Word Embedding using SkipGram and GLoVE Techniques

XXXXXXX[a]

[a]*Department of Engineering and Information Technology, Ajman, Ajman University United Arab Emirates*

Abstract

Word embedding aims to represent language semantics, syntax and pragmatics in a dense representation. Which is essential part of Natural Language Processing(NLP), extracting meaningful information from sentences, and vectorizing sentences to be processed nlp mathematical computations. This study conducts experiments on Word2Vec and GLoVE which are co-occurrence based word embedding models using the Arabic wiki dump 2018. The experiments demonstrates the fundamental differences between the two models operation as they differ in the principle and computation of similarity in terms is reflected in the execution time and memory consumption.

Keywords: Artificial Intelligence, Natural Language Processing , Word Embedding, Machine Learning, SkipGram, Word2Vec, GLoVE

1. Introduction

Ever since Humankind lived on the planet, Communication is an essential element for the survival of population. Communication has many forms namely, gestures, vocal it can be direct and indirect. As humans evolved over time, they explored and claimed grounds of this earth. Along with the evolution of life and cognitive abilities, communication language among humans evolved to several languages relative to the geographical location. Effective communication using a specific language requires the comprehension of that specific language in terms of syntax, semantics and pragmatics[1]. The syntax of a language is the structure and rules of constructing sentences which some papers such as Gildea and Jurafsky [2] suggests is a prerequisite for any semantic role labeling. Good semantics skills manifest in meaningful sentences, expressions, in which pragmatics indicates the relevance of a sentence/expression in a context of conversations, intuitive human rules and its ability to correctly use the language to deliver a message or convey a meaning.

The objective of Natural Language Processing(NLP) is allowing machines to understand and comprehend a language in similar manners to humans.

This report is organized as follows, section 2 is dedicated to demonstrate two different methodologies of word embedding, namely Word2Vec in subsection2.1 and GLoVE in subsection2.2. Followed by series of experiments of these tow methodologies on the Arabic language in section3. Finally, in section4 conclusion of our work is stated with several suggestions for the upcoming work.

2. background

Word Representations dates to the early 1986, tackling the encoding of linguistic regularities and patterns [3]. The main characteristics that differentiates co-occurrence based word representation from predictive models, is that these models are trained over the full corpus to capture dependencies and context[3]. Unlike co-occurrence based word representation, predictive models are trained over subset of the corpus, therefore capturing only local dependencies. The most popular co-occurrence based models are Word2Vec and GLoVE, due to their ability to capture high quality yet dense word representation from corpus.

2.1. Word2Vec Word Embedding

The study conducted in Mikolov et al. [4] proposes the use of unsupervised artificial neural network(ANN) to build a word embedding vector for large-sized corpus. The neural networks consists of three layers namely input, hidden and output where the number of neurons in each layer depends on the type of word2vec embedding type. The main three types of word2vec are continuous bag of words(CBOW), SkipGram(SG), and SkipGram with negative sampling(SGNS). Each word in the corpus at a time is captured as center word where as its neighbor word are captured within a distance (window) are called context words. At the first layer each word is fed to the neural network as one-hot-encoding vector and at last layer, the softmax function is computing the probability of the outputs. The CBOW and SG are opposite of each other where CBOW's aim is to predict the center words using the context words as input. On the other hand, SG predicts the context words of a center word. As a results CBOW operation is much faster than SG however SG performs better in case of infrequent words whereas CBOW performs well for frequent words. Additionally negative sampling is often used with skipgram as can be deduced from Fig1, the structure is large which slows down the performance and its performance against frequent words. The authors of the SkipGram proposed negative sample to reduce the effect of more frequent words by selecting it as negative samples. as result, each training sample of the negative sampling words updates a tiny percentage of the model's weights.

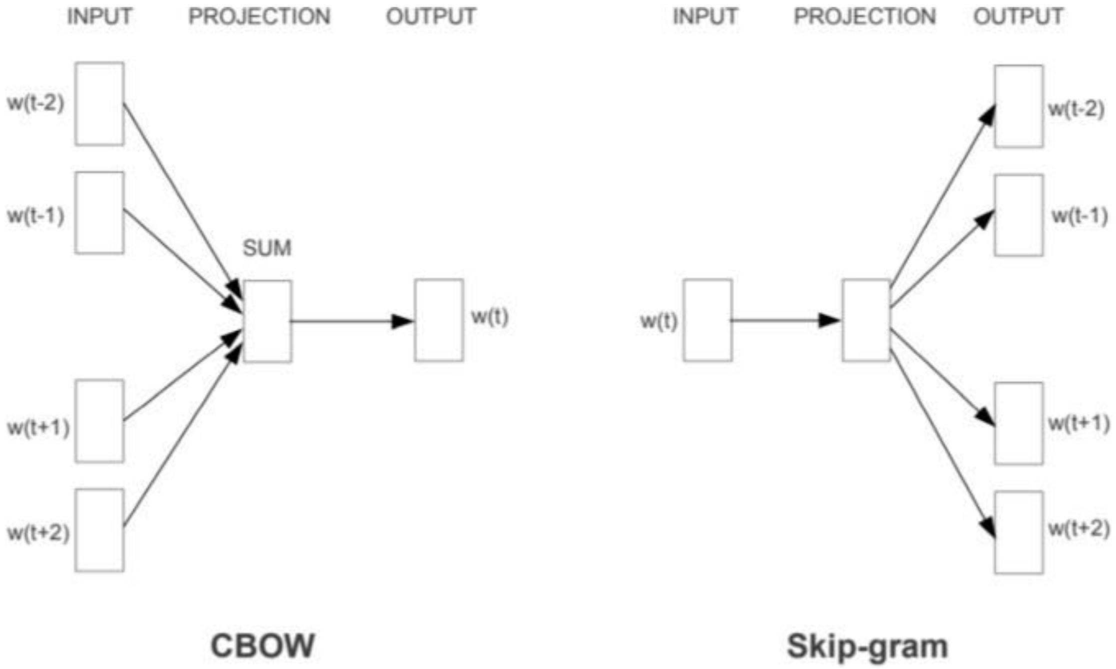


Figure 1: The shallow neural network structure of SkipGram and continuous bag of words

2.2. GLoVE Word Embedding

In 2013, Mikolov et al. [4] paper presented an unsupervised vector representation of words called Global Vectors for word representation(GLoVE). The relationship between words are deduced from the word-to-word co-occurrence probability which is computed as depicted in Fig3. The GLoVe implementation as in Fig2 does not associate a single word with a number rather draws the relation between pair of words therefore capture more semantic information than word2vec.

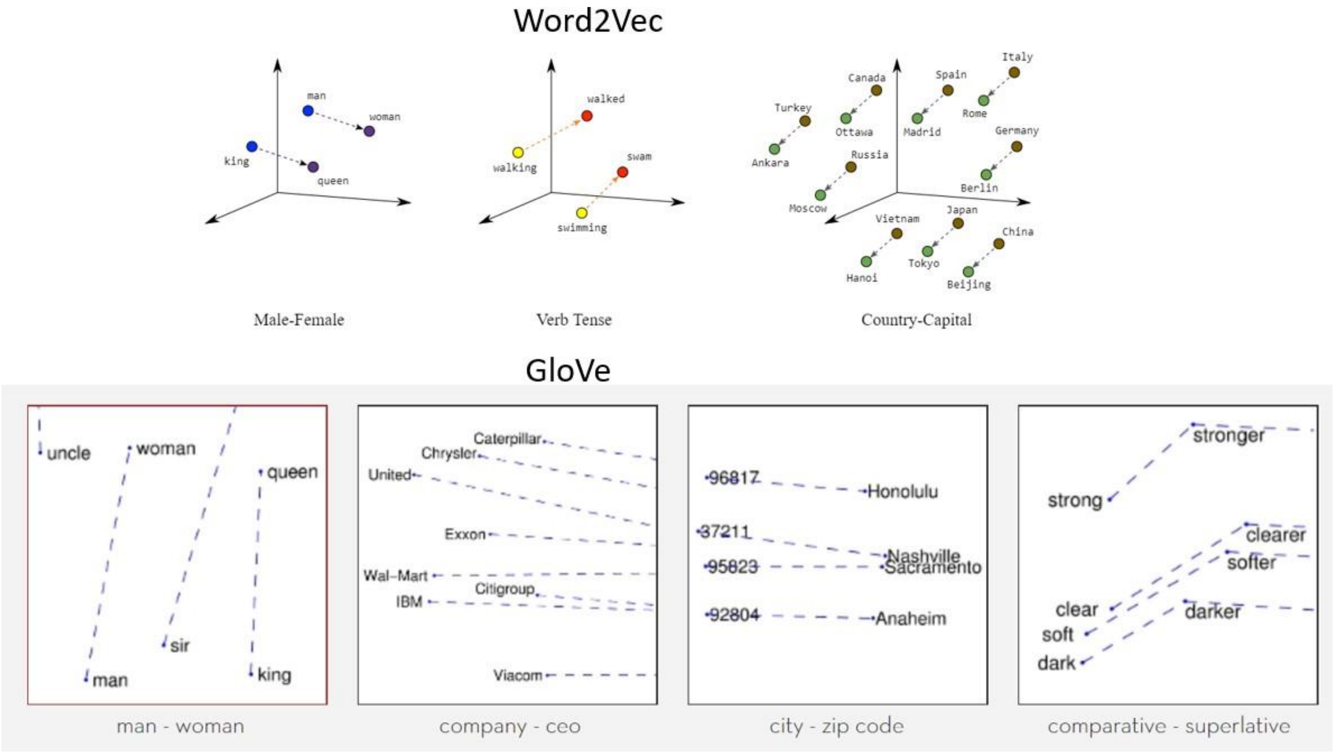
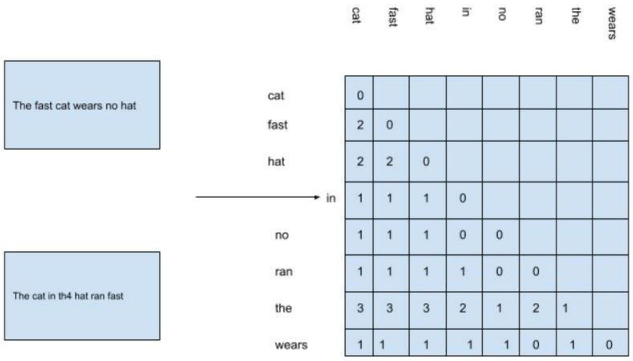


Figure 2: GLoVE Word Vectorization vs Word2Vec Vectorization



The fast cat wears no hat

The cat in th4 hat ran fast

	cat	fast	hat	in	no	ran	the	wears
cat	0							
fast	2	0						
hat	2	2	0					
in	1	1	1	0				
no	1	1	1	0	0			
ran	1	1	1	1	0	0		
the	3	3	3	2	1	2	1	
wears	1	1	1	1	1	0	1	0

Figure 3: Co-Occurrence Matrix between 2 words

3. Simulations and Experiments

The following experiments are Training SkipGram and GLoVE word embedding models on the Formal Arabic language(*Fus-ha*) benchmarked by arabic wikidump 2018 multi-stream with article based pages in arabic language hosted on Kaggle website and on wikidump regularly-updated webpage of archived wiki data dumps. Furthermore, SkipGram is implemented using GenSim ver 4 while GLoVE is implemented using glove-python-binary, written in python programming language version 3.10 on Anaconda - Jupyter Notebook and google colab. Parsing the data from bz2 format to corpus is done using GenSim's corpuswiki which is used to construct a corpus from a Wikipedia database dump. In addition, all experiments in this report are executed on HP-Omen-15 Laptop with core i7 CPU, 16 GB of RAM, and 64-bit Windows 11 operating system.

Parameter control and Tuning

Skip gram and GLoVE word embedding have a parameter of window size that must be tuned as it heavily depends on the training corpus complexity and size. As common knowledge in the NLP research community the window size starts from 5, therefore we have tried 10,15,20 on SkipGram and on GLoVE we tired 10,15. Another parameter dependent on the training corpus, is the embedding matrix size which is tested as 500, and 1000. Unfortunately, (as expected) the value 1000 generated a memory error as the environment memory is unable to allocate the enough space to run either of the algorithms therefore is fixed to 500. Lastly some parameters are dedicated to GLoVE are also experimented with such as the learning rate 0.01 and 0.05 while epochs parameter is fixed to 50 to avoid extensive runtime. It worth mentioning, the experiments are tested on 3 variations of SkipGram model and 4 varations of GLoVE model and the results discussed are chosen from the full results which are available and can be tested using the provided code.

3.1. Results and Discussion

3.1.1. Most Similar Words Test

The first performance measure for a word embedding is to request from the word embedding model the topmost similar words to a word (ranked according to the probability) according to the word dictionary. In such, the test retrieved a reasonable result. The SkipGram model with the largest window size did slightly better than the rest as can be seen in Table 1

Table 1: results of most similar words test

	SkipGram				**GLoVE**			
	model 1		**model3**		**model 1**		**model4**	
	word	**probability**	**word**	**probability**	**word**	**probability**	**word**	**probability**
test 1 = مراجع	**وصلات**	**0.693913758**	**خارجية**	**0.7487398**	**وصلات**	**0.948014**	**وصلات**	**0.906035**
	المراجع	0.6707	وصلات	0.748647	فون	0.92654	خارجية	0.883696
	خارجية	0.6622	تصنيف	0.742483	مستوطنة	0.925813	أنهار	0.722843
	تصنيف	0.6364	المراجع	0.65884	الديني	0.925098	مواليد	0.70448
	أشخاص	0.601	أشخاص	0.63865	بمثابة	0.924796	دولية	0.694218
	روابط	0.5513	قيد	0.6068	سابق	0.92431	بحار	0.684722
	قيد	0.53644	الحياة	0.549145	الجدول	0.923955	عواصم	0.68312
	مصادر	0.5285	روابط	0.54232043	المقر	0.92383	أفريقية	0.6718
	انظر	0.4955	مصادر	0.533717	الجبال	0.923391	مستعمرات	0.670298
	المصادر	0.4906	انظر	0.52633	إعصار	0.923018	اجتماع	0.66711
test 2=الإسلامية	**الاسلامية**	**0.7152**	**الاسلامية**	**0.67836**	**العراق**	**0.99912**	**للثقافة**	**0.865543**
	الشريعة	0.6029	الإسلامي	0.676527	حرب	0.9991	المؤسسة	0.83938
	الإسلامي	0.5938	الشريعة	0.5850021	مسقط	0.99906	الفتوحات	0.838295
	الأسلامية	0.5534	الدولة	0.5784173	شرق	0.999026	الثورة	0.821365
	مبايعتها	0.5425	الإسلام	0.575439	موسكو	0.998972	الأميركية	0.815516
	بداعش	0.54113	إسلامية	0.559486	السياحة	0.9989	لرئاسة	0.81395
	داعش	0.530151	تنظيم	0.5369313	بحر	0.998946	القضايا	0.8108
	إسلامية	0.52253	داعش	0.5171026	العصر	0.9989	الهيئة	0.805389
	أحمديون	0.52	إسلامي	0.513713	الألماني	0.998941	المدن	0.805286
	بانغسامورو	0.5142	المسلمين	0.5134093	تأسيس	0.9989	النهضة	0.803
test 3=ابن	**وابن**	**0.72276914**	**وابن**	**0.6988572**	**خلدون**	**0.999733**	**خلدون**	**0.985505**
	لابن	0.61664116	لابن	0.640947998	العسكرية	0.9941	مقدمة	0.977738
	سمجون	0.597453	أبي	0.6371892	قطاع	0.994136	أربيان	0.971074
	الصفدي	0.59119	البغوى	0.5945927	قوات	0.99408	باناخ	0.940257

	السخاوي	0.59112	الهيثمي	0.5941	الرياضيات	0.994068	الهيثم	0.935648
	العسقلانى	0.58559	الآبنوسي	0.59092348	البحرية	0.994051	فلسفة	0.91468
	الطبرى	0.583175	البلقيني	0.5873	ضد	0.99403	مؤلف	0.86757
	الهيثمي	0.580761	الصائن	0.5771	أقدم	0.994037	سعود	0.838698
	بزيزة	0.57983	القتيبي	0.575	فرنسا	0.994017	عمه	0.783267
	ماكولا	0.5776532	للرضي	0.5726	الخوارزمي	0.993956	عباس	0.77757

As can be seen from the table, the test conveyed what each model considers a similar word, such that word2vec linked words within the same topic together such that مراجع,مصادر ,أشخاص which are represent in our life the "resources-مراجع " where all words retrieved by the model are very similar to the original word and even words like "look-انظر " is frequently appearing with the word مراجع, in the context of "don't forget/look at the resources" of a webpage/article. Also, we can observe that the collection of words retrieved by the two skipgram models are almost identical yet with different probability. Also, both models understood that different derivation (الإسلامي,الإسلام ,إسلامية) of the same word still yields that almost same meaning. In case the word derivations are not found in the corpus the nearest words to a word (i.e.ابن) according to Skipgram are the most occurring ones such as arabic scientist compound names starting ابن.

On the other hand, GLoVE model linked similarity to only on occurrence of words, such that the derivatives of words are rarely linked together due to its rare appearance together in the same sentence in Arabic. In the test3 (word=ابن) the model outputted the most similar words as the most occurring words with the given word (ابن) which are mainly compound men names and their characteristics such as فلسفة and مؤلف. The different parameters supplied to the 2 glove models highly influenced the performance of each where it can be seen, only the highest probability word is common while the rest of the nine most similar words are different yet exists within the same topic of the original word.

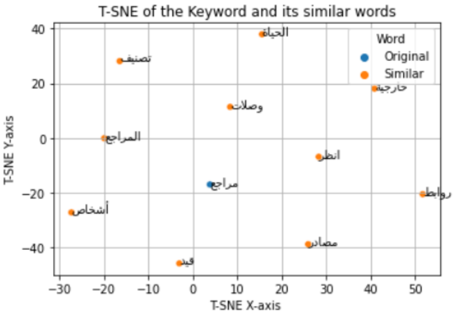


Figure 2: test1 T-SNE visualization using skipgram model with window size 20

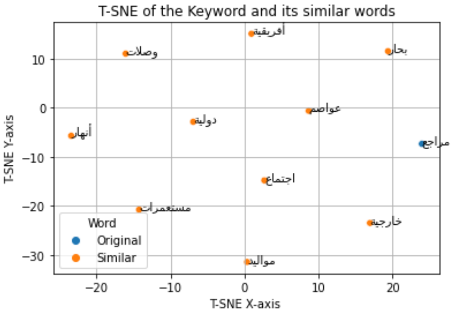


Figure1: test1 T-SNE visualization using GLoVE model , learning rate =0.01 and window size of 15

3.1.2. The Odd-Ones-Out Test

In this test the models are tested on its ability to distinguish not-fit words based on the different meanings of each word. The test not only measures the similarities and dis-similarities between different words but also shows the models ability to represent/store different meaning of a word and compare it against the rest of the words. Both models showed an excellent job in this task such as in Figure 3 and Figure 4 differentiating a month name from an object name such as "home - دار" and in Figure 5 and Figure 6 distinguish between the peace against words like war, military.

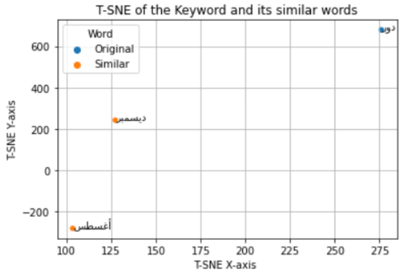


Figure 4: skipgram model(w=20)

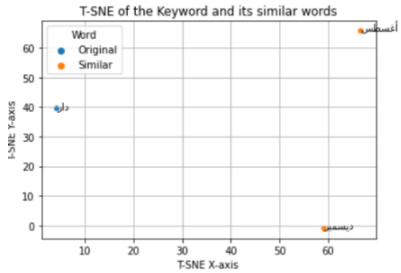


Figure 3: GLoVE model (lr=0.01, w=15)

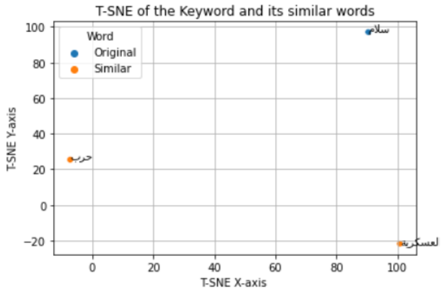


Figure 6: skipgram model(w=20)

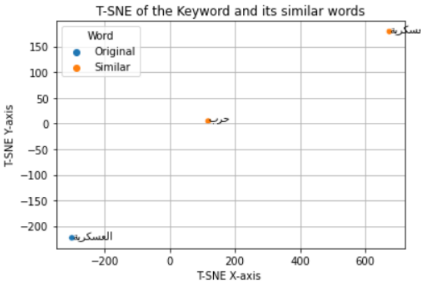


Figure 5: glove(lr=0.01, w=15)

3.1.3. Measuring Sentence Similarity Test

This test is the measurement of the vector representation on the scale of a full sentence instead of just word similarity comparison. The similarity of two sentences can be expressed as the distance between the 2 identical sentences as originally zero then based on alterations made in each sentence the distance will increase. The first alteration was exchanging a word direction to an opposite one which yielded in all models small and insignificant increase in distance (similarly in alteration 2). As well as in alteration5 (changing destination from home to school) which affected the distance in the skipgram models more than GLoVE. Alterations 3,4 are just in the meaning, keeping the structure of the sentence the same affected SkipGram more than it did to GLoVE model. GLoVE model conveyed more realistic results than Skipgram where distance is not drastically changed, only affected by major changes such as including words of different topic. While SkipGram is influenced by any alterations whether the new replacement is within the topic, opposite meaning and when the structure of the sentence is changed. However, GLoVE performance shown, is dependent on the parameter values while skipgram hold consistence regardless of the window size.

Table 2: similarity probability of 2 sentences where each alteration/s effect is observed and studied

	SkipGram		GLoVE	
	model1 (w=10)	model3 (w=20)	model1 (lr=0.05, w=10)	model4 (lr=0.01, w=20)
Alteration 1	0.08733	0.0878482	0.008125	0.022298
Alteration 2	0.2479238	0.2367178	0.038421	0.073875
Alteration 3	0.1679753	0.1625338	0.016505	0.047463
Alteration 4	0.3018217	0.28917	0.045964	0.108044
Alteration 5	0.1562023	0.1500819	0.002092	0.020931

3.1.4. Analogy Test

Last test “analogy” is frequently used in test new word embedding as it displays the model’s ability to conclude a relationship between a pair then emulate it by generating the second item of the second pair while maintaining the relation between the 2 pair of words. So, this test requires the model to deduce relationships between words (synonym, antonyms, within the topic) and generate suitable word for the second pair that maintains the 2-pairs relationship.

The first test the pair of months (يناير and مارس) with a pair of numbers, providing ثلاثة as the first element in the second pair. Skipgram models easily suggested 90% of the words as numbers while glove models provided words that are highly linked to number words such as “students”, “kilometer”.
The second test the first pair provided is (أمريكا and الأمريكية) indicating a derivative relationship of country-nationality so in the second pair , مصر is provided. GLoVE model outputted words like 'التاريخ' and 'الأمم' which are not very similar to the desired output as SkipGram did as the highest probability 'المصرية'. In addition to many tests conducted regarding the analogy, the skipgram showed better performance due its ability to understand derivatives unlike GLoVE which highly depends in its operation on the co-occurrence of the pair of words.

Table 3: analogy test results

	SkipGram		GLoVE	
	Model1	**Model3**	**Model1**	**Model4**
Test 1	[('أربعة', 0.802947998046875),	[('أربعة', 0.8120179176330566),	[('أربعة', 0.9984818696975708),	[('الطلاب', 0.8163548111915588),
	('خمسة', 0.777356743812561),	('خمسة', 0.7660980224609375),	('دون', 0.998397946357727),	('التبخر', 0.8125852942466736),
	('ستة', 0.7427753806114197),	('ستة', 0.7478825449943542),	('المواقع', 0.9983669519424438),	('كيلومترا', 0.8067097067832947),
	('سبعة', 0.7234185338020325),	('ثلاث', 0.700265109539032),	('نسمة', 0.9983471632003784),	('إضافة', 0.8018900156021118),
	('ثمانية', 0.7017174363136292),	('سبعة', 0.6996170282363892),	('بها', 0.9983181953430176),	('دقيقة', 0.798265099525451‍7),
	('ثلاث', 0.6849888563156128),	('ثمانية', 0.6553516387939453),	('الغرب', 0.9982529282569885),	('الوصول', 0.7929408550262451),
	('تسعة', 0.6687284111976624),	('تسعة', 0.6185495257377625),	('القبائل', 0.9982279539108276),	('انتقلوا', 0.7863315343856812),
	('أربع', 0.6121086478233337),	('أربع', 0.6114603877067566),	('مساحة', 0.9981310367584229),	('مصطفى', 0.785034656524658‍2),
	('وثلاثة', 0.6014035940170288),	('واحد', 0.5933309197425842),	('أي', 0.9981250762939453),	('الحوض', 0.7786349058151245),
	('وأربعة', 0.5499442219734192)]	('اثنين', 0.5726674199104309)]	('حوالي', 0.9980805516242981)]	('أغنية', 0.7779162526130676)]
Test 2	[('المصرية', 0.5245948433876038),	[('المصرية', 0.5877627730369568),	[('التاريخ', 0.9938900470733643),	[('والأمم', 0.8707531690597534),
	('القاهرة', 0.44952476024627686),	('القاهرة', 0.4777747690677643),	('العقبة', 0.9938133955001831),	('والمملكة', 0.8594424724578857),
	('بمصر', 0.4348808825016022),	('المصري', 0.474521666765213),	('صور', 0.9935498237609863),	('المتوكلية', 0.8297244310379028),
	('المصري', 0.423753947019577),	('العربية', 0.43108224868774414),	('الرئيس', 0.9932907819747925),	('الأمم', 0.8137847185134888),
	('لمصر', 0.41580915451049805),	('للمصريات', 0.4185960590839386),	('وسام', 0.9931195378303528),	('إلينوي', 0.8042823076248169),
	('ستشارا', 0.4060576856136322),	('بعسكرية', 0.4111025333404541),	('أيضا', 0.9930180311203003),	('العراقية', 0.8000868558883667),
	('لمفتشي', 0.4053516685962677),	('محمد', 0.41023173928260803),	('فإن', 0.9930170774459839),	('بالولايات', 0.7971137762069702),
	('ومندوبها', 0.40483376383781433),	('بمصر', 0.4095444977283478),	('السعودية', 0.9929596781730652),	('والولايات', 0.795896053314209),
	('بعسكرية', 0.4011751413345337),	('وقنصلياتها', 0.40688779950141907),	('ولكن', 0.9929119348526001),	('العربيه', 0.7949668169021606),
	('العرابى', 0.3983742892742157)]	('المتحدة', 0.4050217568874359)]	('منها', 0.992822229862213‍1)]	('برعاية', 0.7882643342018127)]
Test3	[('دمشق', 0.49183279275894165),	[('دمشق', 0.5070635676383972),	[('وقت', 0.9989140629768372),	[('قلب', 0.8341658115386963),
	('سورية', 0.47409504652023315),	('السورية', 0.4606870710849762),	('الوسطى', 0.998856067657470‍7),	('نيويورك', 0.8072588443756104),
	('درعا', 0.44870463013648987),	('سورية', 0.4527646005153656),	('وكذلك', 0.9988399744033813),	('سورية', 0.8068636655807495),
	('واشنطن', 0.43179112672805786),	('واشنطن', 0.4451574683189392),	('الزراعة', 0.9988201856613159),	('هلسنكي', 0.802869439125061),
	('اللاذقية', 0.42612236738204956),	('لبنان', 0.43993714451789856),	('جدا', 0.9988193511962891),	('قاسيون', 0.7995375394821167),
	('السورية', 0.4221523404121399),	('السوري', 0.437679648399353),	('السلطنة', 0.9987720251083374),	('عبري', 0.7974895238876343),
	('حمص', 0.41198962926864624),	('درعا', 0.4253350794315338),	('أكتوبر', 0.9987605810165405),	('البندقية', 0.7914597988128662),
	('بالعاصمة', 0.4072580933570862),	('السوريين', 0.42169952392578125),	('الرئيسية', 0.9987384080886841),	('تريستي', 0.7883596420288086),

	('0.4019525945186615 ,'صقبا'),	('0.41713207960128784 ,'الأردن'),	('0.9987114667892456 ,'شمال'),	('0.7849987149238586 ,'باعتبارها'),
	('0.39999309182167053 ,'حلب')]	('0.4132705330848694 ,'اللاذقية')]	('0.9986864328384399 ,'الجامعات')]	('0.7848200798034668 ,'بوابة')]

3.2. *Resources Allocations and Execution time*

The time consumption of the Skipgram is very high compared to glove models where each GLoVE model finishes training within half an hour while Skipgram models ranges from 2-3 hours based on the parameters values fed. On the other hand, GloVE consumes more memory than skipgram models which is the traditional tradeoff between the memory and training time. Surprisingly, memory size of saved models did not differ regardless of the training parameter values unlike training time which is strongly correlated to the training parameter values.

	skipgram			**GLoVE**			
	model1	**model2**	**model3**	**model1**	**model2**	**model3**	**model4**
Training Time(sec)	7,200.00	7,350	7,800	610.114	610.05	608.41	607.06
Momory Allocation(k)	15,744	15,744	15,744	71,324	71,324	71,324	71,324

4. Conclusion and Future Work

There is no doubt that natural language tasks are challenging to evaluate and build since the language is relative from person to person and from area to area within the same country, not forgetting the evolution of vocabulary and its use from generation to generation. For that reason, we chose to test the word embedding models on the formal Arabic Language used in academic articles. The performance of both models differs due to the difference in the techniques and principles of capturing word meaning and representation. Word2Vec SkipGram throughout the experiments showed a good understanding of the derivatives of the words which is essential to the Arabic language however, it did not fully grasp the semantics of each sentence. While GLoVE understood the sentences and words meanings in the realm of topic by linking words of a topic together evident by their frequent co-occurrence in that topic. Another insight deduced from the experiments is importance of the parameter tuning on GLoVE model as it highly influenced the results therefore, a future work is considered of training GLoVE on larger dataset and focus more on the parameters of GLoVE aiming to optimize its performance within realistic measures leaving margin for acceptable error.

5. References

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

[2] D. Gildea, D. Jurafsky, Automatic labeling of semantic roles, Computational linguistics 28 (2002) 245–288.

[3] H. Hapke, C. Howard, H. Lane, Natural Language Processing in Action: Understanding, analyzing, and generating text with Python, Simon and Schuster, 2019.

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