selectional association of a candidate type signature (t_s, t_o) and the verb pattern v to compute a score A_R for the candidate type signature [Resnik, 1997]:

$$A_r(t_s, v, t_o) = \frac{1}{S_R(v)} P(t_s, t_o|v)\ log\ \frac{P(t_s, t_o|v)}{P(t_s, t_o)}$$

where $S_R(v)$ is the selectional preference of the verb pattern that uses the KL divergence to express how much information the verb pattern expresses about the possible semantic class of its argument [Jurafsky and Martin, 2014].

$$S_R(v) = \sum_{t_s, t_o} P(t_s, t_o|v)\ log\ \frac{P(t_s, t_o|v)}{P(t_s, t_o)}$$

The score of a type signature for a verb pattern is, therefore, the strength of association between the verb pattern and the type. For each verb pattern, we rank its candidate type signatures based on these selectional association scores. We use the method in [Satopaa et al., 2011] to automatically threshold the scores based on finding the knee in the sorted scores curve, where saturation occurs and the difference between any two scores are negligible. We select signatures whose scores are above this threshold.

Some examples of the resulting type signatures (and their frequencies of occurrence with the verb pattern (in percentages)) can be seen in Table 5.1[12]. For example, some selected type signatures for the verb pattern "die at" are (*person, location*) e.g., "Michael Jackson died at his house" and (*person, nonNegativeInteger*) e.g., "Michael Jackson died at 50".

For each selected type signature for a verb pattern, we also select all its child signatures (as defined by the NELL hierarchy) that occur with the verb pattern as acceptable type signatures for the verb pattern.

As we can see in Table 5.1 however, the method is not perfect. There may still be some noisy type signatures selected for a verb pattern. For example, the type (*location, date*) for the verb pattern "die in". Upon closer examination, we find that some of these incorrectly selected types are due to the fact that we type the verb pattern's subject and object out of context and the fact that some of the subjects or objects have ambiguous types.

For example, a lot of proper names in NELL ("Edleston", "Helen", "Weber", etc.) are labeled with either the category *person* or the category *city*, whose parent category is *location*. Since we type subjects and objects in SVO triples out of context i.e., we do not take the verbs in the triples into consideration when typing their subjects and objects; the proper name: "Davis" in the triple: "Davis *died in* October 1" for example, are labeled by NELL with the category *city* and therefore *location*. This is how (*location, date*) ends up being one of the top ranked signatures for the verb pattern "die in", albeit with a much lower frequency than the highest ranked type: (*person, date*). In future, we can envision a method for typing the subjects and objects in the SVO triples

[12] Note that for each verb pattern, these percentages do not sum to 100 because we are not using *all* type signatures that occur with the verb pattern in the SVO. We compute their selectional association scores and only select signatures whose scores are above certain automatically computed threshold

Verb Pattern	**Type signatures**
play	(*person*, *game*) (6.3%), (*person*, *person*) (3%), (*person*, *musicInstrument*) (2.6%), ...
die in	(*person*, *date*) (14.2%), (*person*, *location*) (7.3%), (*location*, *date*) (1.1%), ...
die of	(*person*, *physiologicalCondition*) (28%), (*location*, *physiologicalCondition*) (1.3%), ...
die at	(*person*, *location*) (12.7%), (*person*, *nonNegInteger*) (4%), ...
sit on	(*person*, *householdItem*) (7.1%), (*location*, *location*) (3.8%), (*person*, *bodypart*) (2%), ...
nibble	(*animal*, *food*) (5.7%), (*animal*, *bodypart*) (3.9%), (*person*, *food*) (3.7%), ...

Table 5.1: Automatically discovered type signatures for verbs sorted by their percentage of occurrences with the verbs in the SVO triples

in the context of the verb pattern and incrementally, using semi-supervised methods such as Expectation Maximization to use some initial type signatures of the verb pattern to type even more of its SVO triples' subjects and objects in the context of the verb pattern and use the new in-context types to reweigh the scores of the current types.

Once we obtain the type signatures of each verb pattern, we treat each verb pattern-type signature pair or simply, the typed verb as a data point in our clustering. For example, "marry"(*person*, *person*), "die in"(*person*, *location*) are typed verbs that are data points in our clustering. In the following sections, we compute similarities among these data points and cluster them.

5.5 Similarity Computation

After obtaining the type signatures for each verb pattern, we treat the resulting typed verbs as data points in our clustering. To cluster the typed verbs, we first compute their pairwise similarities (section 5.5.1). We compute pairwise similarities between typed verbs that have the same type signatures and call them *local* similarities, and between all verbs i.e., without types and call them *global* similarities.

Since local similarities are computed only among typed verbs with the same type signature, overlaps of their subject and object pairs may be sparser than if we use all the verbs, resulting in sparse pairwise similarities or erroneous similarities that happen by chance. We select/add/remove similarities in the local similarities by using synonymy and antonymy relations among verbs that we obtain from Moby thesaurus and WordNet. To improve coverage of the synonymy relations, we use global similarities between verbs as an additional information of synonymy. We also use global similarities to supervise local similarities, to ensure that selected local similarities are not those that happen by chance (section 5.5.2).

We detail these steps for (1) computing pairwise relations and then (2) constraining them in the following sections.

5.5.1 Computing pairwise similarities

We compute similarities between typed verbs with the same type signatures, which we call *local* similarities. We also compute similarities between verbs, which we call *global* similarities. We use the method in DIRT [Lin and Pantel, 2001a] to compute these pairwise similarities based on the verbs' subject and object pairs co-occurrence in the SVO triples. We consider only subject-object pairs that occur with more than one verb pattern. Following Lin and Pantel [2001a], we first compute the mutual information between the verb pattern and its subject and object pair as:

$$I(v,(s,o)) = log\,\frac{|s,v,o| \times |*,*,*|}{|s,*,o| \times |*,v,*|}$$

where $|s,v,o|$ is the frequency count of the subject-verb-object triple in the collection of triples that are being considered for the similarity computation (i.e., triples with similar typed signatures for local similarity computation and all SVO triples for global similarity computation). Similarly, $|s,*,o| = \sum_v |s,v,o|$, $|*,v,*| = \sum_{s,o} |s,v,o|$, and $|*,*,*| = \sum_{s,v,o} |s,v,o|$.

Letting $T(v)$ be the set of pairs (s,o) such that $I(v,(s,o))$ is positive, DIRT then defines similarity (>=0) between a pair of typed verbs (or verbs for the global similarity computation) as:

$$sim(v_1,v_2) = \frac{\sum_{(s,o)\in T(v_1)\cap T(v_2)} \big(I(v_1,(s,o))\ +\ I(v_2,(s,o))\big)}{\sum_{(s,o)\in T(v_1)} I(v_1,(s,o))\ +\ \sum_{(s,o)\in T(v_2)} I(v_2,(s,o))}$$

which is based on an extended Harris' distributional hypothesis that states that words that occur in the same contexts tend to be similar. In our case, two verbs have high similarity if they have a large number of common subject-object pairs. However, not all subject-object pair is equally important. For example, the pair (*man*, *woman*) is much more frequent than the pair (*Bruce Jenner*, *Caitlyn Jenner*). Two verbs sharing the pair (*man*, *woman*) is less indicative of their similarity than if they shared the pair (*Bruce Jenner*, *Caitlyn Jenner*). DIRT similarity measure takes this into account by computing the mutual information between the verb and the subject-object pair i.e., $I(v,(s,o))$.

Using these pairwise similarities, we construct similarity graphs where the nodes are verbs and the edges are non-zero similarities between verbs. For each verb node, we keep only the top 50 most similar verbs to the verb node as its neighbors in the similarity graphs.

We observe that a verb pattern's neighbors in its global similarity graph consist of verbs that are synonymous to the verb pattern but may express diverse senses. On the other hand, the verb pattern's neighbors in its local similarity graph consist of verbs that are synonymous to the verb pattern and express the same sense. For example, the global neighbors of the verb pattern "follow" include verbs such as "pursue", "adopt", "understand" and "watch", which are synonyms of the verb pattern "follow" but express diverse senses. On the other hand, the local neighbors of the typed verb "follow"(*location*, *location*) include typed verbs such as "continue along"(*location*, *location*), "run along"(*location*, *location*) and "continue on"(*location*, *location*) that express the "move along" sense of the verb pattern "follow". Similarly, the local neighbors of the typed verb "follow"(*event*, *eventOutcome*) include typed verbs such as "happen after"(*event*, *eventOutcome*), "erupt upon"(*event*, *eventOutcome*), "break after"(*event*, *eventOutcome*) that express the "happen after" sense of the verb pattern "follow". Our observation of the differences between the global and local neighbors of verb patterns shows support of the hypothesis that type signatures can help disambiguate different senses of the verb pattern.

5.5.2 Constraining local similarities

For each type signature, we have a *local* similarity graph that contains similarities among typed verbs with the same type signature. In this graph, since similarities are computed only among typed verbs with the same type signature, the subject and object pairs co-occurrence may be sparse among these verbs, resulting in a sparse local similarity graph or erroneous edges that only happen by chance. We use synonymy and antonymy relations among verbs to select/add/remove edges from this similarity graph.

Specifically, we *remove* edges in the local similarity graph between verbs that are antonyms[13] and verbs that are *not* respective local neighbors (RLNs) – two verbs are a pair of RLNs if *each* is among the other's neighbors in the *local* similarity graph.

Among the remaining edges, we *select* edges between verbs that are synonyms[14] or verbs that are respective global neighbors (RGNs) – two verbs are a pair of RGNs if *each* is among the other's neighbors in the *global* similarity graph. If there is no existing edge between a pair of synonymous verbs in the local similarity graph, we *add* an edge between them if *either* is in the other's global neighbors. The rationale behind this is that we are using global similarities as a proxy for synonymy relations to improve the coverage of synonymy relations.

We also *select* edges between verbs that are respective nearest neighbors (RNNs) in the local similarity graph – two verbs are a pair of RNNs if each is the other's most similar verb pattern. This is based on the observation in [Lin and Pantel, 2001a] that words that are RNNs are semantically most meaningful. To make sure that the local RNNs are not by chance, we only select an RNNs edge if *either* of the verbs is in the other's global neighbors. The rationale behind this is that we are using global similarities to supervise local similarities, to ensure that selected local similarities are not those that happen by chance.

Remaining edges that are not selected or newly added are discarded from the graph. Then, we normalize the edges' similarities in the graph to between 0 and 1[15].

For each type signature in our data set, we obtain a constrained and normalized local similarity graph. Then we cluster the verbs in each of these graphs separately. We detail our clustering approach in the following section.

[13] Two verbs are also considered antonyms if they have the same verb but antonymic preposition e.g., "vote for" and "vote against" are considered antonyms. Two verbs are also antonyms if the verb of one is an antonym for the other's and they have the same preposition e.g., "buy from" and "sell from" are considered antonyms. This is how we extend the antonymy relations between verb lexemes in Moby Thesaurus and WordNet to verb patterns in VerbKB.

[14] Two verbs are also considered synonyms if the verb of one is a synonym for the other's and they have the same preposition e.g., "buy from" and "purchase from" are considered synonyms. This is how we extend the synonymy relations between verb lexemes in Moby Thesaurus and WordNet to verb patterns in VerbKB.

[15] Synonymous verbs are given edges of weight 1 between them.

5.6 Clustering

Once we obtain the filtered and normalized similarity graph for each type signature, we cluster verbs in each graph separately to produce clusters of typed verbs with a specific type (section 5.6.1).

However, we believe that clusters of types that are lower in the hierarchy of type signatures[16] can inform the clustering of typed verbs that are higher in the hierarchy.

For example, given the type (*person*, *person*), we will find that there is a large number of verbs that can occur between subject and object type pair (*person*, *person*). For example, a *person* can "marry" another *person*, "vote for" the other *person*, or "play with" the other *person*. These verbs, in turn, may share a lot of subject-object pairs in common even though they are not semantically similar. However, going down the hierarchy of type signatures to the level of (*politician*, *politician*) – a child of the type signature (*person*, *person*) – we will observe fewer number of verbs that can occur between the subjects and the objects of the type (*politician*, *politician*) e.g., "vote for", "elect", "nominate", "vote against", "campaign with" etc., and these verbs are more semantically similar. As a concrete example, we observe that when we cluster verbs of type (*person*, *person*), the typed verb "stump for"(*person*, *person*) which means "to make a speech in support of" is clustered with the typed verbs "lambaste"(*person*, *person*) and "stomp for"(*person*, *person*), which have opposite meanings to the former. In contrast, when we go down the hierarchy and cluster verbs of type (*politician*, *politician*), we observe that the typed verb "stump for"(*politician*, *politician*) is clustered with the typed verbs "endorse"(*politician*, *politician*), "back"(*politician*, *politician*), "campaign for"(*politician*, *politician*), which have more similar meanings to the former.

Going down the hierarchy can also help disambiguate verb clusters that belong to the different children of (*person*, *person*) such as (*politician*, *politician*) or (*athlete*, *athlete*). For example, an *athlete* is more likely to "play with" another *athlete* while a *politician* is more likely to "campaign with" another *politician*.

Hence, we cluster typed verbs starting from type signatures that are at the bottom of the constructed hierarchy. We then propagate the discovered clusters up the hierarchy (section 5.6.2).

5.6.1 Clustering algorithm

Given a similarity graph, we cluster the verbs in the graph using Markov clustering (MCL) [Van Dongen, 2001], a clustering algorithm for graphs that is based on the simulation of random walks in graphs. The MCL algorithm simulates random walks within a graph by alternating two operators called expansion and inflation. Expansion coincides with computing random walks with many steps. Since longer length walks are more common within clusters than between different clusters, the probabilities of walks associated with node pairs lying in the same cluster will, in general, be relatively large as there are many ways of going from one to the other. Inflation will then have the effect of boosting the probabilities of these intra-cluster walks and will demote inter-cluster walks. Eventually, iterating expansion and inflation results in the separation

[16] Type signatures are arranged from the bottom-up into a hierarchy based on NELL's type hierarchy e.g., (*person*, *dayOfTheWeek*) is a child of (*person*, *date*) since *dayOfTheWeek* is a child of *date* in NELL's hierarchy.

of the graph into different segments. There are no longer any walks between these segments and the collection of segments is interpreted as a clustering.

We choose MCL due to its fast and scalable nature since we have a large number of verbs in dense similarity graphs to cluster. Furthermore, MCL automatically decides on the number of clusters, which is a desirable feature since we do not know in advance how many verb clusters we should have for each type signature.

5.6.2 Bottom-up cluster propagation

We construct the hierarchy of type signatures based on NELL's type hierarchy e.g., the type signature (*politician*, *politician*) is a child of (*person*, *person*) since *politician* is a child of *person* in NELL's hierarchy.

Given the hierarchy, we start by clustering typed verbs whose type signatures are at the bottom of the hierarchy (i.e., the leaf type signatures).

Then, for each non-leaf signature, we use clusters of its children to inform its clustering. The idea is to treat *prevalent* clusters among its children as data points in its clustering. To define prevalent clusters, we generate (non-singleton) subsets of the children clusters and compute their frequencies among the children clusters. We select subsets that are *supported* by at least two children (i.e., subsets that occur in at least two children clusters) and compute the *average support score* for each subset p as:

$$score(p) = \frac{1}{|p|} \sum_{v \in p} \frac{f(p)}{f(v)}$$

where v is a verb pattern member of the subset p, $f(p)$ is the frequency of the subset (i.e., the number of children clusters that contain the subset) and $f(v)$ is the number of children clusters that contain the verb pattern v. In this definition, a subset that is supported by *all* the children will have a score of 1.

We further select only subsets that have scores $>$ 0.5 (i.e., "supported" by at least half of the children) and rank the selected subsets first by frequency then by length. We discard lower ranked subsets when *any* of its members is already contained in higher ranked subsets. The remaining subsets are then treated as data points in the clustering.

For example, assume that the type signature (*person*, *person*) has two child signatures: (*politician*, *politician*) and (*personByLocation*, *politician*) and that we are going to extract prevalent clusters for the type (*person*, *person*) among its children.

Given for example that the clusters of the type (*politician*, *politician*) are {"elect", "select", "vote for"} and {"slam", "criticize"}; and the clusters of the type (*personByLocation*, *politician*) are {"elect", "vote for"} and {"slam", "criticize", "roast"}; the non-singleton subsets generated from these children clusters (and their frequencies) are: {"elect", "select"} (1), {"elect", "vote for"} (2), {"select", "vote for"} (1), {"elect", "select", "vote for"} (1), {"slam", "criticize"} (2), {"slam", "roast"} (1), {"criticize", "roast"} (1), and {"slam", "criticize", "roast"} (1).

Out of these subsets, we select those that are supported by at least two children, namely {"elect", "vote for"} and {"slam", "criticize"}.

Then, we compute the average support score for each of the selected subsets. The score for $\{$“elect”, “vote for”$\}$ is $\frac{1}{2}(\frac{2}{2}+\frac{2}{2}) = 1$. The score for $\{$“slam”, “criticize”$\}$ is similarly 1 because these two subsets are supported by all the children of type (*person*, *person*).

The two subsets $\{$“elect”, “vote for”$\}$ and $\{$“slam”, “criticize”$\}$ are thus prevalent clusters for the type (*person*, *person*) and are treated as atomic data points in its clustering.

For example, if verbs that are of type (*person*, *person*) are “elect”, “vote for”, “select”, “slam”, “criticize”, “roast”, “marry”, and “wed”; the clustering will be over these points: $\{$“elect”, “vote for”$\}$, $\{$“slam”, “criticize”$\}$, “select”, “roast”, “marry”, and “wed”. In other words, “elect” will always be in the same cluster as “vote for” and “slam” will always be in the same cluster as “criticize” in the resulting clusters for type (*person*, *person*).

Thus, for each non-leaf type signature, we cluster over the “new” data points (which are *prevalent* cluster subsets among its children) and the “old” data points (verbs not yet contained in the “new” data points).

We compute similarities between these data points d in the manner of average linkage clustering, where similarities between any two points are the average similarities of their members:

$$sim(d_1, d_2) = \frac{1}{|d_1||d_2|} \sum_{v_1 \in d_1} \sum_{v_2 \in d_2} sim(v_1, v_2)$$

Then, we use MCL to cluster the data points based on their similarities.

This “cluster-then-propagate” method that starts from the bottom type signatures repeats until the topmost type signature is reached. The resulting clusters are the verb synsets in VerbKB. Some example verb clusters from VerbKB and their proposed relation names – their subject types followed by the most frequent verbs of the clusters and their object types – are shown in Table 5.2. As can be seen in the table, the different senses of the verbs such as “have” or “play” are naturally separated into different verb clusters with different type signatures, highlighting the value of using the verbs’ type signatures or selectional preferences together with the more traditional feature for verb clustering that is the verbs’ lexical co-occurrence.

5.6.3 Extending Coverage

The “cluster-then-propagate” method works in clustering verbs which have sufficient co-occurrence data in the SVO triples. For other typed verbs that are not yet included in any of the resulting clusters in VerbKB (or VKB in short), we create a cluster for each of them by finding the most frequent WordNet verb sense of its neighbors in the similarity graph. The result is an extended VerbKB (or VKB full).

5.6.4 Mapping clusters to relations in NELL

To link the resulting verb clusters to the NELL knowledge base, each cluster in VerbKB is then either mapped to an existing NELL relation or added as a new relation in NELL. The mapping is based on the overlap between the typed verbs in the cluster and the typed verbs we have learned for NELL relations in Chapter 3.

Proposed Relation Name	Verbs
cityHaveAttraction	have, boast, feature, house, ...
personHave-PhysiologicalCondition	have, experience, get, suffer, survive, sustain, risk, endure, tolerate, ...
writerHaveEmotion	have, feel, know, understand, experience, ...
musicianPlayMusicInstrument	play, pick, strum, play_like, ...
actorPlayPerson	play, star_as, portray, play_as, return_as, ...
personPlayHobby	play, make, do, bet, suck_at, dabble_in, ...

Table 5.2: Some verb clusters proposed as new relations

More specifically, for each NELL relation, we return a ranked list of candidate clusters to map to the relation based on the number of typed verbs they have overlap with the NELL relation. Since there are only 298 NELL relations in our data set, we manually select among the top 5 candidate clusters for each NELL relation the cluster(s) that map to the relation. Other clusters in VerbKB that are not mapped to any NELL relation are then added as new relations in NELL.

5.7 Design Choices

To make our clustering scalable on our similarity graphs, we make several design choices that are specific to this chapter. First, to make the similarity graph less dense, we keep only the top 50 most similar verbs for each verb pattern as its neighbors in the similarity (local or global) graph (as we have mentioned in section 5.5.1) .

Secondly, for each *local similarity* graph – graph that contains similarities between typed verbs that have the same type signatures, we remove edges from the graph whose similarities are lower than 0.01 to reduce noise in the graph – pairwise similarities that only happen by chance.

Thirdly, we run Markov clustering (MCL) with the default configuration. However, as a part of the clustering process, we add the constraint that the cluster size should be smaller than or equal to 15. This is mainly for scalability reason (since we are propagating the clusters from the bottom-up) and also because our analysis of WordNet verb synsets shows that more than 98% of the verb synsets in WordNet have less than or equal to 15 verb lexeme members.

During the clustering, given the output of MCL, for each cluster that has $>$ 15 members, we apply agglomerative single-link clustering [Manning and Raghavan] on members of the cluster to re-cluster it, skipping over merges that will result in a cluster of size $>$ 15. We use the mean Silhouette Coefficient [Rousseeuw, 1987] that balances between optimal inter-cluster tightness and intra-cluster distance to choose the optimal stopping point of the agglomeration. The Silhouette Coefficient is calculated for each member v in the clustering as:

$$s(v) = \frac{b(v) - a(v)}{max\{a(v), b(v)\}}$$

where $a(v)$ is v's average intra-cluster distance: average distance from v to all other v' in the same cluster, and $b(v)$ is v's lowest average inter-cluster distance: lowest average distance from v to all other v' in a cluster that does not contain v (we define distance as $dist(v_1, v_2)$ = $1 - sim(v_1, v_2)$).

At each step in our agglomerative process, we compute the mean Silhouette Coefficient of the resulting clusters. The stopping point of the agglomeration is the point where the mean Silhouette Coefficient is highest.

5.8 Experiments

We evaluate the quality of our verb clusters by comparing them to the reference clusters that are coarser grained WordNet verb senses induced by the Oxford Dictionary of English (ODE) [Stevenson, 2010][17]. Previous investigations using the ODE [Navigli, 2006, Navigli et al., 2007] have shown that coarse-grained word senses induced by the ODE inventory address problems with WordNet fine-grained inventory and are useful for word sense disambiguation. Since we are interested only in the quality of our verb clusters, we use only the verb synsets in the reference clusters to compare to our clustering.

We evaluate the quality of our verb clusters using three standard metrics: the V-measure, paired F-measure and node-based F-measure.

5.8.1 Evaluation Metrics

V-measure evaluates the quality of a clustering solution against reference clusters in terms of clustering homogeneity and completeness [Rosenberg and Hirschberg, 2007]. Given N data elements that are partitioned into classes in the reference clusters denoted by $C = \{c_1, ..., c_{|C|}\}$ and the clustering solution over these N elements that is denoted by $K = \{k_1, ..., k_{|K|}\}$, we construct a contingency matrix $A = \{a_{ij}\}$ such that a_{ij} is the number of data elements that are members of the reference class c_i and are assigned by the clustering solution to cluster k_j. In order to satisfy homogeneity, a clustering solution must assign only those data points that are members of a single class to a single cluster. In a perfectly homogeneous clustering solution, the class distribution within each cluster will be skewed to a single class or zero entropy i.e., $H(C|K) = 0$. Normalizing this value by the maximum reduction in entropy the clustering information can provide i.e., $H(C)$ and adhering to the convention of 1 being desirable and 0 undesirable, homogeneity is defined as $h = 1 - \frac{H(C|K)}{H(C)}$ or $h = 1$ if $H(C, K) = 0$ where

$$H(C|K) = -\sum_{k=1}^{|K|}\sum_{c=1}^{|C|} \frac{a_{ck}}{N}\, log\, \frac{a_{ck}}{\sum_{c=1}^{|C|} a_{ck}}$$

$$H(C) = -\sum_{c=1}^{|C|} \frac{\sum_{k=1}^{|K|} a_{ck}}{N}\, log\, \frac{\sum_{k=1}^{|K|} a_{ck}}{N}$$

Completeness, on the other hand, is symmetrical to homogeneity. In order to satisfy the completeness criteria, a clustering solution must assign ***all*** of the data points that are members of a

[17] Available from http://lcl.uniroma1.it/coarse-grained-aw

single class to a single cluster. In a perfectly complete clustering solution, the distribution of cluster assignments within each class will be completely skewed to a single cluster or zero entropy i.e., $H(K|C) = 0$. Therefore, symmetric to the computation of homogeneity, completeness is defined as $c = 1 - \frac{H(K|C)}{H(K)}$ or $c = 1$ if $H(K, C) = 0$ where

$$H(K|C) = -\sum_{k=1}^{|K|}\sum_{c=1}^{|C|} \frac{a_{ck}}{N}\ log\ \frac{a_{ck}}{\sum_{k=1}^{|K|} a_{ck}}$$

$$H(K) = -\sum_{k=1}^{|K|} \frac{\sum_{c=1}^{|C|} a_{ck}}{N}\ log\ \frac{\sum_{c=1}^{|C|} a_{ck}}{N}$$

V-measure is then the harmonic mean of homogeneity h and completeness c i.e., V-measure $= \frac{2.h.c}{h+c}$

Paired F-score evaluates the quality of a clustering solution like a classification task [Manandhar et al., 2010]. It generates the set of all data pairs belonging to the same reference cluster $F(C)$ and the set of all data pairs belonging to the same cluster in the clustering solution $F(K)$. Precision, recall, and F-score are then calculated as $P_{pair} = \frac{F(K) \cap F(C)}{F(K)}$, $R_{pair} = \frac{F(K) \cap F(C)}{F(C)}$, and $F_{pair} = \frac{2.P_{pair}.R_{pair}}{P_{pair}+R_{pair}}$.

Node-based F-score evaluates the quality of a clustering solution using purity as a measure of precision and inverse purity as a measure of recall [Sun and Korhonen, 2009, Kawahara et al., 2014]. The idea of purity is that each cluster k_j is associated with its most prevalent reference class c_i. Therefore, we define our node-based precision as $P_{node} = \frac{1}{|K|}\sum_{j=1}^{|K|} max_i\ |k_j \cap c_i|$. The inverse purity is then used to measure our node-based recall as $R_{node} = \frac{1}{|C|}\sum_{i=1}^{|C|} max_j\ |k_j \cap c_i|$. The node-based F-score is then defined as the harmonic mean of the node-based precision and recall $F_{node} = \frac{2.P_{node}.R_{node}}{P_{node}+R_{node}}$.

5.8.2 Experimental Results

We evaluate the quality of the verb clusters in our VerbKB (VKB) and the full version of our VerbKB (VKB full, see section 5.6.3). We compare the quality of our verb clusters with verb clusters extracted from PATTY and PPDB synsets. Our evaluation is over all the non-trivial (i.e., non-singleton) clusters so as not to introduce a bias towards singletons.

Table 5.3 shows the statistics of the resources that we compare with[18]. As can be seen from Table 5.3 our knowledge base of verbs covers the most number of verbs in ODE (ODE has 9,279 unique verb patterns). In terms of cluster size, PPDB has the largest average cluster size (possibly due to the nature of PPDB, which is first and foremost a large repository of paraphrases) while

[18] The actual number of verbs and verb clusters in PPDB can be larger. This is the number of verbs and clusters in PPDB that contain ODE verbs and their paraphrases

	PATTY	**PPDB**	**VKB**	**VKB full**
Number of unique verb patterns	12,275	6,785	29,866	65,679
Number of verb clusters	187,464	67,101	58,359	215,106
Average cluster size	1.2	6.7	3.7	2.0
Coverage of ODE-verbs	0.23	0.46	0.51	0.66

Table 5.3: Statistics of verb clusters.

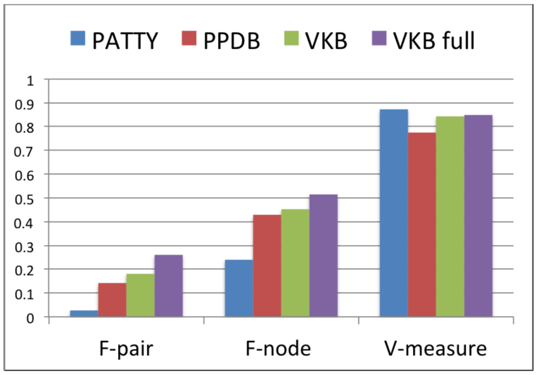


Figure 5.1: Clustering performance against verb clusters in ODE.

PATTY has the smallest. We observe that this translates to the performance of the clusters in terms of the alignment to the reference ODE verb clusters.

Figure 5.1 shows the performance of our verb clusters against PATTY and PPDB verb clusters in terms of the alignment to ODE verb clusters. In terms of V-measure PATTY performs the best. This maybe due to the fact that PATTY has many small clusters that are highly homogeneous. The V-measure is known to be biased towards a clustering solution that has many small clusters [Reichart and Rappoport, 2009]. However, in terms F-scores, PATTY that has many small clusters performs the worst because of low recall (as can be seen in Figure 5.2). On the other hand, PPDB that has large clusters on average performs better than PATTY in terms of F-scores (Figure 5.2). Although PPDB has a lower precision than PATTY, it has a much larger recall and an overall larger F-scores. Our knowledge base of verbs (VKB and VKB full) strike a balance between high V-measure and high F-scores (Figure 5.1). Qualitatively, methods that strike a balance between high V-measure and F-score tend to produce the 'best' clusters by human judgment [Cocos and Callison-Burch, 2016]. If we consider the average of F-score and V-measure as a comprehensive performance measure, our knowledge base of verbs (VKB and VKB full) outperform PATTY and PPDB.

To analyze the difference in performances of the different resources (PATTY, PPDB, and VKB) in terms of the actual clusters, we align each ODE cluster to a cluster that is most prevalent (i.e., has the highest node based F-score) to it in each of the resources. Closer examination of the best and the worst among these prevalent clusters in each of the resources highlights the difference in the nature of verb clusters in each of these resources. In Table 5.4, we show some of the best and the worst prevalent clusters in each of the resources that are representative of the general trend.

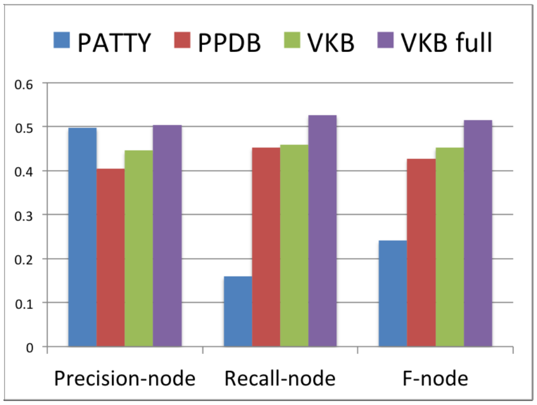


Figure 5.2: Clustering performance against verb clusters in ODE.

As seen in Table 5.4, we observe that in general, PATTY clusters that align the best to ODE clusters are those that consist of variations of a single verb e.g., {“work”, “work on”, “work at”}. Since PATTY clusters are generally small in size – the average size is 1.2 – and do not offer much in terms of generalization, some of these clusters align the worst to ODE clusters, especially those very large ODE clusters with multiple verbs. For example, in Table 5.4, PATTY cluster for the verb pattern “capture”: expresses a dominant sense of the verb pattern, which is “to defeat”. This cluster aligns badly to a large ODE cluster for the verb pattern “capture” that expresses a more obscure sense of the verb pattern, which is “to becharm” or “beguile”. Since PATTY clusters are smaller in size and mostly contain variations of a single verb, they are very precise but have low recall. This is what we have previously observed in Figure 5.1 and Figure 5.2.

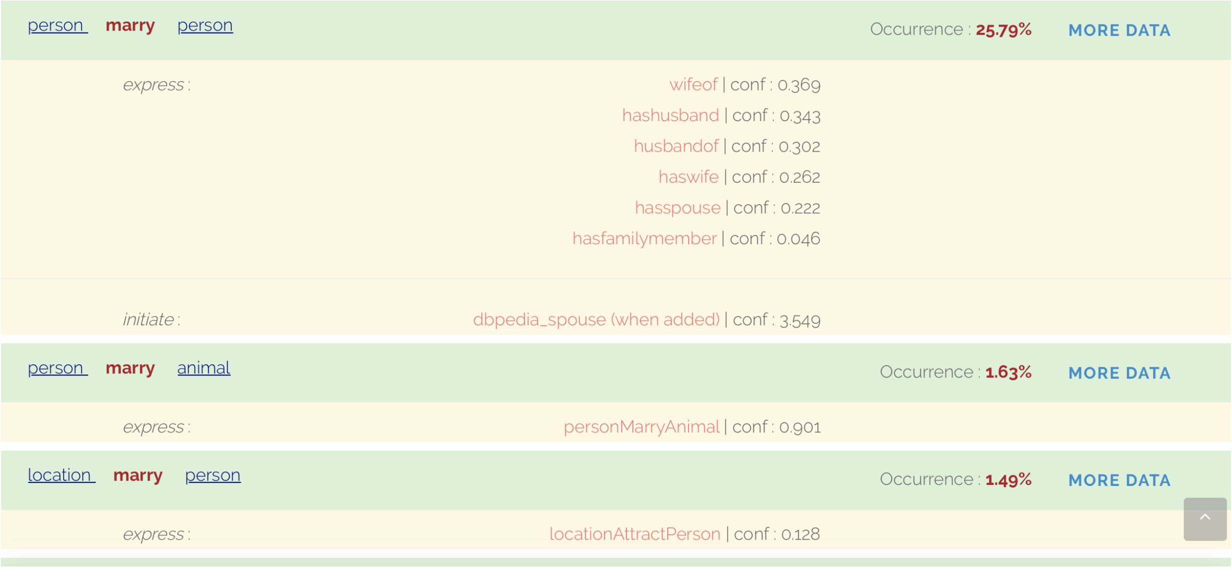
PPDB clusters, on the other hand, are large in size – the average size is 6.7 – and contain more verbs per cluster than PATTY clusters. As we can see in Table 5.4, PPDB clusters have no problem aligning to large ODE clusters with multiple verbs. However, PPDB clusters that are too large have problems aligning to small ODE clusters. Also, because PPDB clusters are large, they may contain a mixture of the different senses of the verbs. For example, as we can see in Table 5.4, PPDB cluster that aligns to ODE cluster of the verb pattern “rake” contains too many diverse verbs: “rake”, “accede”, “account”, “achieve”, “act”, etc. that it is hard to see which sense of the verb pattern it is trying to capture. Since PPDB clusters are larger in size and contain more diverse verbs, they have high recall but lower precision. This is what we have previously observed in Figure 5.1 and Figure 5.2.

VKB, on the other hand, strikes a balance between the precision and recall in its clusters. As we can see in Table 5.4, VKB clusters have no problem aligning to ODE clusters with multiple verbs. Furthermore, VKB clusters are not too large in size – the average size is 3.7 – and the clusters are constrained by the synonymy and antonymy relations among verbs. So even when VKB cluster does not align perfectly to the ODE cluster (as can be seen in Table 5.4), the cluster still appears reasonable and not too diverse in terms of the verb senses.

Other errors that we observe in VKB are those that we believe are caused by (among others): incorrect types in NELL, polysemous noun phrases, incorrect segmentations, the use of slang, or the use of metonymy. For example, when we observe type signatures that appear strange for

Resource	Reference (ODE) Cluster	Resource Cluster	F-score
PATTY	{“work”, “work at”, “work on”}	{“work”, “work at”, “work on”}	1.0
PPDB	{“aid”, “assist”, “facilitate”, “help”}	{“aid”, “assist”, “facilitate”, “help”}	1.0
VerbKB	{“build”, “construct”, “make”}	{“build”, “construct”, “make”}	1.0
PATTY	{“becharm”, “beguile”, “bewitch”, “captivate”, “capture”, “catch”, “charm”, “delight”, “enamor”, “enamour”, “enchant”, “enrapture”, ... (10 more verbs)}	{“capture”, “defeat”}	0.083
PPDB	{“rake”}	{“accede”, “account”, “achieve”, “rake”, “act”, “address”, “adopt”, “affect”, “aim”, “allocate”, “amount”, “answer”, “apply”, “arise”, “arrive”, “ask”, ... (more than 20 more verbs)}	0.0094
VerbKB	{“abominate”, “loathe”, “accurse”, “anathematise”, “anathematize”, “anathemise”, “anathemize”, “comminate”, “execrate”}	{“hate”, “insult”, “despise”, “loathe”, “smear”, “ridicule”, “disrespect”, “scream”}	0.118

Table 5.4: Clusters in each resource that are most prevalent to the reference (ODE) clusters



person marry person Occurrence : 25.79% MORE DATA

express : wifeof | conf : 0.369
hashusband | conf : 0.343
husbandof | conf : 0.302
haswife | conf : 0.262
hasspouse | conf : 0.222
hasfamilymember | conf : 0.046

initiate : dbpedia_spouse (when added) | conf : 3.549

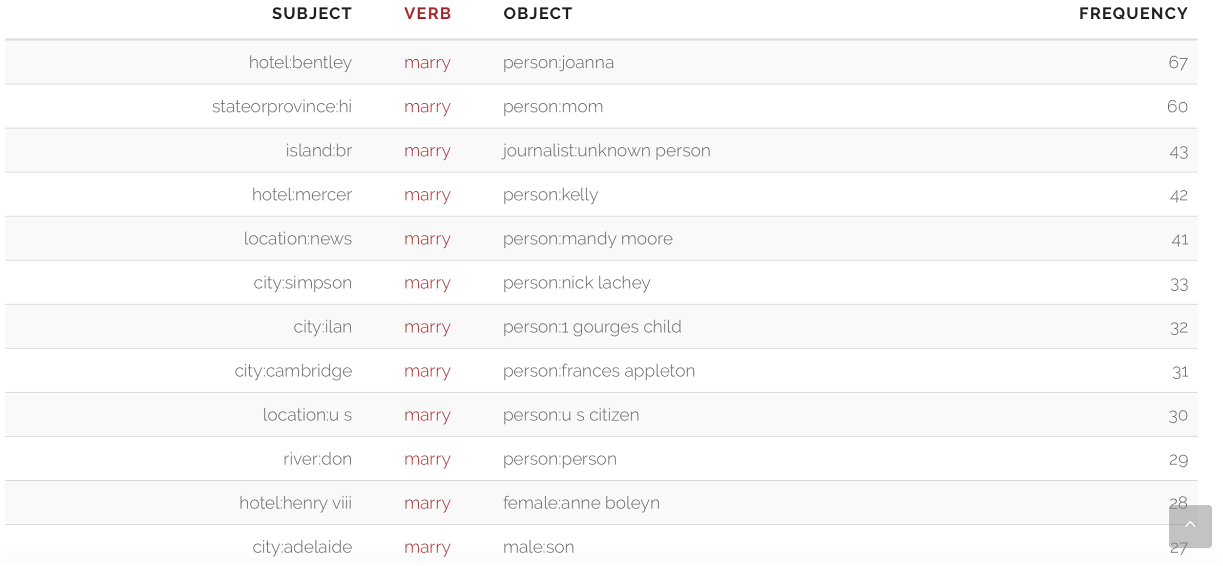
person marry animal Occurrence : 1.63% MORE DATA

express : personMarryAnimal | conf : 0.901

location marry person Occurrence : 1.49% MORE DATA

express : locationAttractPerson | conf : 0.128

Figure 5.3: The VerbKB entry for the verb pattern “marry”, showing its various type signatures sorted by their frequencies of occurrences in the SVO triples. As can be seen here, there are strange type signatures for “marry” such as (*person*, *animal*) and (*location*, *person*).



SUBJECT	VERB	OBJECT	FREQUENCY
hotel:bentley	marry	person:joanna	67
stateorprovince:hi	marry	person:mom	60
island:br	marry	journalist:unknown person	43
hotel:mercer	marry	person:kelly	42
location:news	marry	person:mandy moore	41
city:simpson	marry	person:nick lachey	33
city:ilan	marry	person:1 gourges child	32
city:cambridge	marry	person:frances appleton	31
location:u s	marry	person:u s citizen	30
river:don	marry	person:person	29
hotel:henry viii	marry	female:anne boleyn	28
city:adelaide	marry	male:son	27

Figure 5.4: Actual SVO triples for the typed verb "marry"(*location*, *person*). As shown here, in VerbKB website, besides browsing the mappings from the typed verb to the knowledge base relations, users can browse actual SVO triples that the typed verb has and their frequencies.

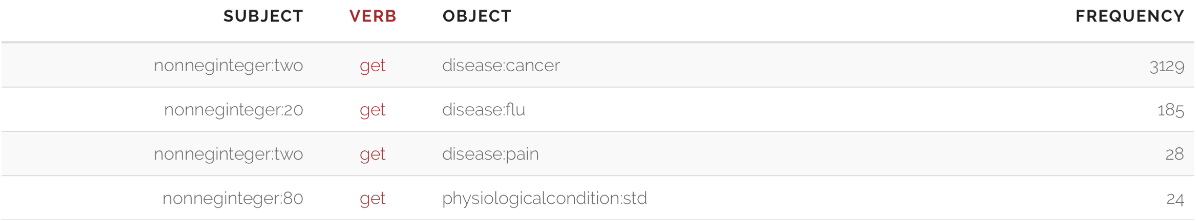
the verb patterns, e.g., the type signatures (*person*, *animal*) and (*location*, *person*) for the verb pattern "marry" (see Figure 5.3 that is taken from our VerbKB website[19]).

Looking at actual SVO triples for the typed verb "marry"(*person*, *animal*), we believe that this strange type is due to the frequent use of an animal slang pejorative in the context of marriage, e.g., "dog" to refer to a *person*.

Looking at actual SVO triples for "marry" that have the type signature (*location*, *person*) in Figure 5.4, which is taken from our VerbKB website, shows that some triples have been typed incorrectly by NELL. For example, NELL has typed incorrectly, the noun "br" as an *island* and the phrase "unknown person" as a *journalist*. Computing the selectional preference of the verbs helps in reducing some of these incorrect types but not all; especially if there are other factors that make (*location*, *person*) an acceptable type signature for the verb pattern "marry", for example, (1) polysemous noun phrases, e.g., "Henry VIII" can be both the name of a *hotel* or the name of a *person*. Without taking the context – in this case, the verb – into consideration, NELL can type "Henry VIII" as a *hotel*. (2) Metonymy, e.g., "US marry US citizen" does not mean that US (the *country* → *location*) is marrying its citizen (the *person*). Rather, "US" is probably used as a metonymy to refer to the US marriage officiant that can marry the citizen off.

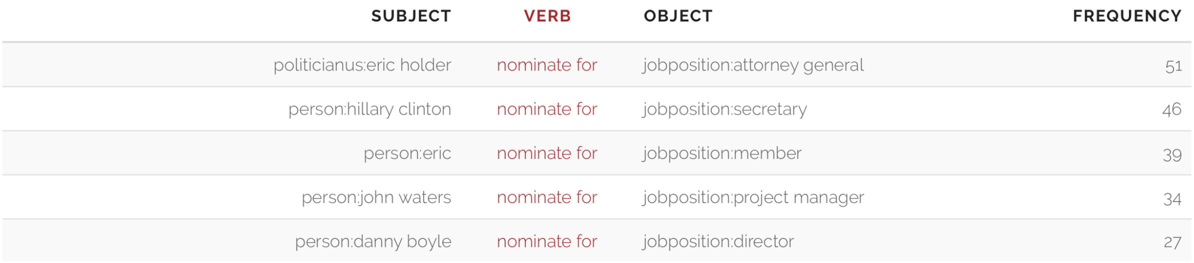
Other strange type signatures may be due to segmentation errors in the process of obtaining SVO triples. For example, the type (*nonNegInteger*, *physiologicalCondition*) for the verb pattern "get" (see Figure 5.5). All the SVO triples that have this type signature for the verb pattern "get" have their subjects incorrectly segmented. For example, in the SVO triple "two get cancer", the subject "two" is probably a part of a bigger noun phrase; the remainder of which is not extracted

[19] http://www.dwijaya.org/dvkb.html#DKVB



SUBJECT	VERB	OBJECT	FREQUENCY
nonneginteger:two	get	disease:cancer	3129
nonneginteger:20	get	disease:flu	185
nonneginteger:two	get	disease:pain	28
nonneginteger:80	get	physiologicalcondition:std	24

Figure 5.5: Actual SVO triples for the typed verb “get”(*nonNegInteger*, *physiologicalCondition*).



SUBJECT	VERB	OBJECT	FREQUENCY
politicianus:eric holder	nominate for	jobposition:attorney general	51
person:hillary clinton	nominate for	jobposition:secretary	46
person:eric	nominate for	jobposition:member	39
person:john waters	nominate for	jobposition:project manager	34
person:danny boyle	nominate for	jobposition:director	27

Figure 5.6: Actual SVO triples for the typed verb “nominate for”(*person*, *profession*).

into the triple for some reason (segmentation errors, parsing errors, etc.).

Segmentation errors can affect the extraction of the verb patterns too. For example, passive construction of verbs may not be extracted properly into the SVO triples, resulting on a cluster that contains the same verb with both active and passive voices that have different meanings, e.g., the ***personRunForProfession*** cluster that contains the verbs: “run for”, “stand for”, “nominate for”, “(passive) nominate for”, ... with the type signature (*person*, *profession*). Here, the typed verb “nominate for”(*person*, *profession*) has a different meaning than the typed verb “be nominated for”(*person*, *profession*). Looking at the actual SVO triples for the typed verb “nominate for”(*person*, *profession*) shows that the verb should be in a passive voice instead (see Figure 5.6). For example, the triple “Eric Holder nominate for Attorney General” does not mean that *Eric Holder* *nominates* someone to be an Attorney General – only the US president can nominate someone to be an Attorney General and *Eric Holder* is not the US president. Rather, it means that he (*Eric Holder*) *is nominated* for the Attorney General position. Since our clustering depends on the verb patterns having the correct types and subject-object pairs; noisy types and incorrect verb extraction can have negative impacts on the clustering result – as we have seen in this ***personRunForProfession*** cluster where active and passive verb voices with different meanings are put in the same cluster. These errors highlight some of the challenges that remain for future work.

5.9 Analysis and Discussion

There are many ways that we can improve on the verb clusters in VerbKB. One way to improve recall will be to add more verb paraphrases to the clusters using resources such as PPDB.

In terms of clustering and precision, we have clustered typed verbs into hard clusters, which means that one typed verb is only mapped to one cluster. In practice, this is not always correct. A verb pattern with the same type signature can mean different things. For example, the verb pattern "shoot" with the type signature (*person, person*) can mean either "kill with a gun" or "take a photograph" [Mechura, 2008]. Our approach does not allow this typed verb to belong to two different clusters; hence the two senses of the verb pattern "shoot" are put in the same cluster. One way to improve the precision of our clusters is to allow soft clustering of the typed verbs such that a verb with the same type signature can belong to more than one cluster.

In terms of type signatures, our decision to use type signatures as additional cues for clustering verbs is motivated by the belief that they can help distinguish verb senses. In the experiments, we have shown that this approach indeed results in a better alignment to the manually created verb synsets. This result is in line with the previous observation that shows how semantic preferences can improve verb classification especially in finer grained verb clusters, which is true in the case of our reference clusters which are fine grained [Sun and Korhonen, 2009]. Furthermore, an analysis of the reference clusters shows that among the clusters that align *well* with VKB clusters[20], a substantial number – 36% – of these clusters have strong selectional preferences i.e., the most frequent type signature of the verbs in the cluster matches the type signature of the VKB cluster that it aligns to[21]. For example, the cluster {"holiday in", "vacation in", ...} has the type signature (*person*, *location*); the cluster {"gaze with", "stare with", ...} has the type signature (*person*, *emotion*). This further confirms our belief that selectional preferences can indeed help in distinguishing verb senses.

Given this result, it will be interesting to further study which categories and selectional preferences are indeed useful for verb clustering. The answers to questions like (1) which type position (subject/object) results in the best clusters – in this work we use both subject *and* object types, (2) which types are semantically meaningful/plausible for which verb clusters, (3) which types are frequent for which verb clusters; can be useful for learning why and when type signatures matter, (4) what are the effects of imposing selectional preference to verbs that do not typically have selectional preferences or whose selectional preferences are too general like "be" or "make"? Does adding type signatures into these verbs affect their clustering results adversely?

In terms of the adequacy of semantic types, one way to increase the coverage of the noun phrases to semantic types mappings is to align types in NELL with types in WordNet, which has 30 times more types than NELL; and to add types in WordNet that are not yet in NELL to extend NELL's vocabulary of types. Currently, NELL can type *both* the subjects and objects of only about 94 million subject-verb-object triples out of the 650 million extracted from ClueWeb. This shows NELL's limited type coverage. For example, NELL does not have types for abstract entities like *idea*, *sound*, or *communication*, therefore, for a lot of these entities, NELL cannot type their SVO triples. NELL also does not have types that indicate properties like *animate*, *rigid*, *solid*, which are semantic types that are useful for classifying verbs in VerbNet. Furthermore, in some ways, NELL types are not fine enough – there are no finer types like *disability* under *nonDiseaseCondition* in NELL, or no finer types like *ink* under *officeItem* – but in some ways, NELL types are not coarse enough – there are no types like *organism* to be a super type of *animal*

[20] have node-based F-score of > 0.5

[21] Moreover, 66% (and 76%) of reference clusters that align well to VKB clusters have the type signatures of VKB clusters that they align to in the top-3 (and top-5 resp.) most frequent type signatures of their verbs.

or *instrumentality* to better classify *item* type based on its functionality. Adding WordNet types can improve this coverage. Unlike NELL, which is noisy and may contain incorrect semantic types for noun phrases, WordNet types are manually and precisely created. However, WordNet types are known to be too fine grained. The question then becomes how far down the hierarchy of WordNet types do we go. One possible direction is to try and learn this from the corpus data and analysis of the verb clusters. For example, if verbs in a particular cluster are divided over the subjects and/or objects that they co-occur with or if clustering their subjects and/or objects[22] results in disjoint clusters; then perhaps it is a good idea to split the category of the cluster's subjects and/or objects into finer categories (see some specific examples we have discussed in section 3.10).

In terms of types, a better representation of types in knowledge bases such as NELL may also be needed. In NELL, an entity belongs to one type and one type only. For example, the entity *Obama* (the *politician*) is different from the entity *Obama* (the *male*) and the entity *Obama* (the *personUS*) even though these are all the same entity in reality. In reality, the entity *Obama* can belong to all these categories: *male*, *politician* or *personUS* but to different degrees depending on context. The categorization of entities to types in knowledge bases should be more fluid taking selectional preferences of verbs into consideration. An example of an ontology that takes selectional preference into account for category typing is the Corpus Pattern Analysis (CPA) project[23] where lexical items are typed numerically and by context. For example, in CPA the noun phrase "meeting" is typed with category *event* to a degree of 0.12 when it is the object of the verb pattern "attend" and to a degree of 0.04 when it is the object of the verb pattern "hold" [Mechura, 2008]. The CPA ontology is populated manually. In contrast, our knowledge base of verbs can be used to produce a more fluid typing of entities in NELL automatically. By similar reasoning, the notion of selectional preferences should be made more fluid. When used in the context of criminal investigation, e.g. in a subframe of CRIMINAL PROCESS, the verb pattern "interrogate" should have a *prisoner* or *suspect* type as object. When used in a more general context, the verb pattern can have a *person* type as object.

In terms of similarity measures used in our approach, a better representation of verbs for computing similarities can be explored. One possibility is to learn embeddings of the typed verbs and then retrofit them based on the verbs' synonymy relations like in [Faruqui et al., 2014] or counter-fitting them based also on the verbs' antonymy relations like in [Mrkšić et al., 2016].

In terms of evaluation, in the future, we can certainly benefit from manual labeling evaluation. However, from our automatic evaluation, we observe that in our evaluation of PATTY, its V-measure specifically is comparable to the evaluation that they reported, which was manually conducted. In the future, we can extend our automatic evaluation by aligning to other verb resources such as FrameNet and VerbNet. Aside from the benefits of having alignments to knowledge in these resources, it will be interesting to use the alignments to examine differences between our verb groupings (that are based on constrained distributional similarities and type signatures) with groupings that are based on the verbs' semantic roles (FrameNet) or syntactic realizations (VerbNet).

[22] Using tools such as SenseMaker [Mechura, 2008] or other methods for acquiring selectional preference [Brockmann and Lapata, 2003]

[23] http://nlp.fi.muni.cz/projekty/cpa/

In terms of algorithm and design choices, our clustering approach is different from the algorithm that we use to map verbs to relations in Chapter 3. In Chapter 3, we classify entity pairs and use verbs as features. We obtain the mappings from verbs to relations as parameters of the classification. In contrast, in this chapter, we cluster verbs and use entity pairs as features. This decision was motivated by our goal of obtaining as much coverage of the verbs in the SVO triples as possible. In Chapter 3, when using verbs as features, we filter out verbs based on their *tf-idf* scores. Hence, we do not cover all the verbs in the SVO triples. In contrast, in this chapter, we cluster the verbs and filter out entity pairs that cannot be labeled with categories in NELL (section 5.4).

5.10 Conclusion

In this section, we have presented a method to cluster verbs into semantically meaningful groups and propose them as new relations to extend the vocabulary of relations in NELL. We have shown in the experiments, that the verb clusters in our knowledge base outperform verb clusters in other existing, large-scale resources in terms of how well the clusters align to manually constructed verb clusters. We release this verb clustering in VKB full as part of our knowledge base of verbs, VerbKB[24].

[24] http://www.dwijaya.org/dvkb.html#DKVB

Chapter 6

Conclusion

In this thesis, we present a hypothesis that *we can semi-automatically construct a verb resource that goes beyond current resources in terms of coverage and links to knowledge bases, by leveraging a combination of high coverage text corpora, a knowledge base with a rich type system over entities, and other pre-existing linguistic resources such as a thesaurus and WordNet.*

We have demonstrated through the construction of our verb resource, VerbKB, that we can indeed construct such a verb resource that contains links from verbs to relations in knowledge bases and that goes beyond existing resources in terms of coverage.

VerbKB contains 65,679 unique verb patterns mapped into 215,106 binary relations, each typed with semantic categories in NELL and organized into a subsumption taxonomy based on NELL's hierarchy of types. The verbs in VerbKB cover subject-verb-object triples that occur a total of over 2 billion times in ClueWeb. We have also shown that the verb clusters in our VerbKB align better with manually constructed verb clusters compared to the verb clusters in other pre-existing large resources such as PATTY and PPDB. VerbKB clusters are then mapped to the existing NELL relations or are added as new relations to extend the vocabulary of relations in the NELL knowledge base.

We show the value of having the links from verbs to knowledge base relations in terms of relation extraction and the value of having the links from verbs to *changes* in knowledge base relations in terms of temporal scoping. To the best of our knowledge, ours is the largest publicly available knowledge base of English verbs to date that contains mappings from verbs to knowledge base relations and changes in these relations.

However, our knowledge base of verbs is by no means complete. We have discussed some of the current shortcomings of our knowledge base of verbs in the discussion section of every chapter. Some of the most important lessons we have learned in our construction of VerbKB are that:

1. We need to extend the definition of verb patterns in our VerbKB. Currently, we extract verb patterns between two noun phrases that are of the form V | VP where V is a verb lexeme and P is a preposition. However, we discover that a lot of relations in the knowledge base are not expressible by verbs and prepositions alone. For example, the relation ***hasBrother*** or ***hasSister*** are expressed by verbs such as "is a brother of" or "is a sister of", which are

combination of verbs, nouns, and propositions[1]. To extend the definition of verb patterns in our VerbKB, we can use ReVerb's definition of verb patterns that is V | VP | VW*P where W are either nouns, adjectives, adverbs, pronouns or determiners.

2. We need to explore other corpora beyond Wikipedia for learning the mappings between verbs and the changes in knowledge base relations as Wikipedia infoboxes are limited in the number of knowledge base relations they contain.

   One possible direction is to use corpora that have time stamps such as news documents as a source for learning the mappings between verbs in these news documents and the relations that are extracted from the documents.

   Another direction is to use knowledge of verbs in VerbNet for inferring the changes affected by verbs from their semantic predicates in VerbNet. For example, the verb pattern "deport" that appears in the syntactic frame "Agent *deport* Theme *to* Destination" has this set of semantic predicates in VerbNet: CAUSE(Agent, Event), LOCATION(START(Event), Theme, ?Source), LOCATION(END(Event), Theme, Destination). We can use it to infer that "deport" initiates LOCATION changes. If we can map these semantic predicates such as LOCATION to their corresponding knowledge base relations, we can map the verb pattern to the *changes* in the knowledge base relation affected by the verb pattern.

3. We need to continue adding knowledge to VerbKB and integrate knowledge from different resources to our VerbKB to further extend the knowledge and the usefulness of this knowledge base of verbs. Currently, we have shown that the knowledge in our VerbKB is useful for the task of relation extraction and temporal scoping of knowledge base instances.

   More knowledge can be added to VerbKB to support even more natural language understanding tasks. For example, knowledge such as typical temporal relations between verbs and the relations that they express [Wijaya et al., 2012] can be useful for improving temporal scoping or for other tasks such as event template construction.

   Another future direction is to align this knowledge base of verbs with other verb resources to add more knowledge about the verbs.

   Alignment to verb groups in VerbNet will be useful for adding knowledge about syntactic realizations of the verbs, semantic roles, semantic predicates and selectional restriction. The semantic predicates of verbs in VerbNet can be used to extend the mappings from verbs to changes in relations in our knowledge base of verbs.

   Alignment to frames in FrameNet will be useful for adding knowledge about frames and semantic roles, with the possibility of adding semantic types to FrameNet roles from the verbs' selectional preference computed from the corpus.

   Alignment to verb synsets in WordNet will be useful for extending the precision and recall of verb clusters and for creating a better subsumption hierarchy in the manner of [Grycner et al., 2015].

   The ultimate goal of alignments with these other existing resources will be to have all the knowledge of verbs, which are currently spread over different resources, related and

[1] Our sample of DBPedia and Freebase relations show that about 64.5% can be expressed by a combination of verbs and prepositions alone.

accessible in one unified place.

Bibliography

Eneko Agirre and Oier Lopez De Lacalle. Clustering wordnet word senses. In *RANLP*, volume 260, pages 121–130, 2003. 2.1.2

Waleed Ammar, George Mulcaire, Yulia Tsvetkov, Guillaume Lample, Chris Dyer, and Noah A. Smith. Massively multilingual word embeddings. *arXiv preprint arXiv:1602.01595*, 2016. 1.3, 5.1

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *The semantic web*. 2007. 1, 3.7.2

Stephen H Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. Hinge-loss markov random fields and probabilistic soft logic. *arXiv preprint arXiv:1505.04406*, 2015. 2.1.3

Collin F Baker, Charles J Fillmore, and John B Lowe. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1*, pages 86–90. Association for Computational Linguistics, 1998. 1.2

Colin Bannard and Chris Callison-Burch. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 597–604. Association for Computational Linguistics, 2005. 2.1.3

Kaustubh Beedkar and Rainer Gemulla. Lash: Large-scale sequence mining with hierarchies. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 491–503. ACM, 2015. 5.4.2

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, 2008. 1, 1.2

Thorsten Brants and Alex Franz. {Web 1T 5-gram Version 1}. 2006. 2.1.3

Carsten Brockmann and Mirella Lapata. Evaluating and combining approaches to selectional preference acquisition. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 27–34. Association for Computational Linguistics, 2003. 22

Lawrence D Brown, T Tony Cai, and Anirban DasGupta. Interval estimation for a binomial proportion. *Statistical science*, pages 101–117, 2001. (document), 3.4

Aoife Cahill, Nitin Madnani, Joel Tetreault, and Diane Napolitano. Robust systems for preposition error correction using wikipedia revisions. In *In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013. 4.8

J. Callan, M. Hoy, C. Yoo, and L. Zhao. Clueweb09 data set. *boston.lti.cs.cmu.edu*, 2009a. 1, 3.2, 2

Jamie Callan, Mark Hoy, Changkuk Yoo, and Le Zhao. Clueweb09 data set, 2009b. 1.2

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, volume 5, page 3, 2010. 1, 3.3, 3.6.2, 3.9, 4.1

Tsz Ping Chan, Chris Callison-Burch, and Benjamin Van Durme. Reranking bilingually extracted paraphrases using monolingual distributional similarity. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, pages 33–42. Association for Computational Linguistics, 2011. 2.1.3

Sam Chapman. Simmetrics. *URL http://sourceforge. net/projects/simmetrics/. SimMetrics is a Similarity Metric Library, eg from edit distance's (Levenshtein, Gotoh, Jaro etc) to other metrics,(eg Soundex, Chapman). Work provided by UK Sheffield University funded by (AKT) an IRC sponsored by EPSRC, grant number GR N*, 15764, 2009. 3.7.2

Timothy Chklovski and Patrick Pantel. Verbocean: Mining the web for fine-grained semantic verb relations. In *Proceedings of EMNLP 2004*, pages 33–40, 2004. 4.8

Anne Cocos and Chris Callison-Burch. Clustering paraphrases by word sense. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1463–1472, San Diego, California, June 2016. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/N16-1172`. 1.3, 2.1.3, 2.3, 5.1, 5.3, 5.8.2

Bhavana Dalvi, Einat Minkov, Partha P Talukdar, and William W Cohen. Automatic gloss finding for a knowledge base using ontological constraints. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 369–378. ACM, 2015. 3.6.4, 7

Anish Das Sarma, Alpa Jain, and Cong Yu. Dynamic relationship and event discovery. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 207–216. ACM, 2011. 4.9

Johannes Daxenberger and Iryna Gurevych. Automatically classifying edit categories in wikipedia revisions. In *EMNLP*, pages 578–589, 2013. 4.8

David R Dowty. *Word Meaning and Montague Grammar: The Semantics of Verbs and Times in Generative Semantics and in Montague's PTO*. Reidel, 1979. 2.1.2

Maisa C Duarte and Estevam R Hruschka. How to read the web in portuguese using the never-ending language learner's principles. In *2014 14th International Conference on Intelligent Systems Design and Applications*, pages 162–167. IEEE, 2014. 1.3, 3.1, 3.7.1

Stefan Engelberg. The magic of the moment: What it means to be a punctual verb. In

Annual Meeting of the Berkeley Linguistics Society, volume 25, pages 109–121, 1999. 2.1.2

Anthony Fader, Stephen Soderland, and Oren Etzioni. 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, 2011. 1.3, 2.1.3, 3.10, 4.1, 5.1, 5.3

Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166*, 2014. 5.9

Charles J Fillmore. The case for case. 1967. 2.1.2

John Rupert Firth. *Papers in Linguistics 1934-1951: Repr*. Oxford University Press, 1961. 2.1.3

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. Ppdb: The paraphrase database. In *HLT-NAACL*, pages 758–764, 2013. 2.1.3, 5.3

Matt Gardner, Partha Pratim Talukdar, Bryan Kisiel, and Tom Mitchell. Improving learning and inference in a large knowledge-base using latent syntactic cues. 2013. 1.2, 5.1

Matt Gardner, Partha Pratim Talukdar, Jayant Krishnamurthy, and Tom Mitchell. Incorporating vector space similarity in random walk inference over knowledge bases. 2014. 1.2

Matt Gardner, Partha Talukdar, and Tom Mitchell. Combining vector space embeddings with symbolic logical inference over open-domain text. In *2015 AAAI Spring Symposium Series*, volume 6, page 1, 2015. 1.3, 5.1

Dedre Gentner. Why nouns are learned before verbs: Linguistic relativity versus natural partitioning. technical report no. 257. 1982. 3.10

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. English gigaword. *Linguistic Data Consortium, Philadelphia*, 2003. 4.2.1

Adam Grycner and Gerhard Weikum. Harpy: Hypernyms and alignment of relational paraphrases. In *25th International Conference on Computational Linguistics*, pages 2195–2204. ACL, 2014. 2.1.3

Adam Grycner, Gerhard Weikum, Jay Pujara, James Foulds, and Lise Getoor. Relly: Inferring hypernym relationships between relational phrases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 971–981, 2015. 2.1.3, 3

Volker Haarslev and Ralf Möller. Racer: A core inference engine for the semantic web. In *EON*, volume 87, 2003. 1.2

Joshua K Hartshorne, Claire Bonial, and Martha Palmer. The verbcorner project: Findings from phase 1 of crowd-sourcing a semantic decomposition of verbs. In *ACL (2)*, pages 397–402, 2014. 2.1.2

Marti A Hearst. 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, 1992. 2.3

Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence*, 194:28–61, 2013. 1

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. Learning to solve arithmetic word problems with verb categorization. In *EMNLP*, pages 523–533. ACL, 2014. 4.8

Paul Jaccard. The distribution of the flora in the alpine zone. *New phytologist*, 11(2): 37–50, 1912. 3.7.2

Dan Jurafsky and James H Martin. *Speech and language processing*. Pearson, 2014. 5.4.3

Daisuke Kawahara, Daniel Peterson, and Martha Palmer. A step-wise usage-based method for inducing polysemy-aware verb classes. In *ACL (1)*, pages 1030–1040, 2014. 1.3, 2.1.3, 5.3, 5.8.1

Angelika Kimmig, Stephen Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. A short introduction to probabilistic soft logic. In *Proceedings of the NIPS Workshop on Probabilistic Programming: Foundations and Applications*, pages 1–4, 2012. 2.1.3

Paul Kingsbury and Martha Palmer. From treebank to propbank. In *LREC*. Citeseer, 2002. 1.2, 2.1.3, 3.9

Karin Kipper, Hoa Trang Dang, and Martha Palmer. Class-based construction of a verb lexicon. In *AAAI/IAAI*, pages 691–696, 2000. 1, 2.1.2, 3.9

Karin Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. Extensive classifications of english verbs. In *Proceedings of the 12th EURALEX International Congress*, pages 1–15, 2006. 1.2

George Lakoff. Stative adjectives and verbs in english. 1966. 2.1.2

Ni Lao, Tom Mitchell, and William W Cohen. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 529–539. Association for Computational Linguistics, 2011. 1.2, 2.2

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195, 2015. 1.2

Beth Levin. *English verb classes and alternations: A preliminary investigation*. University of Chicago press, 1993. 2.1.2

Marc Light and Warren Greiff. Statistical models for the induction and use of selectional preferences. *Cognitive Science*, 26(3):269–281, 2002. 5.2

Dekang Lin and Patrick Pantel. Dirt@ sbt@ discovery of inference rules from text. In

Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pages 323–328. ACM, 2001a. 1.2, 5.5.1, 5.5.2

Dekang Lin and Patrick Pantel. Discovery of inference rules for question-answering. *Natural Language Engineering*, 7(04):343–360, 2001b. 2.1.3, 3.8.2, 3.9, 5.2, 5.3

Dekang Lin, Kenneth Ward Church, Heng Ji, Satoshi Sekine, David Yarowsky, Shane Bergsma, Kailash Patil, Emily Pitler, Rachel Lathbury, Vikram Rao, et al. New tools for web-scale n-grams. In *LREC*, 2010. 2.1.3

Hugo Liu and Push Singh. Conceptneta practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226, 2004. 1.2

Yue Lu and Chengxiang Zhai. Opinion integration through semi-supervised topic modeling. In *Proceedings of the 17th international conference on World Wide Web*, pages 121–130. ACM, 2008. 3.6.1

Suresh Manandhar, Ioannis P Klapaftis, Dmitriy Dligach, and Sameer S Pradhan. Semeval-2010 task 14: Word sense induction & disambiguation. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 63–68. Association for Computational Linguistics, 2010. 5.8.1

CD Manning and P Schütze Raghavan. et-al. 2008. introduction to information retrieval. 5.7

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, 2014. URL `http://www.aclweb.org/anthology/P/P14/P14-5010`. 3.8.1, 4.4

Michal Mechura. *Selectional Preferences, Corpora and Ontologies*. PhD thesis, Ph. D. thesis, Trinity College, University of Dublin, 2008. 2.1.3, 5.2, 5.9, 22

Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K Gray, Joseph P Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, et al. Quantitative analysis of culture using millions of digitized books. *science*, 331(6014): 176–182, 2011. 4.2.1

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013. 2.1.3

George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995. 1

George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. Introduction to wordnet: An on-line lexical database*. *International journal of lexicography*, 3(4):235–244, 1990. 2.1.2, 3.9

Tom M. Mitchell, William W. Cohen, Estevam R. Hruschka Jr., Partha Pratim Talukdar, Justin Betteridge, Andrew Carlson, Bhavana Dalvi Mishra, Matthew Gardner, Bryan Kisiel, Jayant Krishnamurthy, Ni Lao, Kathryn Mazaitis, Thahir Mohamed, Ndapandula

Nakashole, Emmanouil Antonios Platanios, Alan Ritter, Mehdi Samadi, Burr Settles, Richard C. Wang, Derry Tanti Wijaya, Abhinav Gupta, Xinlei Chen, Abulhair Saparov, Malcolm Greaves, and Joel Welling. Never-ending learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 2302–2310, 2015. 4.1

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. Counter-fitting word vectors to linguistic constraints. *arXiv preprint arXiv:1603.00892*, 2016. 5.9

Ndapandula Nakashole and Tom M Mitchell. A knowledge-intensive model for prepositional phrase attachment. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics,(ACL)*, pages 365–375, 2015. 1.2

Ndapandula Nakashole and Gerhard Weikum. Real-time population of knowledge bases: opportunities and challenges. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pages 41–45. Association for Computational Linguistics, 2012. 4.1

Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. Patty: a taxonomy of relational patterns with semantic types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1135–1145. Association for Computational Linguistics, 2012. 1.3, 2.1.3, 8, 5.1, 5.3

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. Annotated gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pages 95–100. Association for Computational Linguistics, 2012. 2.1.3

Roberto Navigli. Meaningful clustering of senses helps boost word sense disambiguation performance. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 105–112. Association for Computational Linguistics, 2006. 5.8

Roberto Navigli, Kenneth C Litkowski, and Orin Hargraves. Semeval-2007 task 07: Coarse-grained english all-words task. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 30–35. Association for Computational Linguistics, 2007. 1.3, 5.1, 5.8

Rani Nelken and Elif Yamangil. Mining wikipedias article revision history for training computational linguistics algorithms, 2008. 4.8

Kamal Nigam, Andrew McCallum, and Tom Mitchell. Semi-supervised text classification using em. *Semi-Supervised Learning*, pages 33–56, 2006. 3.6.1

Ekaterina Ovchinnikova. *Integration of world knowledge for natural language understanding*, volume 3. Springer, 2012. 1.2, 2.1, 2.1.2, 2.2

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: bringing order to the web. 1999. 3.7.2

Marta Recasens, Cristian Danescu-Niculescu-Mizil, and Dan Jurafsky. Linguistic models for analyzing and detecting biased language. In *ACL (1)*, pages 1650–1659, 2013. 4.8

Roi Reichart and Ari Rappoport. The nvi clustering evaluation measure. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 165–173. Association for Computational Linguistics, 2009. 5.8.2

Philip Resnik. Selectional preference and sense disambiguation. In *Proceedings of the ACL SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How*, pages 52–57. Washington, DC, 1997. 5.2, 5.4.3

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. Relation extraction with matrix factorization and universal schemas. 2013. 2.2, 3.9

Ricardo Rodrigues, Hugo Gonçalo Oliveira, and Paulo Gomes. Lemport: a high-accuracy cross-platform lemmatizer for portuguese. *Maria João Varanda Pereira José Paulo Leal*, page 267, 2014. 1.4, 3.8.1

Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *EMNLP-CoNLL*, volume 7, pages 410–420, 2007. 5.8.1

Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987. 5.7

Herbert Rubenstein and John B Goodenough. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633, 1965. 2.1.3

Josef Ruppenhofer, Michael Ellsworth, Miriam RL Petruck, Christopher R Johnson, and Jan Scheffczyk. Framenet ii: Extended theory and practice, 2006. 2.1.2, 3.9

Ville Satopaa, Jeannie Albrecht, David Irwin, and Barath Raghavan. Finding a" kneedle" in a haystack: Detecting knee points in system behavior. In *2011 31st International Conference on Distributed Computing Systems Workshops*, pages 166–171. IEEE, 2011. 5.4.3

Roy Schwartz, Roi Reichart, and Ari Rappoport. Symmetric patterns and coordinations: Fast and enhanced representations of verbs and adjectives. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 499–505, San Diego, California, June 2016. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/N16-1060`. 2.3, 5.2

Stephen Soderland, Brendan Roof, Bo Qin, Shi Xu, Oren Etzioni, et al. Adapting open information extraction to domain-specific relations. *AI Magazine*, 31(3):93–102, 2010. 2.1.3, 2.2, 5.3

Angus Stevenson. *Oxford dictionary of English*. Oxford University Press, USA, 2010. 5.8

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM, 2007. 4.1, 4.4

Lin Sun and Anna Korhonen. Improving verb clustering with automatically acquired selectional preferences. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 638–647. Association for Computational Linguistics, 2009. 5.2, 5.3, 5.8.1, 5.9

P. P. Talukdar and K. Crammer. New regularized algorithms for transductive learning. *ECML*, 2009. 3.2, 7

Partha Pratim Talukdar, Derry Wijaya, and Tom Mitchell. Coupled temporal scoping of relational facts. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 73–82. ACM, 2012. 4.2.2

Stijn Marinus Van Dongen. Graph clustering by flow simulation. 2001. 5.6.1

Zeno Vendler. Verbs and times. *The philosophical review*, pages 143–160, 1957. 1.1, 2.1.2

Derry Wijaya, Partha Pratim Talukdar, and Tom Mitchell. Acquiring temporal constraints between relations. In *Proceedings of the Conference on Information and Knowledge Management (CIKM 2012)*, Hawaii, USA, October 2012. Association for Computing Machinery. 4.2.2, 3

Derry Wijaya, Partha Pratim Talukdar, and Tom Mitchell. Pidgin: ontology alignment using web text as interlingua. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 589–598. ACM, 2013. 3.1, 3.2, 3.9

Derry Wijaya, Ndapa Nakashole, and Tom Mitchell. CTPs: Contextual temporal profiles for time scoping facts via entity state change detection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2014a. 4.1, 4.2, 4.2.2

Derry Tanti Wijaya and Philip Gianfortoni. Nut case: What does it mean?: Understanding semantic relationship between nouns in noun compounds through paraphrasing and ranking the paraphrases. In *Proceedings of the 1st international workshop on Search and mining entity-relationship data*, pages 9–14. ACM, 2011. 1.2

Derry Tanti Wijaya and Tom M. Mitchell. Mapping verbs in different languages to knowledge base relations using web text as interlingua. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 818–827, San Diego, California, June 2016. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/N16-1096`. 1.3, 3.1

Derry Tanti Wijaya and Reyyan Yeniterzi. Understanding semantic change of words over centuries. In *Proceedings of the 2011 international workshop on DETecting and Exploiting Cultural diversiTy on the social web*, pages 35–40. ACM, 2011. 4.2, 4.2.1, 4.2.2

Derry Tanti Wijaya, Ndapandula Nakashole, and Tom M Mitchell. Mining and organizing a resource of state-changing verbs. In *Proceedings of the Joint Workshop on Automatic*

Knowledge Base Construction and Web-scale Knowledge Extraction, 2014b. 4.8

Derry Tanti Wijaya, Ndapa Nakashole, and Tom M Mitchell. ”A Spousal Relation Begins with a Deletion of engage and Ends with an Addition of divorce”: Learning State Changing Verbs from Wikipedia Revision History. In *Proceedings of the Comnference on Emprical Methods in Natural Language Processing (EMNLP)*, 2015. 1.3, 4.1

Fabio Massimo Zanzotto and Marco Pennacchiotti. Expanding textual entailment corpora from wikipedia using co-training, 2010. 4.8

Benat Zapirain, Eneko Agirre, Lluis Marquez, and Mihai Surdeanu. Selectional preferences for semantic role classification. *Computational Linguistics*, 39(3):631–663, 2013. 2.3, 5.2