Unsupervised Grammatical Pattern Discovery from Arabic Extra Large Corpora

Adelle Abdallah[1][a], Hussein Awdeh[1][b], Youssef Zaki[1], Gilles Bernard[1][c] and Mohammad Hajjar[2]
[1]*LIASD Lab, Paris 8 University, 2 rue de la Liberté 93526 Saint-Denis, Cedex, France*
[2]*Faculty of Technology, Lebanese University, Hisbeh Street, Saida, Lebanon*

Keywords: Arabic Language, Arabic Natural Language Process, Validation Information Retrieval, Silver Standard Corpus.

Abstract: Many methods have been applied to automatic construction or expansion of lexical semantic resources. Most follow the distributional hypothesis applied to lexical context of words, eliminating grammatical context (stopwords). This paper will show that the grammatical context can yield information about semantic properties of words, if the corpus be large enough. In order to do this, we present an unsupervised pattern-based model building semantic word categories from large corpora, devised for resource-poor languages. We divide the vocabulary between high-frequency and lower frequency items, and explore the patterns formed by high-frequency items in the neighborhood of lower frequency words. Word categories are then created by clustering. This is done on a very large Arabic corpus, and, for comparison, on a large English corpus; results are evaluated with direct and indirect evaluation methods. We compare the results with state-of-the-art lexical models for performance and for computation time.

1 INTRODUCTION

Lexical semantic resources are essential for a wide range of Natural Language Processing (NLP) tasks. The way to build automatically such resources from large corpora is usually by deducing semantic similarities between linguistic items from their distribution. The distributional hypothesis says that "words that are utilized and happen in the same contexts tend to purport similar meanings" (Harris, 1954), in other words "a word is characterized by the company it keeps" (Firth, 1957).

This hypothesis has been applied to measuring semantic similarity, to word clustering, to automated creation of thesauri and bilingual dictionaries, etc. But nearly every work has restricted the context to lexical items (hence the use of stopwords). This restriction is not part of the initial hypothesis, as seen in the work of Harris disciples on links between lexicon and grammar (Gross, 1994).

We try here a new approach, which is to identify the semantic properties that can be discovered from the grammatical patterns a word can be used into. The idea is to build a word vector based on grammatical patterns and then compare it to vectors produced by state-of-the-art methods based on lexical context. This approach, inspired by previous works (Bernard, 1997; Lebboss et al., 2017), is especially aimed at resource-poor languages as Arabic, as it is mostly built from knowledge extracted from big corpora.

Detection of grammatical patterns without knowledge from the language, even stopwords, is done by dividing the corpus vocabulary between high frequency items and low frequency items. Then we compute all possible patterns of high frequency items in the neighborhood of a word, considered as features of this word. The produced vector is the number of occurrences of each feature.

Evaluation of our results will be double: on one side, we will compare semantic properties deduced from grammatical patterns with those deduced by state-of-the-art methods from lexical data; on the other side, we will compare the results obtained from Arabic corpora with those obtained from an English corpus. We will use the few semantic gold standards available for Arabic and English, and apply an indirect evaluation in vector clustering, using WordNets for both languages.

[a] https://orcid.org/0000-0001-5837-8688
[b] https://orcid.org/0000-0002-2805-4444
[c] https://orcid.org/0000-0002-4587-4209

Abdallah, A., Awdeh, H., Zaki, Y., Bernard, G. and Hajjar, M.
Unsupervised Grammatical Pattern Discovery from Arabic Extra Large Corpora.
DOI: 10.5220/0010651700003063
In *Proceedings of the 13th International Joint Conference on Computational Intelligence (IJCCI 2021)*, pages 211-220
ISBN: 978-989-758-534-0; ISSN: 2184-2825

Next section will give a short survey of related works. Section 2 describes our method. Section 4 presents our results, before concluding.

2 RELATED WORKS

Unsupervised methods for extracting semantic knowledge from corpora are usually divided in two types: statistical and embedding methods. According to the distributional hypothesis, in both types of methods, a context must be defined, and each word is characterized by the frequency of its occurrence in each of these contexts. The contexts are usually defined as neighboring words in some window. The window can be a document, paragraph or sentence, or a fixed-length window.

Most of these methods follow a bag-of-words approach, where the order of items is not taken into account – among the rare exceptions, HAL (Lund et al., 1995); nearly all also follow a lexical context approach, where functional words are not taken into account.

The statistical methods whose origins date back to (Salton, 1971) are still very active. The principle of operation of these methods is simply to produce a vector where each column contains the number of occurrences of some context in the vicinity of the word. With a high number of contexts, as in big corpora, this produces a huge sparse vector, with a high computational and memory cost, so these methods use dimensionality reduction methods as singular value decomposition or other kind of matrices factorization, random mapping, etc.

These vectors are generated by various algorithms such as Latent Dirichlet Analysis (Blei et al., 2003), Probabilistic HAL (Azzopardi et al., 2005), Generalized Vector Space (Wong et al., 1985), Latent Semantic Analysis (Dumais, 2004), Rocchio Classification (Schutze et al., 2008), Random Indexing (Kanerva et al., 2000). Some methods enhance the vectors by pre- or post-processing techniques (Turney and Pantel, 2010; Erk, 2012; Clark, 2012), for instance weighing values with TF-IDF or PPMI.

To overcome the sparsity problem, word embedders have been developed. They represent a word in a dense low-dimensional vector using predictive models, usually neural network ones (Bengio et al., 2003; Collobert and Weston, 2008; Mnih and Hinton, 2008; Collobert et al., 2011; Dhillon et al., 2011; Mikolov et al., 2013a; Mnih and Kavukcuoglu, 2013; Collobert, 2014; Pennington et al., 2014). The principle here is to learn a weight matrix able to predict some neighboring word(s) of a given word. Components of a word vector do not represent directly the number of occurrences of some context, even if the number of times each context occurred influences the learning model.

Other than capturing semantic information, word embedding have successfully been employed by several downstream NLP applications such as named-entity recognition, semantic role labeling, sentiment analysis, machine translation and dependency parsing.

Despite their empirical success, recent papers shed light on the limitations of word embedders: (Levy et al., 2015b) targets its limitation in extracting the hyponymy and entailment relations, while (Rubinstein et al., 2015) pointed at its failure in capturing attributive properties. (Levy et al., 2015a) shows that the advantage of word-embedding methods over the count-based methods (Baroni et al., 2014) can be overridden with a good choice of hyper-parameters, which have more influence on the results than the chosen method. On the other hand, (Levy et al., 2015b) unveils that the word-embedding methods outperform the count-based models in diverse semantic tasks, such as word association, synonym detection and word clustering. (Mikolov et al., 2013b) use the offset method to solve analogical questions such as "man is to king as woman is to…?" by addition of vectors.

Whatever the method, the bag-of-word approach disregards grammar and word order and yields very little information on the syntactico-semantic relations between the words. To overcome this limitation, some solutions have been recommended. The first solution proposes injecting lexico-syntactic knowledge into the embeddings, such as dictionary, ontology or thesaurus information, to the objective function (Kiela et al., 2015; Lazaridou et al., 2015; Liu et al., 2015), or at the post-processing step (Faruqui et al., 2014; Mrkšić et al., 2016).

Others suggest a lexical pattern-based approach, replacing the bag-of-word contexts with other context types considering syntactic items. These methods were pioneered by (Hearst, 1992; Hearst, 1998) who manually coded patterns to capture hyponymy relations. Hand crafted patterns were used for many applications, such as extracting hyperonymy/hyponymy (Rydin, 2002; Tjong Kim Sang and Hofmann, 2007), meronymy (Berland and Charniak, 1999), antonymy (Lin et al., 2003), noun categories (Widdows and Dorow, 2002). Some methods based on Hearst patterns implement weakly supervised bootstrap techniques (Riloff and Shepherd, 1997), while others refined its algorithm using the syntactic structure (Roark and Charniak, 1998; Widdows and Dorow, 2002; Phillips and Riloff, 2002; Pantel and Ravichan-

dran, 2004; Tanev and Magnini, 2006). Other attempts use lexical-syntactic contextual patterns (Riloff et al., 1999; Thelen and Riloff, 2002), one even uses only grammatical information (Bernard, 1997).

(Caraballo, 1999) was the first to use conjunctions and appositive features to build a hyperonym-labeled noun hierarchy similar to WordNet. (Alfonseca and Manandhar, 2002) use Hearst patterns in an unsupervised manner to extend ontology and compare it with WordNet. In this sense, Hearst patterns were applied to increase recall in Information Extraction systems as KNOWITALL (Etzioni et al., 2004; Ritter et al., 2009), to extract web-based semantic relations (Pasca, 2004), and in various other studies (Sang, 2007; Sombatsrisomboon et al., 2003; Sumida and Torisawa, 2008).

Other works take into account the dependency relations (Lin, 1998; Padó and Lapata, 2007; Murphy et al., 2012; Levy and Goldberg, 2014). Some even use clusters of lexical-syntactic patterns in which the word occurs as context to represent it (Baroni et al., 2010). (Bollegala et al., 2015) replaces bag-of-words contexts with various patterns (lexical, POS and dependency). Another work (Yatbaz et al., 2012) uses paradigmatic representations of word context by replacing the bag-of-word contexts with substitute vectors, which include the potential words that could replace the target word given its neighboring words.

All these methods make use of knowledge about the language. Alternatively, the flexible approaches aim to extract patterns in an unsupervised manner. Such methods were used in various NLP tasks as constructing noun categories (Davidov and Rappoport, 2006) analogy question answering (Biçici and Yuret, 2006), extracting semantic relationships (Turney, 2008a; Bollegala et al., 2009; Davidov et al., 2007), detecting synonyms (Turney, 2008b), disambiguation of nominal compound relations (Davidov and Rappoport, 2008), sentiment analysis (Davidov et al., 2010) and detection of sarcasm (Tsur et al., 2010) or irony with Probabilistic LDA (Nozza et al., 2016).

In a similar way, (Lebboss et al., 2017; Lebboss et al., 2019), building on (Bernard, 1997), cluster word vectors based on grammatical patterns. Others use statistical graph-based approaches for automatic extraction of semantic relations (Vossen, 1998; Widdows and Dorow, 2002; Shinzato and Torisawa, 2004).

3 OUR METHOD

In order to assess the quality of semantic information that can be extracted from automatically constructed grammatical pattern, the flexible approach is the most convenient. Very few tests have been made on other languages than English, and only one (Lebboss et al., 2017) has been done on Arabic.

In this last work, the corpus vocabulary was cut in two parts, high-frequency elements and lower-frequency elements. Then patterns were generated from the high-frequency elements in the vicinity of words, forming vectors to be clustered and compared for evaluation with Arabic WordNet. There are drawbacks: the method explores only some of the possible patterns; it cannot work on a really big corpus; the evaluation was partial and only done on Arabic data. But on the other side, the method was not language-dependant, was based on grammatical information[1], and had a very few number of parameters, relatively easy to tune.

The method presented here corrects all these drawbacks, adding different testing protocols (every protocol available for Arabic), with a much larger corpus (one billion instead of six millions words), while reducing the computation time and minimizing occupied space.

This is done in a modular and multi-threaded framework (https://gitlab.com/Data-Liasd/Interface), used by various projects of our laboratory since 2014, and collecting tools for extracting language structure from extra-large corpora. It is mainly written in C++, with parts in Python and Java, Qt5 library, PostgreSQL database and Cmake. Figure 1 presents the general operation of our system, and its three main stages will be presented in next subsections.

3.1 Word Trie Preparation

A Trie or Prefix Tree, is one of the best data structures for word indexing. It stores words by nodes; each node has sons and brothers, and contains one Unicode letter. Every common prefix is represented by one node; each node branches off when the letters diverge from the other prefixes in the Trie. The word is represented by merging the characters from the root node to the end node. The structure is compact as it stores the common prefixes only once. Unlike hash-code or binary trees where the search time of a word is proportional to the number of stored elements, search time here is proportional to word length. It is thus faster as the maximum length of a word is lower than

[1] In a big enough corpus, high-frequency words are, as is well known, mostly functional ones.

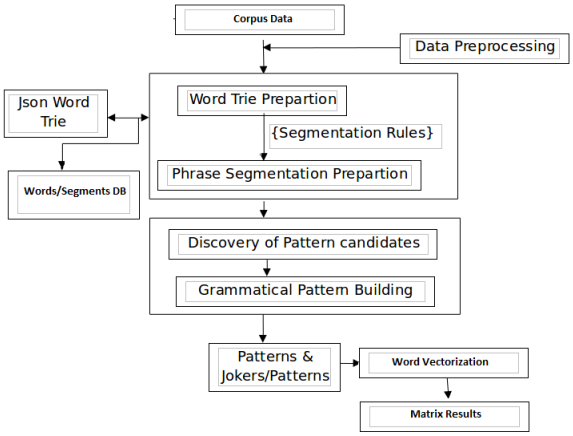


Figure 1: Our system.

the number of words. It is also more efficient than hashing since it does not require hash function computing or collision handling. The only drawback is that it consumes a lot of space if the words are heterogeneous – few common prefixes (which is not the case here).

We store the Trie in two ways: globally as a Json file or by nodes in the database. It is worthy to note that the word-trie is built before applying any preprocessing, in order to generate various tests without having to re-read the corpus. For reasons that will appear later, word identifiers are frequency-based.

Word preprocessing integrates Khoja stemmer (Khoja and Garside, 1999) implemented in Java by Motaz Saad[2]), Arabic IBM normalization rules implemented by Stanford NLP group[3], among others. In normalization preprocessing, we also replace all numbers by '1', and all words in foreign scripts by '0', in order to keep track of those as potential context. After every preprocessing, a mapping to the initial word-trie is stored in the database.

3.2 Phrase Segmentation Preparation

Before building patterns, we first segment the corpus in small textual units that will serve as context. The segments are delimited with punctuation (including carriage return and linefeed). The type of punctuation used for boundaries has an impact on the pattern results. For Arabic, cutting at double punctuations (parenthesis, brackets) or at quotes gave us lowest quality results and induced bad grammar rules. For instance, prepositions were linked to the word before and not to the following word. So we suppressed double punctuations and quotes from the list of delimiters, and obtained much better results.

Segments are stored as arrays of preprocessed word identifiers (the Trie structure is not convenient here) with file swapping for multithreading. We then clean the segments that would not be useful for context determination: those reduced to one word, or containing only digits, only foreign words, or both.

3.3 Pattern Discovery

The first step here is to split the vocabulary between high-frequency and lower-frequency items. This is done by selecting a threshold, the first parameter of our system; though we have done it by hand, it could be automated by looking at the Zipf curve. We term the high-frequency items "markers"; most of them are grammatical words; among the rare exceptions, "ibn" (literally "son of") marks proper nouns, "he said" marks a hadith (as in English "once upon a time" marks a tale). Lower-frequency items will simply be called words. Testing if a form is a marker is done by comparing the form identifier with the identifier of the last defined marker. If it is greater, it is a marker, otherwise it is a word. We have two basic grammar rules:

1. A pattern contains at least one word,
2. A pattern contains at most JL consecutive words.

The JL parameter (for JokerLength) mentioned in the last rule is our second parameter; it is the maximum number of consecutive words that can be attached to a pattern. We define a joker (a wildcard) as a sequence of words whose length is comprised between 1 and JL. A pattern is then a sequence of markers, coined m_i in the following examples, and jokers, coined $*$. Patterns are most appropriately stored in a Trie; every node in the Trie contains:

- a label (the marker or joker identifier),
- the number of outputs attested,
- a map of word identifiers and occurrences in the pattern.

The second step is to cut the segments into pieces that corresponds to patterns of maximum length (according to rules 1 and 2 above). For instance, if words are coined x, the segment $x\ m_1\ x\ x\ x_i\ x\ x\ m_2\ x$, if $JL = 3$, must be cut in two: $x\ m_1\ x\ x\ x_i$ and $x_i\ x\ x\ m_2\ x$ (note that x_i is in both parts).

The third step is to generate, for each part of a segment, every pattern that is compatible with it[4], to

[2]https://github.com/motazsaad/khoja-stemmer-command-line

[3]https://github.com/stanfordnlp/CoreNLP/blob/master/src/edu/stanford/nlp/trees/international/arabic/ArabicUtils.java

[4]In (Lebboss et al., 2017) this third step did not exist.

store the pattern in the pattern-trie, and to store in the map of the pattern every word that has occurred in it, updating its number of occurrences. The pattern-trie is constructed in multithreaded master-slave mode.

We will keep only the patterns that appear in the corpus more than some frequency threshold: this threshold is our last parameter.

The last step is to generate the matrix *words* × *patterns*. We developed three methods to generate the matrix, depending on the corpus size, to optimize the performance. The matrix is stored in compressed format; only the non-zero components are recorded with the word and pattern identifier and the number of occurrences.

1. For small corpora: After merging the pattern-tries, we parsed and generated the *word* × *pattern* matrix (as a map) along with the parsing.
2. For medium corpora (> 70 million segments, 6 million unique words): After merging the pattern-tries, each component of the matrix is stored in a table, and then SQL queries are used to group all patterns for each word.
3. For large corpora: At the end of each local Trie, the partial components of the matrix are stored in a table. We retrieve the different identifiers for the same pattern. In the database, we merge these identifiers (by SQL queries) to obtain the components of the matrix; then we proceed as for the medium corpora.

3.4 Evaluation

We conduct two types of evaluation. In the first one, based on the few semantic similarity gold standards available for Arabic (RG65, MC30 and WordSim353), dot products[5] between vectors of the words included in the gold standard are computed; then correlations (Pearson, Spearman) between these and the human scores are computed. Both correlations give in general close results, so we will only give Pearson here.

The second evaluation is more complex but better suited for Arabic and resource-poor languages. It is based on WordNet. The idea is to compare similarities in WordNet (graph-vicinity) to the result of a clustering. Of course, WordNet similarities compare concepts (*synsets*), not words, so they cannot be directly used. Following (Aliane, 2019), the similarity between words chosen is derived from *wup_similarity* (Wu and Palmer, 1994) as follows:

$$sim(m_k, m_j) = \underset{\substack{k \in synset(m_k),\\ j \in synset(m_j)}}{ArgMax}\ wup\_similarity(k, j) \quad (1)$$

Among the numerous clustering models avalaible, we chose SOM (Kohonen, 1982); the main reason for this choice is that in later stages we aim to explore cluster topology and analyse metaclusters in the map; another reason was the existence of a very efficient version for sparse vectors (Melka and Mariage, 2017), which we used for our method, while using regular SOM with dense vectors. The quality of each cluster is computed on the basis of similarity between words in the cluster, according to:

$$Qlt(C_i) = \frac{\sum_{k=1}^{|c_i|-1} \sum_{j=k+1}^{|c_i|} sim(m_k, m_j)}{|c_i|.|c_i - 1|/2} \quad (2)$$

The global quality is the average on all k clusters:

$$Qlt = \frac{\sum_{i=1}^{k} Qlt(C_i)}{k} \quad (3)$$

We compare the results of our method based on grammatical information to three state-of-the-art methods based on lexical information, CBoW, Skip-Gram and Glove (Mikolov et al., 2013a; Pennington et al., 2014), both on Arabic and on English.

The corpora used are given in table 1. For Arabic, we used three corpora of different sizes, in order to study the influence of corpus size on the results: a big extract of a very large corpus created by (Lebboss, 2016), hereafter ABC (for Arabic Big Corpus), the reduced corpus on which his method was experimented (ARC), and Arabic Wikipedia 2019 (AW). For English, we used a single large corpus, Wikipedia 2019 (EW). The support is the number of words common to the corpus and WordNet (Arabic WordNet and Princeton WordNet respectively), for wup evaluation. Uniques means number of unique normalized words.

Table 1: Corpora.

Corpus	Words	Uniques	Support
EW	2,250,594,333	9,257,846	47,118
ABC	1,021,006,238	6,084,342	4,159
AW	119,435,658	2,700,803	3,566
ARC	8,573,345	544,796	822

4 EXPERIMENTS

Our experiments were carried on a 64-bits server with 72 CPU and 125 GB of memory. For each method tested, we have empirically determined the parameter values giving the best results. For CBOW and

[5]We also tried Euclidian distance but dot product gives better results.

SkipGram, we used negative sampling and a four-words window. For GloVe, we used a ten-words window. For all three methods, we adopted 15 iterations and 300 as vector dimension. For our method, based on the observations below, we choose the number of markers as 150 in Arabic and 75 in English, with JL=4. Threshold for the patterns (depending on the corpus) was fixed for obtaining a vector of $\approx$ 1500 dimensions. For the clustering, the best results were obtained with the SOM architecture and operation recommended by (Kohonen, 1982) (hexagonal neighborhood, Euclidian distance, and learning parameter computed from the number of samples in two stages), and with $\approx$ 550 clusters.

We will begin by some preliminary observations, then give the results of the two types of evaluation.

4.1 Observations

As expected, the frequency distribution of words follows a Zipf Law. In English, 50 markers cover more than 40% of the occurrences, while Arabic needs more markers to cover the same percentage (150 with ABC corpus). This implies that Arabic has more structural diversity of the patterns.

More interesting, as shown in figure 2, the distribution of the segments, while being a long tail distribution (with 4,357,913 segments having only one occurrence), does not obey any kind of Zipf distribution. The black line in the figures represent the Zipf curve closer to the data. To the contrary, the distribution of the patterns (figure 3) strictly follows a Zipf Law (with only two humps compared to word distributions). This is independent of the value of the parameters, and of the language.

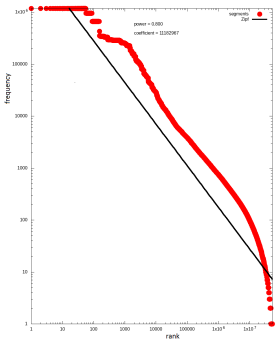
The average length of the segments is 8.67 in English and 9.22 in Arabic (for the biggest corpus). Average length of the patterns is 7,1 in English and 7,75 in Arabic. The average number of markers in a pattern is $\approx$ 5.

Study of the influence of number of markers and JL value on the number and length of patterns yields the following observations:

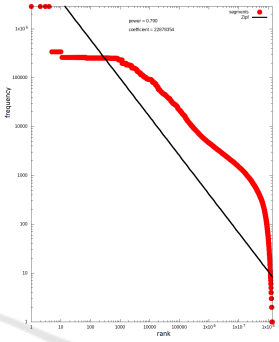
- when the number of markers grows, so does the number of patterns, but their average length diminishes;
- length and number of patterns grow with JL until a plateau is reached; the bigger the corpus, the sooner the plateau is reached (JL = 3 or 4).

4.2 Semantic Similarity

The results for WordSim353 (table 2, with Pearson correlation) are not good for any method, but very bad



(a) Arabig Big Corpus.



(b) English Wikipedia.

Figure 2: Log-log segment frequency distribution.

for our method. For Arabic they are worse when the corpus size is bigger.

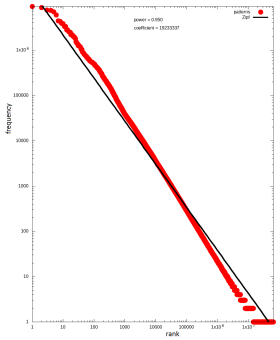
Table 2: Results for Direct Evaluation with WordSim353.

Corpus	CBow	SkipG.	GloVe	Our
EW	0.63	0.66	0.66	0.03
ABC	0.36	0.39	0.40	0.1
AW	0.46	0.48	0.49	0.07
ARC	0.52	0.45	0.46	-0.06

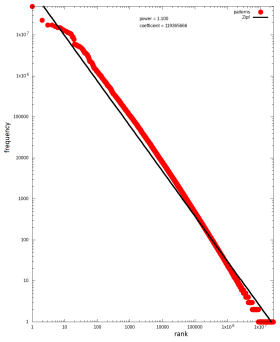
The results for RG-65 and MC-30 are very similar, so we give only the first ones (table 3); Pearson correlation is good with all state-of-the-art methods (Spearman, not given, is somewhat lower) and not good with our method, but especially bad for English. For Arabic the same hold than for WordSim353: the correlation worsens when the corpus size is bigger.

Table 3: Results for Direct Evaluation with RG.

Corpus	CBow	SkipG.	GloVe	Our
EW	0.78	0.78	0.74	0.02
ABC	0.88	0.89	0.92	0.41
AW	0.90	0.92	0.92	0.47
ARC	0.99	0.99	0.99	0.59



(a) Arabig Big Corpus.



(b) English Wikipedia.

Figure 3: Log-log pattern frequency distribution.

Thus it is clear that our method, only based on grammatical information, does not capture this type of semantic similarity. Still, results are better for Arabic and RG65 (or MC30).

4.3 WordNet based Similarity

The situation is not at all the same with WordNet similarity in indirect evaluation by clustering. As shown in table 4, our method can give better results in *Qlt* for Arabic than state-of-the-art ones; the condition seems to be that the corpus be large. For English, even if the results are not as good as state-of-the-art ones, they are largely better than those for semantic similarity.

Table 4: Results for WordNet similarity.

Corp.	CBoW	SkipG.	GloVe	Our
EW	**0.563**	0.560	0.518	0.399
ABC	0.419	0.402	0.398	**0.504**
AW	0.425	0.422	0.411	**0.459**
ARC	**0.455**	0.446	0.450	0.390

Difference between English and Arabic here is perhaps correlated to the more diverse pattern structure in Arabic observed before, which could reflect that grammatical structure conveys more semantic information in Arabic.

5 CONCLUSION

The most notable (and unexpected) observation, never done before to the best of our knowledge, is that patterns, like words but unlike segments, follow Zipf Law, at least in English and Arabic. The cause of Zipf Law in word distribution has not been settled; if this phenomenon is confirmed in other languages, it would mean that patterns are structurally similar to words and not to segments or sentences, and perhaps bring new light into the debate. The "monkey" model for Zipf Law (Miller, 1957) aims to explain it as a random effect; but why does it apply to words and patterns and not to segments or sentences?

Second, considering the issue we have adressed here, our results show that at least in some cases semantic information can be deduced from grammatical patterns, even at times better than from lexical patterns. It seems clear that this holds more for languages with more diverse pattern structure, as Arabic, than for English. Of course that does not mean that one should not look at lexical patterns, rather than both types of information should be combined.

It is hard to assess what exactly is at stake between semantic similarity gold standards and WordNet cluster similarity. First, one may note that those gold standards are very small (hundreds of word pairs at most), while WordNet, even for Arabic, is much more populated (with millions of potential word pairs). Second, those gold standards do not distinguish between types of similarity, while WordNet is by design much more precise.

Whatever the case, we hope to have convinced the reader that the contribution of grammatical structure to semantic characterization of words deserves to be explored.

Our strategy for discovering grammatical structure functions at an acceptable computational cost, as shown in table 5, at least for Arabic. While Lebboss program took 125 minutes for the ARC corpus (and did not compute all patterns), our program takes only 2 minutes. We have optimized our algorithm only on the basis of Arabic data, and a comparable optimization is probably needed for English, in order to shorten the computation time. The most costly in the process is the pattern discovery (taking more than 90% of the total time).

Table 5: Computation times.

Method	ABC	AW	ARC	EW
GloVe	31mn	7mn	1mn	3.88h
CBoW	1.68h	28mn	2mn	9.8h
SkipGram	7.15h	128mn	5mn	45.56h
Our	7.49h	36mn	2mn	122.9h

For the time being a human intervention is necessary for splitting the vocabulary, but this could be automated by taking into account the higher hump of Zipf curve. Absolutely no knowledge of the language nor of its grammar is needed here.

We aim to develop further different aspects of this work: analyse the relations between clusters in the SOM map produced; research about distributional properties of grammatical patterns on other languages; look closer at English features that could explain the much longer computation times; last, but not least, elaborate a method integrating lexical and grammatical patterns in order to categorize words, especially for resource-poor languages.

ACKNOWLEDGEMENTS

This work has been done as a part of the project "Analyses sémantiques de textes arabes utilisant l'ontologie et WordNet" supported by Paris 8 University and the Lebanese University.

REFERENCES

Alfonseca, E. and Manandhar, S. (2002). Improving an ontology refinement method with hyponymy patterns. *Language Resources and Evaluation. Las Palmas: LREC*.

Aliane, N. (2019). *Evaluation des représentations vectorielles de mots.* PhD thesis, Paris 8.

Azzopardi, L., Girolami, M., and Crowe, M. (2005). Probabilistic hyperspace analogue to language. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 575–576.

Baroni, M., Dinu, G., and Kruszewski, G. (2014). Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247.

Baroni, M., Murphy, B., Barbu, E., and Poesio, M. (2010). Strudel: A corpus-based semantic model based on properties and types. *Cognitive science*, 34(2):222–254.

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Berland, M. and Charniak, E. (1999). Finding parts in very large corpora. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics*, pages 57–64.

Bernard, G. (1997). Experiments on distributional categorization of lexical items with Self Organizing Maps. In *Proceedings of WSOM*, volume 97, pages 4–6.

Biçici, E. and Yuret, D. (2006). Clustering word pairs to answer analogy questions. In *Proceedings of the Fifteenth Turkish Symposium on Artificial Intelligence and Neural Networks (TAINN 2006), Akyaka, Mugla, Turkey*.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Bollegala, D., Maehara, T., Yoshida, Y., and Kawarabayashi, K.-i. (2015). Learning word representations from relational graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Bollegala, D. T., Matsuo, Y., and Ishizuka, M. (2009). Measuring the similarity between implicit semantic relations from the web. In *Proceedings of the 18th international conference on World wide web*, pages 651–660.

Caraballo, S. A. (1999). Automatic construction of a hypernym-labeled noun hierarchy from text. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics*, pages 120–126.

Clark, S. (2012). Vector space models of lexical meaning. *Handbook of Contemporary Semantics–second edition. Wiley-Blackwell*, page 8.

Collobert, R. (2014). Word embeddings through Hellinger PCA. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*. Citeseer.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of machine learning research*, 12:2493–2537.

Davidov, D. and Rappoport, A. (2006). Efficient unsupervised discovery of word categories using symmetric patterns and high frequency words. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 297–304.

Davidov, D. and Rappoport, A. (2008). Classification of semantic relationships between nominals using pattern clusters. In *Proceedings of ACL-08: HLT*, pages 227–235.

Davidov, D., Rappoport, A., and Koppel, M. (2007). Fully unsupervised discovery of concept-specific relationships by web mining. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 232–239.

Davidov, D., Tsur, O., and Rappoport, A. (2010). Enhanced sentiment learning using twitter hashtags and smileys. In *Coling 2010: Posters*, pages 241–249.

Dhillon, P. S., Foster, D., and Ungar, L. (2011). Multi-view learning of word embeddings via cca. *Proc. Of NIPS*.

Dumais, S. T. (2004). Latent semantic analysis. *Annual review of information science and technology*, 38(1):188–230.

Erk, K. (2012). Vector space models of word meaning and phrase meaning: A survey. *Language and Linguistics Compass*, 6(10):635–653.

Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D. S., and Yates, A. (2004). Web-scale information extraction in knowitall: (preliminary results). In *Proceedings of the 13th international conference on World Wide Web*, pages 100–110.

Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E., and Smith, N. A. (2014). Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166*.

Firth, J. R. (1957). A synopsis of linguistic theory 1930-55. *Studies in Linguistic Analysis*, Special vol:1–32.

Gross, M. (1994). Computational approaches to the lexicon. chapter Constructing Lexicon-Grammars, pages 213–263. Oxford University Press.

Harris, Z. S. (1954). Distributional structure. *Word*, 10(2-3):146–162.

Hearst, M. (1998). Wordnet: An electronic lexical database and some of its applications. *Automated Discovery of WordNet Relations*.

Hearst, M. A. (1992). Automatic acquisition of hyponyms from large text corpora. In *Coling 1992 volume 2: The 15th international conference on computational linguistics*.

Kanerva, P., Kristoferson, J., and Holst, A. (2000). Random indexing of text samples for latent semantic analysis. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 22.

Khoja, S. and Garside, R. (1999). Stemming arabic text. *Lancaster, UK, Computing Department, Lancaster University*.

Kiela, D., Hill, F., and Clark, S. (2015). Specializing word embeddings for similarity or relatedness. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2044–2048.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 43(1):59–69.

Lazaridou, A., Baroni, M., et al. (2015). A multitask objective to inject lexical contrast into distributional semantics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 21–26.

Lebboss, G. (2016). *Contribution à l'analyse sémantique des textes arabes*. PhD thesis, Paris 8.

Lebboss, G., Bernard, G., Aliane, N., Abdallah, A., and Hajjar, M. (2019). Evaluating methods for building arabic semantic resources with big corpora. In *Studies in Computational Intelligence*, volume 829, pages 179–197. Springer International Publishing.

Lebboss, G., Bernard, G., Aliane, N., and Hajjar, M. (2017). Towards the enrichment of Arabic WordNet with big corpora. In *IJCCI*, pages 101–109.

Levy, O. and Goldberg, Y. (2014). Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308.

Levy, O., Goldberg, Y., and Dagan, I. (2015a). Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Levy, O., Remus, S., Biemann, C., and Dagan, I. (2015b). Do supervised distributional methods really learn lexical inference relations? In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 970–976.

Lin, D. (1998). Automatic retrieval and clustering of similar words. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2*, pages 768–774.

Lin, D., Zhao, S., Qin, L., and Zhou, M. (2003). Identifying synonyms among distributionally similar words. In *IJCAI*, volume 3, pages 1492–1493. Citeseer.

Liu, Q., Jiang, H., Wei, S., Ling, Z.-H., and Hu, Y. (2015). Learning semantic word embeddings based on ordinal knowledge constraints. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1501–1511.

Lund, K., Burgess, C., and Atchley, R. A. (1995). Semantic and associative priming in high-dimensional semantic space. In *Proceedings of the 17th annual conference of the Cognitive Science Society*, volume 17, pages 660–665.

Melka, J. and Mariage, J.-J. (2017). Efficient implementation of self-organizing map for sparse input data. In *IJCCI*, pages 54–63.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Yih, W.-t., and Zweig, G. (2013b). Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 746–751.

Miller, G. A. (1957). Some effects of intermittent silence. *The American Journal of Psychology*, 70(2):311.

Mnih, A. and Hinton, G. E. (2008). A scalable hierarchical distributed language model. *Advances in neural information processing systems*, 21:1081–1088.

Mnih, A. and Kavukcuoglu, K. (2013). Learning word embeddings efficiently with noise-contrastive estimation. *Advances in neural information processing systems*, 26:2265–2273.

Mrkšić, N., Séaghdha, D. O., Thomson, B., Gašić, M., Rojas-Barahona, L., Su, P.-H., Vandyke, D., Wen, T.-H., and Young, S. (2016). Counter-fitting word vectors to linguistic constraints. *arXiv preprint arXiv:1603.00892*.

Murphy, B., Talukdar, P., and Mitchell, T. (2012). Learning effective and interpretable semantic models using non-negative sparse embedding. In *Proceedings of COLING 2012*, pages 1933–1950.

Nozza, D., Fersini, E., and Messina, E. (2016). Unsupervised irony detection: a probabilistic model with word

embeddings. In *International Conference on Knowledge Discovery and Information Retrieval*, volume 2, pages 68–76. SCITEPRESS.

Padó, S. and Lapata, M. (2007). Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2):161–199.

Pantel, P. and Ravichandran, D. (2004). Automatically labeling semantic classes. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 321–328.

Pasca, M. (2004). Acquisition of categorized named entities for web search. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 137–145.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Phillips, W. and Riloff, E. (2002). Exploiting strong syntactic heuristics and co-training to learn semantic lexicons. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 125–132.

Riloff, E., Jones, R., et al. (1999). Learning dictionaries for information extraction by multi-level bootstrapping. In *AAAI/IAAI*, pages 474–479.

Riloff, E. and Shepherd, J. (1997). A corpus-based approach for building semantic lexicons. *arXiv preprint cmp-lg/9706013*.

Ritter, A., Soderland, S., and Etzioni, O. (2009). What is this, anyway: Automatic hypernym discovery. In *AAAI Spring Symposium: Learning by Reading and Learning to Read*, pages 88–93.

Roark, B. and Charniak, E. (1998). Noun-phrase co-occurrence statistics for semi-automatic semantic lexicon construction. In *COLING 1998 Volume 2: The 17th International Conference on Computational Linguistics*.

Rubinstein, D., Levi, E., Schwartz, R., and Rappoport, A. (2015). How well do distributional models capture different types of semantic knowledge? In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 726–730.

Rydin, S. (2002). Building a hyponymy lexicon with hierarchical structure. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition*, pages 26–33.

Salton, G. (1971). The smart system. *Retrieval Results and Future Plans*.

Sang, E. T. K. (2007). Extracting hypernym pairs from the web. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 165–168.

Schutze, H., Manning, C. D., and Raghavan, P. (2008). *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge.

Shinzato, K. and Torisawa, K. (2004). Acquiring hyponymy relations from web documents. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 73–80.

Sombatsrisomboon, R., Matsuo, Y., and Ishizuka, M. (2003). Acquisition of hypernyms and hyponyms from the www. In *Proceedings of the 2nd International Workshop on Active Mining*.

Sumida, A. and Torisawa, K. (2008). Hacking wikipedia for hyponymy relation acquisition. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II*.

Tanev, H. and Magnini, B. (2006). Weakly supervised approaches for ontology population. In *11th Conference of the European Chapter of the Association for Computational Linguistics*.

Thelen, M. and Riloff, E. (2002). A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *Proceedings of the 2002 conference on empirical methods in natural language processing (EMNLP 2002)*, pages 214–221.

Tjong Kim Sang, E. and Hofmann, K. (2007). Automatic extraction of dutch hypernym-hyponym pairs. *LOT Occasional Series*, 7:163–174.

Tsur, O., Davidov, D., and Rappoport, A. (2010). Icwsm—a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 4.

Turney, P. D. (2008a). The latent relation mapping engine: Algorithm and experiments. *Journal of Artificial Intelligence Research*, 33:615–655.

Turney, P. D. (2008b). A uniform approach to analogies, synonyms, antonyms, and associations. *arXiv preprint arXiv:0809.0124*.

Turney, P. D. and Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141–188.

Vossen, P. (1998). A multilingual database with lexical semantic networks. *Dordrecht: Kluwer Academic Publishers. doi*, 10:978–94.

Widdows, D. and Dorow, B. (2002). A graph model for unsupervised lexical acquisition. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Wong, S. M., Ziarko, W., and Wong, P. C. (1985). Generalized vector spaces model in information retrieval. In *Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 18–25.

Wu, Z. and Palmer, M. (1994). Verb semantics and lexical selection. *arXiv preprint cmp-lg/9406033*.

Yatbaz, M. A., Sert, E., and Yuret, D. (2012). Learning syntactic categories using paradigmatic representations of word context. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 940–951.