ISSN 2007-9737

Reasoning over Arabic WordNet Relations with Neural Tensor Network

Mohamed Ali Batita[1], Rami Ayadi[2,3], Mounir Zrigui[1]

[1] University of Monastir, Faculty of Sciences,
Science Department, Monastir,
Tunisia

[2] University of Gabes,
Higher Institute of Computer of Medenine,
Tunisia

[3] Jouf University, College of Science and Arts in Al Qurayyat,
Computer Science Department,
Kingdom of Saudi Arabia (KSA)

batitamohamedali@gmail.com, rayadi@ju.edu.sa, mounir.zrigui@fsm.rnu.tn

Abstract. Arabic WordNet is an important resource for many tasks of natural language processing. However, it suffers from many problems. In this paper, we address the problem of the unseen relationships between words in Arabic WordNet. More precisely, we focus on the ability for new relationships to be learned 'automatically' in Arabic WordNet from existing relationships. Using the Neural Tensor Network, we investigate how it can be an advantageous technique to fill the relationship gaps between Arabic WordNet words. With minimum resources, this model delivers meaningful results. The critical component is how to represent the entities of Arabic WordNet. For that, we use AraVec, a set of pre-trained distributed word representation for the Arabic language. We show how much it helps to use these vectors for initialization. We evaluated the model, using a number of tests which reveal that semantically-initialized vectors provide considerable greater accuracy than randomly initialized ones.

Keywords. Arabic WordNet, natural language processing, neural tensor network, AraVec, word representation, word embedding.

1 Introduction

Arabic WordNet (AWN) [2, 12, 5, 6] is a lexical database for the Arabic language. It has been developed following the development process of Princeton WordNet (WN) [15, 24, 25] and Euro WordNet [36]. There is a degree of uncertainty around the term *wordnet*. It was created as a lexical database, however, some researchers refer to it as a semantic network because of its hierarchy in the way entities are seen as nodes and connected with edges [16]. Others define it as an ontology regarding some specific relations between conceptual categories [9]. However, these denotations cannot disguise the fact that many wordnets represent the largest publicly available lexical resource for many language. AWN is one of them. It is widely used in various fields of Natural Language Processing (NLP). Therefore, a considerable volume of research has been published to improve its content, in term of quantity [1, 3, 29] and quality [2, 9]. Along with this growth, however, there is increasing concern over the incompleteness and lack of the relationships in AWN.

These missing inter-relationships should be able to be addressed through the development of auto-reasoning within AWN. Auto-reasoning is most commonly seen in people, in what is usually called 'commonsense reasoning'. For

instance, if a new species of plant is discovered, we do not have to specify that it has leaves. That is learned 'commonsense' knowledge which people can impute from existing knowledge without reference to external resources, otherwise defined by Davis *et al.* [10] as a simulation of how humans think and react in their every-day situations. It is therefore essential to look for missing relationships to improve AWN.

Much of the current research is focused on enhancements of existing knowledge bases (KB), using patterns or classifiers applied to external corpus, regardless of the type of KB [39, 21, 37]. This approach has limitations however, given that not all the embodied knowledge in a text is explicitly presented, and particularly, in respect of the Arabic language, as it is a low-resource language.

Our major objective is to find a way to predict new relations from existing ones in the AWN. To do this, we embrace the Neural Tensor Network (NTN) proposed by [32] because it gave better results when they applied it to WN and Freebase [7]. Also, It uses only the database for the prediction, no external resources. NTN operates by transforming entities into vectors – one vector representing the distribution of an entity in the database and its relationship with others. Relationships between entities are then expressed using a group of parameters within the NTN. This word-embedding feature makes the model more accurate when compared with other models.

We have structured this article in six sections, including this introductory section. Section 2 will describe the AWN and what version are we going to work on. Related works will be detailed in section 3. Section 4 will describe the word embedding technique that we use and the NTN model. An outline of some of the tests used and the evaluation of their results is provided in section 5, with our conclusions following in section 6.

2 Overview of Arabic WordNet

Arabic WordNet is a large lexical database of Arabic, which is available for public use, free of charge[1]. It comprises 5 parts of speech: nouns, verbs, adverbs, adjectives, and adjective satellites, which are categorized into groups of synonyms called *synsets*. Each synset expresses a distinct sense or concept. They are interconnected by semantic and lexical relations. It is freely and publicly available for many tasks of NLP concerning the Arabic language. Currently, there are two versions of AWN. The version we are interested in is the LMF version. It is structured under the LMF standard, which makes it a practical tool for computational linguistics.

We find in the LMF Version[2] 60,154 entities in total. Most of them are verbs (42,298 entities) and nouns (16,432 entities). The rest are adverbs with 771, adjectives with 270, and adjective satellites with 386. With this variety, entities are connected with only 41,135 relations (5 type of relations). Our attention has been drawn to make progress in this version since it is rich in terms of entities but not in term of connection between them. First, we are going to cite some interesting works in this field.

3 Related Work

There is a large and growing body of literature which has been dedicated to an exposition of the incompleteness of existing KBs. The open-source KB, Freebase, before it closed, only held information regarding the nationality and birthplace of a small minority of those listed on it (25 and 29 percent, respectively) [11]. Studies of other KBs such as DBpedia [4] and YAGO [13] reveal that up to 99 percent of categorizes lack at least one property that others in the same class possess [34]. Galarraga *et al.* [17] discovered in 2016 that Wikidata knows the father of only 2 percent of all the people in it. For that, many attempts have been made to address this issue [14, 31]. Most of them used external resources in order to mine relevant rules such as if *X has children* then probably *X is married*. Thus, they can impute new facts and relations between entities. Use of externals resources is an important factor but this can pose problems for AWN, given the low-resource nature of the Arabic language when compared to other like English.

[1] http://compling.hss.ntu.edu.sg/omw/

[2] For the rest of the paper, AWN will refer to the LMF version.

There is nonetheless a significant minority of projects which have focused solely on improvements to a KB itself. In their major study, Nickel *et al.* [26] presented RESCAL, an approach based on the factorization of a three-way tensor. The main idea is to match the latent semantics of entities and relations, where a KB was regarded as a three-dimensional tensor. The matching score of a triplet is given by a bilinear function. Similarly, Jenatton *et al.* [20] proposed another tensor approach to model multi-relational datasets. It transformed entities into vectors and set a matrix for each relation. To reduce the number of parameters and avoid overfitting, relations were represented as a combination of latent factors. As Jenatton *et al.* observe, the bilinear structure of a latent factor model can reveal unseen shared similarities between different relation types.

The impact of tensor factorization has been explored in several studies of NLP. Setiawan *et al.* [30], He *et al.* [19], and Qiu *et al.* [28] proposed a different type of neural network that includes tensor decomposition. Sutskever *et al.* [35] proposed the Bayesian Clustered Tensor Factorization (BCFT). Their principal objective was to find interpretable structures within a KB and predict the accuracy of unobserved relationships within that KB. This model clusters entities and relations to share statistical strength through a three-way interaction using a bayesian non-parametric method. Socher *et al.* [32] found a way to predict new relations based on others. They developed the Neural Tensor Network (NTN), which is a tensor decomposition in a neural network architecture. It has been applied to WN and Freebase. Each relation in the two resources has its own tensor parameters and each entity is represented as an average of its constituent word vectors in high dimension.

Socher *et al.* also claim that representing entities with a single vector does not allow the sharing of statistical strength between similar entities. For instance, 'living room' and 'dining room' are two entities that have much in common, yet previous approaches are likely to embed each one separately. So, they embedded each of 'living', 'room', and 'dining' and then built the representations for entities as the average of the vectors of the three words. For that purpose, they initialized word vectors with pre-trained vectors using a word embedding model, demonstrating that this can increase the reasoning accuracy. In the same vein, Bordes *et al.* [8] proposed the Structured Embedding model. The main idea is that two entities of a correct triple should be closely related to each other in the relation space. Hence, two entities are defined by two (-head, and tail) relation-specific matrices and the score function correlates with the degree of relationship between the words.

While there is are many different models for word embedding, all of which work in their different ways, ultimately, in every model words will be represented as vectors in a continuous space. In 2013, Mikolov *et al.* [23] developed the famous word2vec, a model for word embedding with two different architectures, the continuous bag of words (CBOW), and the skip-gram (SKIP-G). The difference between them is that the first one learns to predict the word from its context and the second one learns to predict the context from the word. But, either way, they both represent the word in vector space according to its local context. CBOW is faster to train and works better with frequent words. SKIP-G performs well with a small training set and works with rare words.

A year later, Pennington *et al.* [27] presented an alternative model called GloVe. The major difference between GloVe and word2vec is that it leans by constructing a co-occurrence matrix: in other words how frequently a word appears in a context. In several cases, word2vec has proven to be a more successful algorithm for unsupervised learning of semantic similarity and relatedness, when compared with GloVe [22].

Recently, in 2016, a group of researchers on Facebook proposed fastText [18]. Rather than a word by word approach, like word2vec and GloVe, it breaks words into n-grams. For instance, the tri-grams for the word *fruit* is *fru*, *rui*, and *uit* and the vector of the word will be the sum of the n-grams. This approach provides a significant advantage in respect of rare words, and words that are not presented in the training set. The authors claim that this outperforms word2vec since it can present a vector for any word, something that neither

word2vec nor GloVe can do. However, research has consistently shown that both these models require large datasets for the training process.

In a major study concerning the Arabic language, Zahran *et al.* [38] built a vectorized word embedding for the Modern Standard Arabic (MSA) using CBOW, SKIP-G, and GloVe. They also evaluated them: intrinsically, using the task of standard word similarity: and extrinsically, using two NLP application, Information Retrieval, and Short Answer Grading.

Soliman *et al.* [33] presented the AraVec, a set of pre-trained distributed word embedding for the Arabic language. Its first version has six word embedding models trained on three datasets (Wikipedia, Twitter, and texts from Arabic web pages) with more than 3,300M tokens. The second version now contains 12 models. The main idea is to present each one of the datasets with the two models of word2vec (CBOW and SKIP-G). This outperformed fastText because it has been trained on a large dataset includes tweets (Arabic dialects).

The most obvious finding to emerge from these studies is that the tensor approach is very effective in predicting accurately missing information in linguistics resources. Also, this has enhanced our understanding of representing entities as feature vectors to preserve the knowledge of the original data and present the interesting ability of generalizing new reasonable relations.

4 Proposed Model

In this section, we focus on how to complete missing relationships between words in AWN. As shown previously, the neural tensor network (NTN), alongside word embedding techniques, arise naturally with the representation of multi-relational data, like AWN. For that, first, we prepare all the entities in AWN and then build the model by training it and adjusting its parameters.

4.1 Word Embedding

Word embedding, as a process, is learned from data. There are two ways to obtain the word embedding. The first method takes random vectors and trains the model until the training loss function can decrease no further and the second takes pre-trained vectors. The high volume of data input required of the first model meant that the second one is better suited to our research, given the low-resource nature of Arabic.

Many previous cited works have similarly relied on the advantage of using pre-trained word embedding vectors in their works, due to many factors. The rationale behind this is it is either because of the insufficient data available to learn truly powerful features (less-resourced language like the Arabic) or the disposition of powerful materials required for the calculation (training word2vec with a normal laptop could take hours). Besides, using a pre-trained word embedding vectors shorten the training time and make a better quality of word vectors. Another cause specific to our goal is that the strength shared between the words in a corpus is stronger than the strength shared between them in AWN.

There are several of pre-trained word embedding datasets with different models at the disposal of the NLP community. Our research used AraVec developed by Soliman *et al.* [33]. It is a set of word embedding models with 100 and 300 dimensions. It has been trained on different datasets. We combined the model CBOW that has been trained on Wikipedia and Web. Entities in AWN are either single or multi-word expression. AraVec made this task easiest because the vector of an expression is represented by the average of its word vectors. Once we indexed each word in AWN with its appropriate vector, we use them to train the NTN.

4.2 Neural Tensor Network

We adopted the NTN model provided by Socher *et al.* [32]. Not only has NTN performed well when used with NW and Freebase, it also outperforms other neural networks, such as the single and bilinear networks in tests, due to its ability to link entities across multiple dimensions.

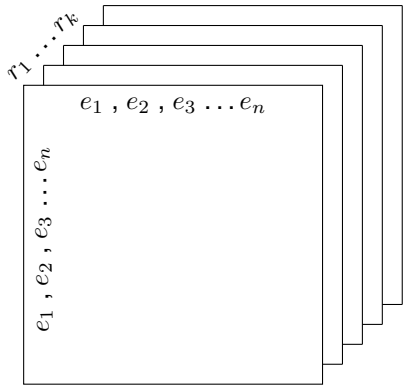

Fig. 1. Illustration of the tensor model

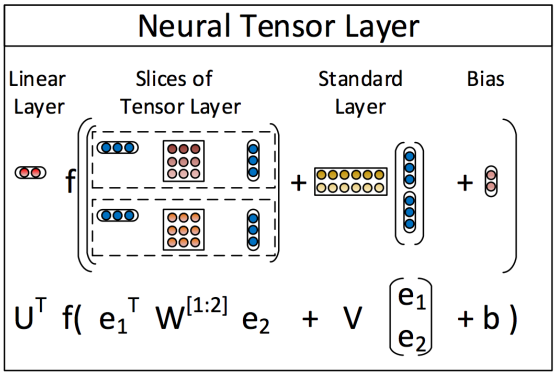

Fig. 2. Neural tensor Layer parameters with 2 slices

As we can see, figure 1 represent entities e_n and relations r_k as triples. This is called a three-way tensor. We can tell that a *slice* of the tensor is defined by a matrix when r is fixed. If we have two entities e_1 and e_2 with a relation r, the score of their relationship is a function $s(e_1, r, e_2)$:

$$s(e_1, r, e_2) = u_r^T f(e_1^T W_r^{[1:k]} e_2 + V_r, \begin{bmatrix} e_1 \\ e_2, \end{bmatrix} + b_r), \quad (1)$$

F is a $\tanh$ nonlinear function. $e_1^T W_r^{[1:k]} e_2$ is a tensor product where $W^{[1:k]}$ is the tensor and k represent the slices of the tensor. U_r, V_r, and b_r are the parameters of a standard neural network. The non-linearity of this model is the main advantage because it joints the entities directly through multiplicity. Each relation r has its own parameters as it is illustrated in the figure 1. A visualization of the tensor layer is shown in figure 2 provided in [32].

The model needs to be trained. This yield to adjust all the parameters U, E, W, V, and b by minimizing the hinge-loss function 2. The main idea is to give a higher score to a positive triple $(e_1^{(i)}, r^{(i)}, e_2^{(i)})$ than a negative triple $(e_1^{(i)}, r^{(i)}, e_c)$, in which e_c is a random entity:

$$J(\Omega) = \sum_{i=1}^{N} \sum_{c=1}^{C} \max(0, 1 - s(T^{(i)}) + s(T_c^{(i)})) + \lambda \|\Omega\|_2^2. \quad (2)$$

With Ω represents all the parameters of the network. 1 is the margin and s is the score. N and C are the set of positive and negative samples, respectively. $T^{(i)}$ and $T_c^{(i)}$ are the positive (correct) and negative triple, respectively. λ is the regularization parameter to prevent overfitting. To update these parameters and calculate $\partial J / \partial \Omega$, we use the backpropagation algorithm.

5 Test and Evaluation

The concept is easier to understand using an example. If [3] (Bosporus) is a (strait) and a is a (canal) then is a too. This is simply how the model will predict new relations in a transitive way. To make the semantic and lexical sharing stronger, entities should be presented with as many relations as possible. For that, we keep only the accurate ones that are presented with more than 2 relations. In total, we have 41,122 connected triples with 5 type of relations. We did not work with the *antonym* relation, and we concatenated the *hyponym/hypernym* (is a) and *HasInstance/isInstance* (instance) as one relation, respectively. We end up with 3 types of relations and we split the data as showed in table 1.

We do not have a large data to be able to train the model perfectly, so we used the k-fold cross validation (we choose $k = 10$) to better entertain the training and the test sets. For each k, we combine the training and validation sets and run

[3] Arabic words are followed by their transliteration using the transliteration system of LaTeX and their English translations in brackets.

Table 1. Statistics of the AWN preparation

Database	Relations	Training set	Validation set	Test set
Arabic WordNet	3	28,272	2,570	10,280

the algorithm once again. Also, we notice that many words do not have any relations at all. So, we created manual associations with at least one relation between a word and a triple and had that verified by a lexicographer. To add more precision to the model and to force it to focus on harder cases, we add some of them to the validation and the rest to the test set, since the main goal is to predict new relations. Negative examples are created randomly by switching one entity in a correct triple in the training set.

Another matter that needs attention is the complexity of the model. The complexity is represented by the number of the parameters within the tensor that need to be trained. It lies exactly in the tensor. To reduce complexity in the model, we choose to work with only 3 slices of tensor. For instance, if the dimension of the entities vectors is $d = 100$ ($e_i \in \mathbb{R}^{100}$) then the tensor $W_R^{[1:k]} \in \mathbb{R}^{d*d*k}$ ($10000 * k$) and the standard parameter $V_R \in \mathbb{R}^{2*d}$ ($2d * k$). In the end, with 3 slices, we have over 30,000 parameters to train. For that, we could not add a new slice otherwise we have to add new relations between all the entities.

In general, the accuracy of the model is based on how correct new triples are.

Table 2. Recall and precision of different relations in AWN

Relations	Recall (%)	Precision (%)
Is a	29.3	71.8
Instance	32.7	45.7
Similar	13.7	32.5

The disparity as shown in table 2 lies in the difference between the relations in terms of the number of triples in the training set. There is also the complexity of the relation. For instance, it is easier to reason for example over the *is a* relation than the *instance*. Furthermore, the vector representation takes part of this disparity.

Table 3 show how much the initialization with pre-trained vectors help.

Table 3. Variance between different entities representations

Initialization	Accuracy (%)
Randomly	52
Pre-trained with 100 dimensions	**71**
Pre-trained with 300 dimensions	68

We tested the model with a random initialization and the two sets of AraVec: 100 and 300 dimensions. Both of the distribution of AraVec gave almost the same results but the random initialization showed lower accuracy. This due to the multi-word expressions presented in the AWN. We considered by presenting the multi-word expressions as the average of the words that combine them it will strengthen the semantic sharing between them. We can explain the low accuracy by the lack of the triples in the training set. Most importantly, this model can reason without recourse to external resources.

6 Conclusion

Our research addressed the incompleteness of the Arabic WordNet using the natural tensor network as a tool. Results achieved using the NTN have improved through the incorporation of pre-trained word vectors. The intersection between relations made via a tensor layer. The study has demonstrated meaningful results extending the number of relations within Arabic WordNet. Such an automated process inevitably brings with it the risk of adding wrong information. Further research will be focused on error detection in the Arabic WordNet and an investigation of the quality of its information.

References

1. **Abouenour, L., Bouzoubaa, K., & Rosso, P. (2010).** Using the yago ontology as a resource for the enrichment of named entities in arabic

wordnet. *Proceedings of The Seventh International Conference on Language Resources and Evaluation (LREC 2010) Workshop on Language Resources and Human Language Technology for Semitic Languages*, pp. 27–31.

2. **Abouenour, L., Bouzoubaa, K., & Rosso, P. (2013).** On the evaluation and improvement of arabic wordnet coverage and usability. *Language resources and evaluation*, pp. 891–917.
3. **Alkhalifa, M. & Rodríguez, H. (2010).** Automatically extending named entities coverage of arabic wordnet using wikipedia. *International Journal on Information and Communication Technologies*, pp. 20–36.
4. **Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., & Ives, Z. G. (2007).** Dbpedia: A nucleus for a web of open data. *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007.*, pp. 722–735.
5. **Black, W., Elkateb, S., Rodriguez, H., Alkhalifa, M., Vossen, P., Pease, A., & Fellbaum, C. (2006).** Introducing the arabic wordnet project. *Proceedings of the third international WordNet conference*, pp. 295–300.
6. **Black, W., Sabri, E., Horacio, R., Musa, A., Piek, V., Adam, P., , & Christiane, F. (2006).** Introducing the arabic wordnet project. *In Proceedings of the third international WordNet conference*.
7. **Bollacker, K., Evans, C., Paritosh, P., Sturge, T., & Taylor, J. (2008).** Freebase: A collaboratively created graph database for structuring human knowledge. *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pp. 1247–1250.
8. **Bordes, A., Weston, J., Collobert, R., Bengio, Y., et al. (2011).** Learning structured embeddings of knowledge bases. *AAAI*, pp. 6.
9. **Boudabous, M. M., Kammoun, N. C., Khedher, N., Belguith, L. H., & Sadat, F. (2013).** Arabic wordnet semantic relations enrichment through morpho-lexical patterns. *Communications, Signal Processing, and their Applications (ICCSPA), 2013 1st International Conference on*, pp. 1–6.
10. **Davis, E. & Marcus, G. (2015).** Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, pp. 92–103.
11. **Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., & Zhang, W. (2014).** Knowledge vault: A web-scale approach to probabilistic knowledge fusion. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 601–610.
12. **Elkateb, S., Black, W., Rodríguez, H., Alkhalifa, M., Vossen, P., Pease, A., & Fellbaum, C. (2006).** Building a wordnet for arabic. *Proceedings of The fifth international conference on Language Resources and Evaluation (LREC 2006)*, pp. 22–28.
13. **Fabian, M., Gjergji, K., Gerhard, W., et al. (2007).** Yago: A core of semantic knowledge unifying wordnet and wikipedia. *16th International World Wide Web Conference, WWW*, pp. 697–706.
14. **Fader, A., Soderland, S., & Etzioni, O. (2011).** Identifying relations for open information extraction. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 1535–1545.
15. **Fellbaum, C. (1998).** *WordNet*. Wiley Online Library.
16. **Fontenelle, T. (2012).** Wordnet, framenet and other semantic networks in the international journal of lexicography – the net result? *International Journal of Lexicography*, pp. 437–449.
17. **Galárraga, L., Razniewski, S., Amarilli, A., & Suchanek, F. M. (2017).** Predicting completeness in knowledge bases. *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 375–383.
18. **Grave, E., Mikolov, T., Joulin, A., & Bojanowski, P. (2017).** Bag of tricks for efficient text classification. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 427–431.
19. **He, X., Xu, H., Sun, X., Deng, J., & Li, J. (2017).** Abircnn with neural tensor network for answer selection. *Proc. Int. Joint Conf. Neural Networks (IJCNN)*, pp. 2582–2589.
20. **Jenatton, R., Roux, N. L., Bordes, A., & Obozinski, G. R. (2012).** A latent factor model

for highly multi-relational data. *Advances in Neural Information Processing Systems*, pp. 3167–3175.

21. **Kaushik, N. & Chatterjee, N. (2016).** A practical approach for term and relationship extraction for automatic ontology creation from agricultural text. *Proc. Int. Conf. Information Technology (ICIT)*, pp. 241–247.
22. **Major, V., Surkis, A., & Aphinyanaphongs, Y. (2017).** Utility of general and specific word embeddings for classifying translational stages of research. *arXiv preprint arXiv:1705.06262*.
23. **Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013).** Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
24. **Miller, G. A. (1995).** Wordnet: a lexical database for english. *Communications of the ACM*, pp. 39–41.
25. **Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., & Miller, K. J. (1990).** Introduction to wordnet: An on-line lexical database. *International journal of lexicography*, pp. 235–244.
26. **Nickel, M., Tresp, V., & Kriegel, H.-P. (2011).** A three-way model for collective learning on multi-relational data. *ICML*, pp. 809–816.
27. **Pennington, J., Socher, R., & Manning, C. (2014).** Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.
28. **Qiu, X. & Huang, X. (2015).** Convolutional neural tensor network architecture for community-based question answering. *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pp. 1305–1311.
29. **Rodríguez, H., Farwell, D., Ferreres, J., Bertran, M., Alkhalifa, M., & Martí, M. A. (2008).** Arabic wordnet: Semi-automatic extensions using bayesian inference. *LREC*.
30. **Setiawan, H., Huang, Z., Devlin, J., Lamar, T., Zbib, R., Schwartz, R., & Makhoul, J. (2015).** Statistical machine translation features with multitask tensor networks. *arXiv preprint arXiv:1506.00698*.
31. **Snow, R., Jurafsky, D., & Ng, A. Y. (2004).** Learning syntactic patterns for automatic hypernym discovery. *Advances in Neural Information Processing Systems 17 [Neural Information Processing Systems, NIPS 2004, December 13-18, 2004, Vancouver, British Columbia, Canada]*, pp. 1297–1304.
32. **Socher, R., Chen, D., Manning, C. D., & Ng, A. Y. (2013).** Reasoning with neural tensor networks for knowledge base completion. *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pp. 926–934.
33. **Soliman, A. B., Eissa, K., & El-Beltagy, S. R. (2017).** Aravec: A set of arabic word embedding models for use in arabic nlp. *Procedia Computer Science*, pp. 256–265.
34. **Suchanek, F. M., Gross-Amblard, D., & Abiteboul, S. (2011).** Watermarking for ontologies. *International Semantic Web Conference*, pp. 697–713.
35. **Sutskever, I., Salakhutdinov, R., & Tenenbaum, J. B. (2009).** Modelling relational data using bayesian clustered tensor factorization. *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada.*, pp. 1821–1828.
36. **Vossen, P. (1998).** *EuroWordNet: A multilingual database with lexical semantic networks*. Kluwer Academic Publishers.
37. **Wang, M., Li, L., & Huang, F. (2014).** Semi-supervised chinese open entity relation extraction. *Proc. IEEE 3rd Int. Conf. Cloud Computing and Intelligence Systems*, pp. 415–420.
38. **Zahran, M. A., Magooda, A., Mahgoub, A. Y., Raafat, H., Rashwan, M., & Atyia, A. (2015).** Word representations in vector space and their applications for arabic. *International Conference on Intelligent Text Processing and Computational Linguistics*, pp. 430–443.
39. **Zhuang, T., Wang, P., Zhang, Y., & Zhang, Z. (2017).** A novel method for open relation extraction from public announcements of chinese listed companies. *Proc. Fifth Int. Conf. Advanced Cloud and Big Data (CBD)*, pp. 200–205.

Article received on 17/01/2019; accepted on 04/03/2019.
Corresponding author is Mohamed Ali Batita.