A Large Resource for Patterns of Verbal Paraphrases

Octavian Popescu[1], Ngoc Phuoc An Vo[2], Vadim Sheinin[3]
IBM Research, Yorktown Heights, US
[1]o.popescu@us.ibm.com, [2]ngoc.phuoc.an.vo@ibm.com, [3]vadims@us.ibm.com

Abstract

Paraphrases play an important role in natural language understanding, especially because there are fluent jumps between hidden paraphrases in a text. For example, even to get to the meaning of a simple dialog like *I bought a computer. How much did the computer cost?* involves quite a few steps. A computational system may actually have a huge problem in linking the two sentences as their connection is not overtly present in the text. However, it becomes easier if it has access to the following paraphrases: [HUMAN] buy [ARTIFACT] $\Longleftrightarrow$ [HUMAN] pay [PRICE] for [ARTIFACT] $\Longleftrightarrow$ [ARTIFACT] cost [HUMAN] [PRICE], and also to the information that *I* **IsA** [HUMAN] and *computer* **IsA** [ARTIFACT]. In this paper we introduce a resource of such paraphrases that was extracted by investigating large corpora in an unsupervised manner. The resource contains tens of thousands of such pairs and it is available for academic purposes.

Keywords: verbal paraphrase, deep learning.

1. Introduction

When two phrases can be interchanged in a text without altering the meaning of the whole, we observe a paraphrasing relationship. Paraphrasing is a fundamental property of natural languages, and it is normally recognized as "saying the same thing with different words". Proposing a paraphrasing relation is a hard task for natural language processing (NLP) systems. The task of recognition of paraphrases was proposed in various SemEval competitions (Butnariu et al., 2009; Mihalcea et al., 2010; Specia et al., 2012; Xu et al., 2015) as an independent task or as a part of larger tasks like semantic similarity, textual entailment, etc. The resource we created is instrumental for all these tasks. Some paraphrases are made out of nominal phrases that contain only ostensible nouns and their adjectival determiners, like in *a fourteen year old boy* $\Longleftrightarrow$ *a teenager*. Another type of paraphrases are the ones that involve a verbal constituent, like *abandon a kid* $\Longleftrightarrow$ *ignore parental obligations*. This later type includes nominalizations, that is, even if the phrase has only noun constituents, at least one of them is a noun coming from a verb, like *abandoning a kid*. One major difference between nominal vs. verbal paraphrases is that the first ones are basically context independent, that is they can be substituted in a text directly, while the second are context dependent, their replacement in a sentence requires changes in the syntactic and semantic role of their complements and adjuncts.

The resource we compiled contains a list of pairs, each member being centered on a verb and its arguments. A pair is a valid verbal paraphrase relation given a certain context that is represented via types associated with each argument. For a given sentence that contains only one verb phrase, we can extract a set of paraphrases. For example, in Figure 1, for the sentence *I pay 1,200 for a laptop from Bestbuy*, we present a few valid verbal paraphrase extracted from the resource. In this example *X, Y, Z, U* are variable representing the head of syntactic components. [MONEY], [HUMAN], [ORG], [ARTIFACT] represent features that the variable must carry in order for a pair to be a valid paraphrase.

The verbal paraphrases occurring in this resource are patterns of verbal phrases, that is, they represent a generalization over various real instances in a text. The names of the features occurring on different syntactic slots in a paraphrase pattern are unimportant, but the class of the words that define the respective feature is important.

2. Related Work

One of the main ideas for the acquisition and recognition of verbal paraphrases was introduced in a seminal paper (Lin and Pantel, 2001). At the core of this approach lies the fact that paraphrases occur in the same context. A statistical approach based on the mutual information measure can filter out pairs of paraphrases from a given corpora.

However, this approach cannot solve two important problems: first, it is not the words by themselves that make two expressions paraphrases but, it is actually the role these words play inside the whole sentence; second it is not clear how the complements and adjuncts are aligned between the pairs. Even if a word is very frequent, like *I* or *you*, it is the feature [HUMAN] carried by both that is actually relevant for the meaning of the verbal phrases. The second problem is very challenging, as the same type of constituent can appear in different syntactic positions in the two expressions. For example, the adjunct [SHOP] in "[HUMAN] pay [PRICE] for [ARTIFACT] from [SHOP]" must appear in the subject position in "[SHOP] sell [ARTIFACT] to [HUMAN] for [PRICE]". In this paper we describe a technique able to cope with these problems which lead to the building of a resource of pattern paraphrases.

3. Pattern Paraphrases

The technique to extract pattern paraphrases is driven by the idea behind chain clarifying relationships (see among others (Popescu and Magnini, 2007; Kawara et al., 2014; Popescu, 2013; Popescu et al., 2014)). A chain clarifying relationship holds between the components of a verbal phrase if there is a unique combinations of senses that is legitimate. For example, in *I saw the river's bank.* vs. *I saw*

Sentence: I pay $700 for a laptop from Lenovo.

pattern: X pay Y for Z from U

X buy Y for Z	X write check of Z for Y	X make payment of Y for Z
X obtain Y for Z	U offer Z for Y to X	X authorize payment of Y for Z
U have Z for Y	X spend Y for Z	U receive Y for Z from X
X acquire Z for Y	X receive Z for Y	U accept payment from X for Z
X get Z for Y	U sell Z for Y to X	X cover expenses for Y
U charge X Z for Y	X give amount of Y for Z to U	X have Y for Z

Y is [MONEY] & X is [HUMAN] & Z is [ARTIFACT] & U is [ORG]
X isa {client, customer, buyer} & Y isa {commodity, merchandise, artifact} & U isa {shop, store}

Figure 1: Examples of verbal paraphrases.

a problem the verb *see* has two different meaning, *perceive by sight* vs. *to understand*. Also, *bank* has two meanings too, *sloping land* vs. *financial institution*, and *problem* has two meanings as well: *state of difficulty*, *question raised*. The combination of senses *perceive by sight a state of difficulty* is not legitimate, and neither *understand a financial institutions* is. In fact, in the sentence *I saw the river bank*, *river* imposes the *sloping land* reading to *bank*, which in turn imposes the *perceive by light* meaning on the verb *see*. That is why we talk about a chain clarifying relationship - words trigger the sense of other words in a chain like relation, as long as the words are components of phrases that have only one combination of senses possible.

The chain clarification relation is not defined by words, which are just instances of lexical units bearing certain features. In the example above, any word which is defined by the [PHYSICAL OBJECT] feature imposes the meaning *perceive by light* to the verb *see*. From this point of view, both *apple* and *book* have the same role, as both are carrying the [PHYSICAL OBJECT] feature. However, this similarity is restricted to the chain clarifying relationship for the verb *see*. While *apple* and *book* are antagonistic with respect to the verb *devour* as they impose two different chain clarifications for this verb, namely *eat* vs. *read avidly*.

Pattern paraphrases are pairs of chain clarifying relationships. The meaning of the whole verbal phrase is preserved, thereby creating a paraphrase relationship, by the fact that the same meaning of the verb and the same features are used.

4. Extracting Pattern Paraphrases from Large Corpora

4.1. Extracting Sub-categorization Framework for Verbal Phrases

The first step in the unsupervised extraction of pattern paraphrases is to consider a large corpus that is already parsed. We used Gigaword, LDC2012T21 (Napoles et al., 2012). For each verb, we extracted the verbal dependents. Due to parser errors, there are many such dependency paths that are noise. To filter them out we used COMLEX, http://nlp.cs.nyu.edu/comlex/, (Grishman et al., 1994). In the case where the direct object was governing a prepositional phrase, this prepositional phrase was included in the dependency path. In Figure 2 we see an example of such dependencies:

the *nsubj, dobj, iobj, prep_** is the head word of the nominal group having the respective role in the dependency path, *v_* marks the verb. As can be seen in this example, we also considered the partial paths, so the same sentence may lead to several instances of paths.

We further removed low frequency verbs, low frequency paths so that from an initial 1, 244, 793, 787 paths we filtered out a large number of them and we arrived to 391, 410, 259 paths that represent the closest approximation to a verb sub-categorization frame we could get. These paths contain 7, 922, 730 nouns in different syntactic positions and 25,812 verbs, which lead to 487,703 verbal phrases. These paths represent the input to a feed forward neural network that predicts the similarity of context. In a sense, we implemented a generalization of the original Lin algorithm that finds the dependencies paths that have the most similar context. From another point of view, we could think of the model created by this NN as dependency paths embedding. See Figure 3. e focused primarily on verbal groups, where a verbal group is defined by the following regular expression over dependency paths: $[sbj] + [obj|objprep]+[iobj]+[prepP||]*[prt]+v$ where `sbj` is the subject, obj is the object, obj prep matches the object and its governed preposition, if any, `iobj` is the indirect object, `prepP` is the attached prepositional group with its head, `prt` matches particles. For example, the following fragments of the dependency paths are matched by the above regular expression: $putprt\_upwith$, $putobj\_questionon$, $john\_sbjwalkto\_store$, $leaf\_sbjtouchhim\_objon_face$.

The obtained model cannot be used directly to predict paraphrasing, but its output represents a large list of candidates.

nsubj_army v_strengthen dobj_hold_hold
nsubj_army v_strengthen dobj_hold_hold prep_on_city_city
nsubj_army v_strengthen prep_on_city_city
v_make dobj_way_way
v_make dobj_way_way prep_for_neighborhoods_neighborhood
v_make prep_for_neighborhoods_neighborhood
nsubj_he_he v_support dobj_seizure_seizure
v_support dobj_seizure_seizure
nsubj_ministers_minister v_say
nsubj_rebels_rebel v_rebuff dobj_efforts_effort

Figure 2: Example of dependency path extracted from GigaWord.

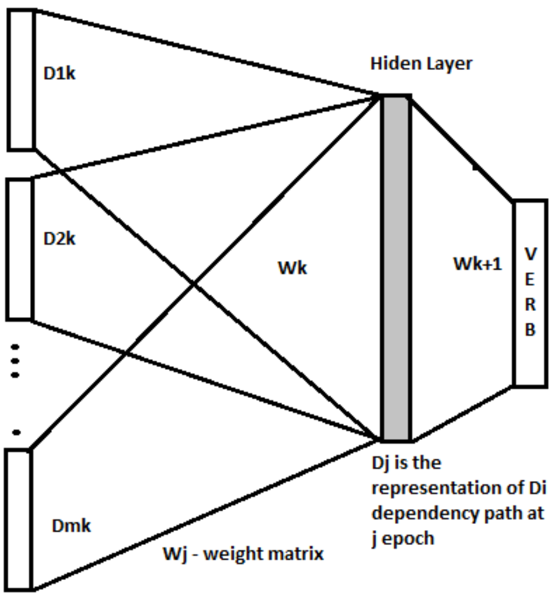

Figure 3: Dependency paths RNN.

The number of candidate pairs is 193, 628, 633. However, most of these pairs do not make it after the next filtering step.

4.2. Boostrapping from Mono-sense Verbal Phrases

In order to find the pattern paraphrases we need to find classes of words that are common between two candidates. However, due to the noise, we cannot get an accurate system of classes. Rather, we implemented a bootstrapping approach. For this, we used Zipf's law: the ambiguity of words is inversely proportional to its frequency rank. We started from verbs that according to WordNet are non ambiguous, therefore they have just one sense. We also considered linking verbs, *make, get, have, be etc.* together with their direct object and the propositional group, like *make way for* in Figure 4. These are mono-sense expressions. Then, we considered only the high probability paraphrases for these, which contain the more ambiguous verbs. We keep in separate classes the verbs from the later category, those multi-sense expressions, according to the mono-sense verb they paraphrase. If a multi-sense verbal expression occurs with two different mono-senses in a paraphrase relationship, then it is discarded. This bootstrapping process continues till we reach the most ambiguous verbs. In Figure 4 we show an example of the bootstrapping process. The *make way for* is a mono-sense verbal phrase, unlike *create* or *pave*. But the fact that at step 1 we determined that *make way for* and *create, pave* are valid candidates for paraphrasing leads to the creation of a cluster inside all the occurrences of *create*, and a cluster inside *pave*. The same happens for very ambiguous verbs like create or accommodate. All the occurrences inside this cluster can be paraphrased via MAKE WAY FOR. At the next step we will compute a precise contextual definition of these clusters.

4.3. Finding the Set of Features

The best way to find a set of features would be to have the agentive subject for each verb, like *buyer* for *buy*, with its preferred adjuncts in the set of paraphrases extracted from corpus. However, this kind of constructions are hardly present in a news corpus, as a sentence like *buyer buys*

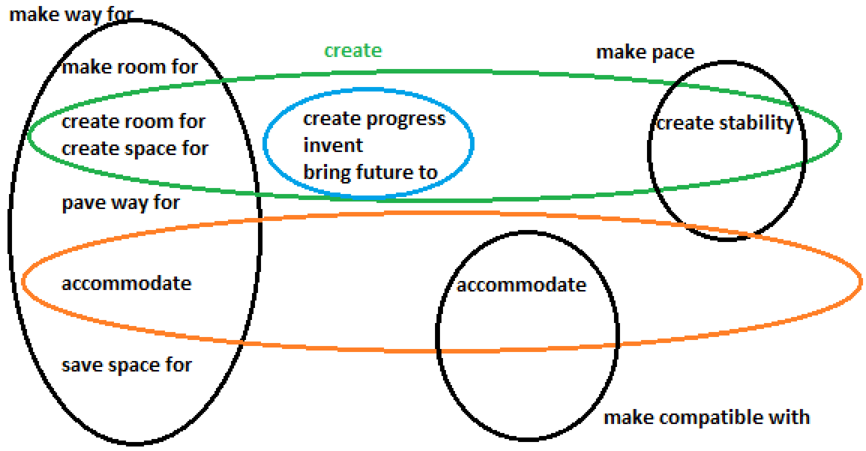

Figure 4: Bootstrapping from mono-sense verbs towards ambiguous verbs.

products is never used. We need to build the features for representing the pattern paraphrases in a bottom up approach, that is by finding the most general words that individualize that cluster vs all other clusters. In order to find the set of features for each verb separately we start from the clusters found at the previous step. Ideally, each cluster corresponds to a distinct sense of the verb. Inside each cluster, by considering the set of respective paraphrases, we compute the mutual information for each syntactic slot together with the word occurring in that syntactic slot, and rank them. At the top of this ranking we find the best representative words for that meaning, together with their syntactic functions. in the case of agentive verbs, we compute the Lin distance (Lin, 1998) on Wordnet(Miller, 1995) between it and the set of words occurring in that syntactic position, and we select the closest ones, for example *nsubj_customer , nsubj_client, nsubj_buyer* are the winners for verb *buy*. So, like in Figure 1, the paradigmatic set for variable **X** denoting the subject position is formed by these. Now, on the basis of the mutual information computed above, we find the most likely complements and their closest neighbors according to the Lin measure. The next step is to generalize the most likely fillers of verbal phrases as much as possible. This was carried out using the hypernym function from WordNet via SUMO ontology (Niles and Pease, 2003). Each word is replaced by its direct hypernym as long as the newly created form is not found in two clusters. In Figure 5 we present schematically this generalization process for three classes for the verb *move*. The three cluster identified at the previous steps, C1, C2 and C3 have different fillers for subject and object position respectively. The process of feature generalization is carried out as long as the obtained form of the pattern stay within the original cluster, that is there is no form that exist in two clusters at any time. For example , the first cluster and the third cluster in Figure 5 collide on object position, so the generalization this syntactic position stops shortly, while for for subject position it can go on further.

4.4. Seeds - Mono sense and frequent

The first observation is that the set of paraphrases generated by the Lin algorithm with class embedding is very accurate when the ambiguity of the target word is low and the number of occurrences is high. In this case the noise in classification is as low as it could be and thus the class context describes precisely the correct usage of the target word. The second observation is that class embedding preserves the meaning of the verbal group, so the semantic similarity between the set of correct paraphrases must be very high. The third observation is that the senses of the verbal group are paraphrased differently by using class embedding and thus a void intersection of class embedding indicates that the set of candidate paraphrases are indeed correct paraphrases. These observations suggest the following post filtering over the paraphrase candidate strategy:

- S1 Identify low ambiguity, high frequency verbal groups
- S2 Consider their set of candidate paraphrase;
- 3 Find the subset that minimizes the semantic distance
- S4 Consider the candidate paraphrase for each verbal group in this subset
- S5 Retain only the paraphrases that are common in these candidate sets
- S6 Repeat step 1 for the verbal groups in the retained paraphrase until the semantic distance is below a fixed threshold.

At S1 we used WordNet to decide on the ambiguity, and we used a linear combination of Lin distance with Roget similarity at step 3. The algorithm above produces a repository of paraphrases for each verbal groups. We obtained highly accurate paraphrases for 75,000 verbal groups, each verbal group being paraphrased in average with more than 250 paraphrases.

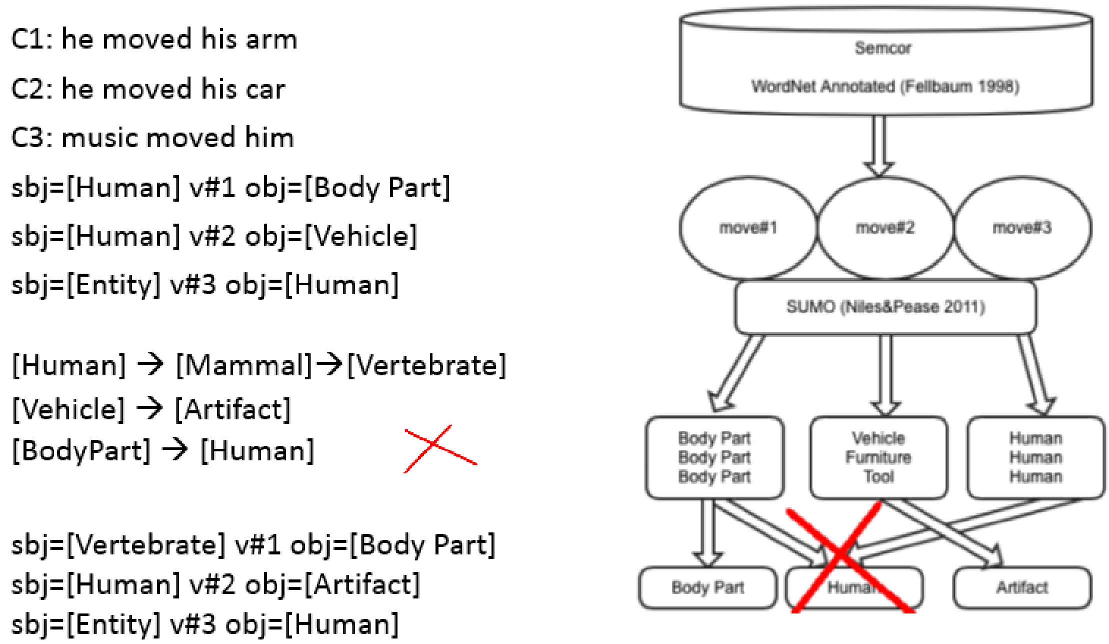

Figure 5: Generalization of features.

4.5. Slot Alignment between Paraphrases

The slot alignment is carried out via a computation of the most probable combination of arguments. This computation takes place in two steps. At the first step we consider the maximally probable configuration for each pair of paraphrases and at the second step we choose from this set the one that is the most probable considering all possible paraphrases into a cluster. Let's consider again Figure 1.

After the generalizing the slots, we have the pairs of the verbal group only, that is, we know that *buy, obtain, make a payment for, sell, get from, pay* etc. can enter a paraphrasing relationship in the same class for the verb *buy* (step 1&2 above) and we also know that this class has [CUSTOMER], [ARTIFACT], [ORG] as features for subject, direct object and prepositional group for the verb buy (step 3). The verbs *obtain, make payment for, sell, get from, pay* have their own syntactic slots for slightly different features, as the generalization process does not necessarily lead to the same features, but to variants of them, for example *client* vs. *customer*, or the same feature occurs in several slots.

First we employ a chain conditional formula for each pair of paraphrases in order to get the first one-to-one alignment. Given the form of one verbal phrase, we compute the probability that another verbal phrase has a certain realization. For example, we compute the probability that the verb *sell* has a certain configuration as $p(nsubj = [human_1], dobj = [artifact], prepTO = [human_2] \mid v = buy, nsubj = [human_2], dobj = [artifact], prepFROM = [human_1])$ (we use indexes to distinguish same feature in different syntactic position). In general, given two paraphrases in the same cluster, with $_{-t}$ denoting the target slots, we compute $argmax_{X_t,Y_t,Z_t} p(X_t, Y_t, Z_t)$ given the distribution of $X_c, Y_c, Z_c, v_c)$ of source pattern and v_t, V_c the target and source verbs respectively. For this probability we use the chain formula and we calculate the necessary independent probabilities over the whole corpus.

$$p(X_t, Y_t, Z_t \mid v_t, v_c, X_c, Y_c, Z_c) \approx \quad (1)$$
$$p(v_t \mid v_c, X_C, Y_c, Z_c)* \quad (2)$$
$$p(X_t \mid v_c, v_t, X_C, Y_C, Z_C)* \quad (3)$$
$$p(Y_t \mid X_t, v_t, , v_t, X_C, Y_C, Z_C)* \quad (4)$$
$$p(Z_t \mid Y_t, Z_t, v_c, , v_t, X_C, Y_C, Z_C) \quad (5)$$

(2) is the probability of v_t and v_c being paraphrase relation when the words of $-c$ are used, (3),(4) and (5) represent the probability of each slot for v_t for a given word, given that the v_t and v_c are in the same cluster of paraphrases.
An example of clusters: Cluster 1 X buys Y from Z vs. Z sell Y to X X make payment of W for Y at Z vs. Y cost W at Z X in[PER], Y in [ARTIFACT], Z in [ORG], W in [MONEY] Cluster 2 X provide assistance for Y vs. X deal with Y vs. X attend over Y, Y in [ACTIVITY] Cluster 3 X acquire Y vs. X become affected with Y , X in [PERS], Y in disease Cluster 4 X acquire Y vs. Y work for X , X in [ORG], Y in [PERS]

5. Evaluation Experiments

To evaluate paraphrases is a very difficult task, because there is not a gold standard. For limited data, human experts can verify manually the validity of some of them. However, our approach, PP, produces hundreds of different paraphrases for each verbal phrase.
We selected a set of 100 verbal expressions among which *abandon, be expert on, begin, buy, employ, expect, have address on, manage, plan , produce, solve work on, write* etc. We implemented the original DIRT algorithm (Lin and Pantel, 2001) and ran it for these 100 verbs over GigaWord, call it D100_G. We also considered word2vec with

	min	max	average
D100_G	40	80	70
S_w2V	30	90	76
G_w2V	30	90	85
PPDB	60	90	79
PP	70	100	90

Table 1: Precision s-level

	min	max	average
D100_G	24	33	27
S_w2V	37	62	58
G_w2V	37	68	59
PPDB	40	70	71
PP	60	100	82

Table 2: Precision m-level

	Precision			Recall		
	min	max	average	min	max	average
D100_G	8	12	8	8	42	20
S_w2v	15	25	17	49	73	58
G_W2V	15	27	19	49	77	60
PPDB	10	68	55	60	79	67
PP	40	78	68	69	94	79

Table 3: Precision l-level

	interagremment
s level	97
m level	93
l level	85

Table 4: annotator inter-agreement

the standard Google news model, call it S_w2v, and trained form GigaWord, call it G_w2v. Finally, we considered the set of paraphrases from PPDB (Ganitkevitch et al., 2013). The PPDB has a few levels of accuracy, s which very precise, small coverage, m, the medium precision and coverage, and l that is the large coverage, lower precision. As in PPDB, there are instances of paraphrases at the sentence level, we extracted 100 sentences from GigaWord for each verbal phrases, for a total of 10,000 sentences. We carried out two evaluation experiments. The first one focuses on pairs of verbal paraphrases, without considering any context. The second one considers the context around the verbal phrases in a given sentence and proposes a new paraphrase , if available. This second experiment cannot be carried out for DIRT, or w2v because these approach do not handle the context, so the systems evaluated here are ours, PP, and PPDB.

5.1. Pair to Pair paraphrase evaluation

For the 100 chosen verbs, we put together all the paraphrases created by each approach. For the DIRT and word2vec approaches we have to set a threshold under which two pairs are not consider paraphrases, as these approaches compute a score for each possible pair. We consider the first 10, 40 and 400 pairs, which create three levels of precision which we roughly equate with the s,m,l levels from PPDB. These thresholds were not exactly a random choice, because 10 is the average number existing in VerbOcean (Chklovski and Pantel, 2004), a paraphrase resource created with DIRT algorithm, while 40 is the standard number of similar phrases returned by word2vec. We also ranked the PP created by our approach based on the probability of occurrence of each pattern. In this way , we could have the same levels of 10, 40 and 400 paraphrases. So we create three distinct test corpora where each verb had the first 10, 40 and 400 returns from our approach, DIRT, word2vec and PPDB, respectively. There are not exactly 4000, 16 000, and 160 000 pairs of paraphrases, as some of the above resources may not have provided the required number of paraphrases. In the end we have three test corpora for the s,m,l level. Our experiment consists in extracting random samples from each of the test corpus and in evaluating their accuracy. We can now estimate how many pairs were correct on average for each approach, and how many correct paraphrases were contributed to the pool of correct paraphrases by each approach. We have three annotators, each one checking 2,500 pairs for correctness, out of which 250 were from the s level, 750 from the m level and 1,500 from the l level . Out of these 7, 500 pairs, 900 where common to all three annotators in order to compute their inter-agreement. The first three tables belows summarize the results of the evaluation for each of the s, m, l levels, and the fourth one shows the inter-agreement percentage.

5.2. Contextual Paraphrasing

There are 10,000 sentences that contain the chosen verbs that we extracted from Gigaword. For this sentences we can compare the accuracy of the whole text, not only of the verb. That is we can compare the effectiveness of paraphrase replacement in a specified context. Only our approach and PPDB can be compared in this experiment, as for DIRT and word2vec there is no immediate way to carry it out as this approaches do not contain contextual information. We considered the large level in order to maximize the chance that a given sentence matches an existing paraphrase in PPDB. From the 10,000 sentences we selected a random sample of 1,500 sentences and we gave them to the same three annotators, that is 500 for each. 90 sentences were common to all three annotators in order to observe their inter-agreement. The pp approach produced the correct replacement in the sentences in 46% of the sentences, while a suitable paraphrase was found in ppdb only in 19%. The inter agreement was 76%.

6. Conclusion and Further Work

We have compiled in a unsupervised way a large resource of pattern paraphrases that is available for academic purposes. A pattern is defined by a verb and a set of features that can occur on a specified syntactic position. A pattern matches some constituents in a given sentence by instantiating the features with corresponding words. The pattern is paraphrased by other patterns which do not necessarily assign the same syntactic roles to those constituents. Each pattern corresponds to a set of specific paraphrases which involve different other verbs. There are a few directions that could be exploited in order to increase the quality of this resource. For the moment, there are no paraphrases for noun phrases, including the ones that may contain adjectival determiners. This is a direction that we would like to exploit further. The adverbs were not taken into account when we extracted dependency paths, but they may play a role in the determination of certain pattern paraphrases. Another direction for improvement is to fill the gaps in the pattern set for certain verbs, that is, the algorithms acknowledges that some patterns have not been found yet, but their instances are present in text.

7. Bibliographical References

Butnariu, C., Kim, S. N., Nakov, P., Ó Séaghdha, D., Szpakowicz, S., and Veale, T. (2009). Semeval-2010 task 9: The interpretation of noun compounds using paraphrasing verbs and prepositions. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, pages 100–105. Association for Computational Linguistics.

Chklovski, T. and Pantel, P. (2004). Verbocean: Mining the web for fine-grained semantic verb relations. In *EMNLP*, volume 4, pages 33–40.

Ganitkevitch, J., Van Durme, B., and Callison-Burch, C. (2013). Ppdb: The paraphrase database. In *HLT-NAACL*, pages 758–764.

Grishman, R., Macleod, C., and Meyers, A. (1994). Comlex syntax: Building a computational lexicon. In *Proceedings of the 15th conference on Computational linguistics-Volume 1*, pages 268–272. Association for Computational Linguistics.

Kawara, D., Pederson, D., Popescu, O., and Palmer, M. (2014). Inducing example-based semantic frames from a massive amount of verb uses. In *Proceedings of the EACL 2014*. Association for Computational Linguistics.

Lin, D. and Pantel, P. (2001). 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.

Lin, D. (1998). An information-theoretic definition of similarity. In *ICML*, volume 98, pages 296–304.

Mihalcea, R., Sinha, R., and McCarthy, D. (2010). Semeval-2010 task 2: Cross-lingual lexical substitution. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 9–14. Association for Computational Linguistics.

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

Napoles, C., Gormley, M., and Van Durme, B. (2012). 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.

Niles, I. and Pease, A. (2003). Mapping wordnet to the sumo ontology. In *Proceedings of the ieee international knowledge engineering conference*, pages 23–26.

Popescu, O. and Magnini, B. (2007). Sense discriminative patterns for word sense disambiguation. In *SCAR workshop, NODALIDA*.

Popescu, O., Palmer, M., and Hacks, P. (2014). Mapping cpa onto ontonotes. In *Proceedings of the 9th International Conference on Language Resources and Evaluation - LREC14*.

Popescu, O. (2013). Learning corpus pattern with finite state automata. In *Proceedings of the ICSC 2013*.

Specia, L., Jauhar, S. K., and Mihalcea, R. (2012). Semeval-2012 task 1: English lexical simplification. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 347–355. Association for Computational Linguistics.

Xu, W., Callison-Burch, C., and Dolan, B. (2015). Semeval-2015 task 1: Paraphrase and semantic similarity in twitter (pit). In *SemEval@ NAACL-HLT*, pages 1–11.