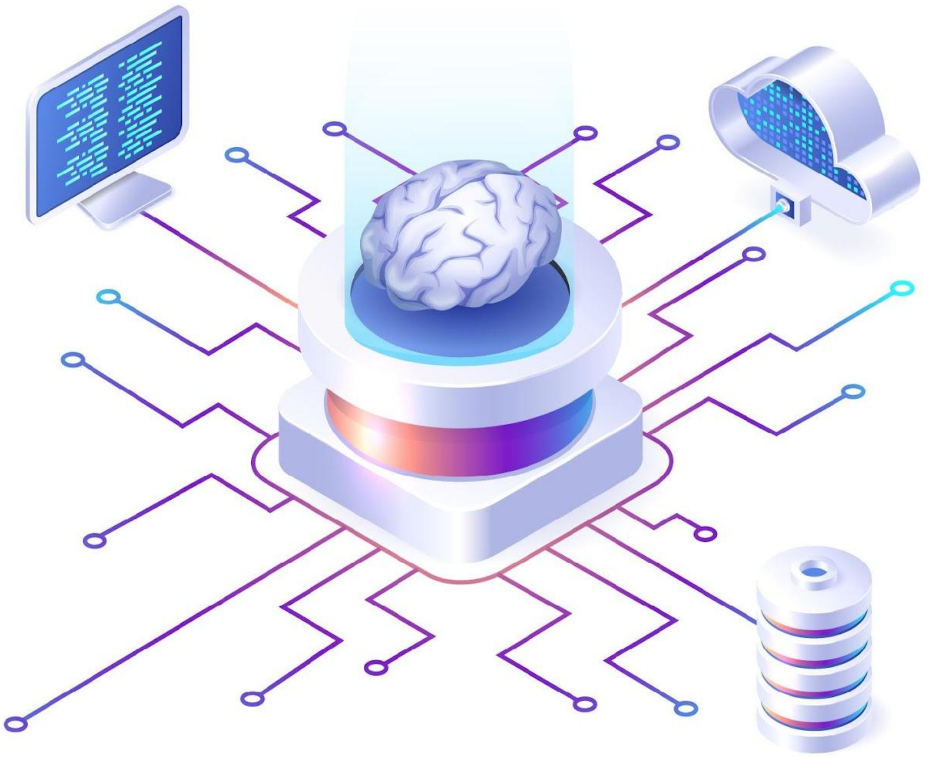
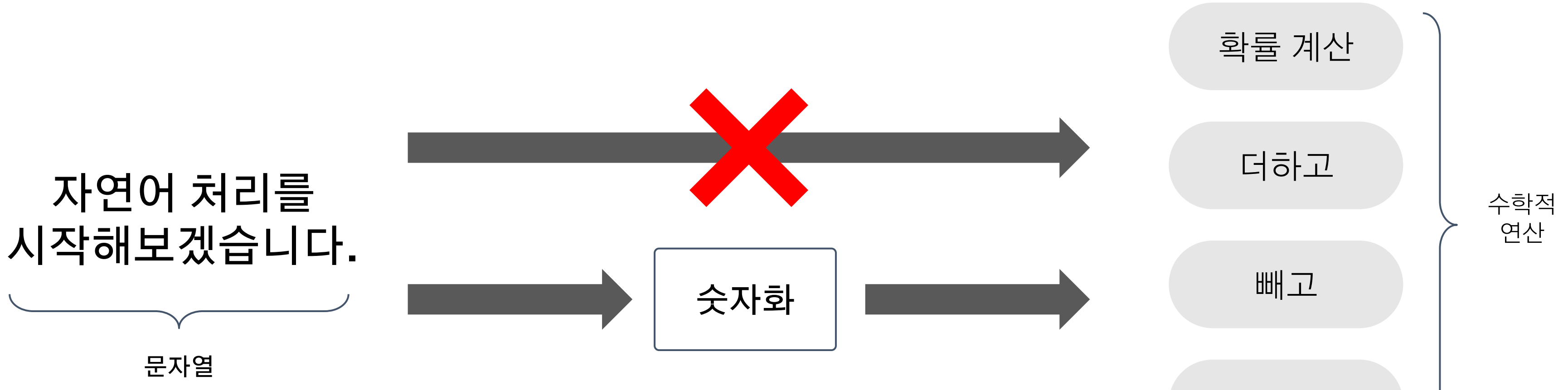
임베딩 (Embedding)

실무형 인공지능 자연어처리



단어의 표현이 필요한 이유



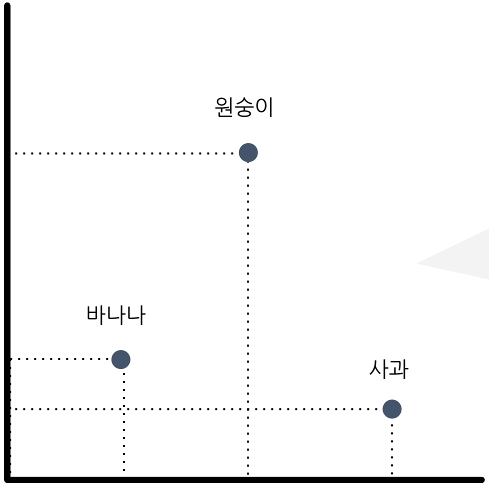
원핫-인코딩(One-Hot-Encoding) 한계점

벡터로 표현한 단어 차원이 너무 큼 → 연산이 낭비되어 모델 학습에 불리하게 적용

단어 의미를 담지 못함 → 분석을 효과적으로 수행할 수 없음

단어 임베딩 (Word Embedding)의 한계

벡터로 표현한 단어 차원이 너무 큼 → 밀집 벡터(Dense vector)로 해결



사과 벡터는 어디에
표현되는 것이 맞을까요?

=> 단어를 벡터로
표현하는 명확한 방법이
존재하지 않음

단어 의미를 담지 못함 → ?

1

Word2Vec

Efficient Estimation of Word Representations in Vector Space

Insight campus SeSAC

Abstract

We propose two novel model architectures for computing continuous vector representations of words from very large data sets. The quality of these representations is measured in a word similarity task, and the results are compared to the previously best performing techniques based on different types of neural networks. We observe large improvements in accuracy at much lower computational cost, i.e. it takes less than a day to learn high quality word vectors from a 1.6 billion words data set. Furthermore, we show that these vectors provide state-of-the-art performance on our test set for measuring syntactic and semantic word similarities.

- 단어의 연속적 벡터 표현의 2가지 모델을 제안
- 단어 유사성으로 이 벡터 표현의 질을 측정
- 더 적은 비용으로 높은 정확도(accuracy)를 개선

Introduction

Many current NLP systems and techniques treat words as atomic units - there is no notion of similarity between words, as these are represented as indices in a vocabulary. This choice has several good reasons - simplicity, robustness and the observation that simple models trained on huge amounts of data outperform complex systems trained on less data. … With progress of machine learning techniques in recent years, it has become possible to train more complex models on much larger data set, and they typically outperform the simple models. Probably the most successful concept is to use distributed representations of words . For example, neural network based language models significantly outperform N-gram models

- 단어를 원자 단위(=원핫 인코딩)로 보기 때문에 단어간 유사성에 대한 고려가 없음
- 많은 양(huge) 데이터를 활용한 단순 모델이 적은데이터 복잡한 모델을 적용한 것보다 성능이 좋다
- 기술발전으로 많은 양 데이터를 복잡한 모델로 학습시키는 것이 가능해짐. => 이 경우 단어의 분산 표형(distributed representation)을 사용

Insight campus SeSAC

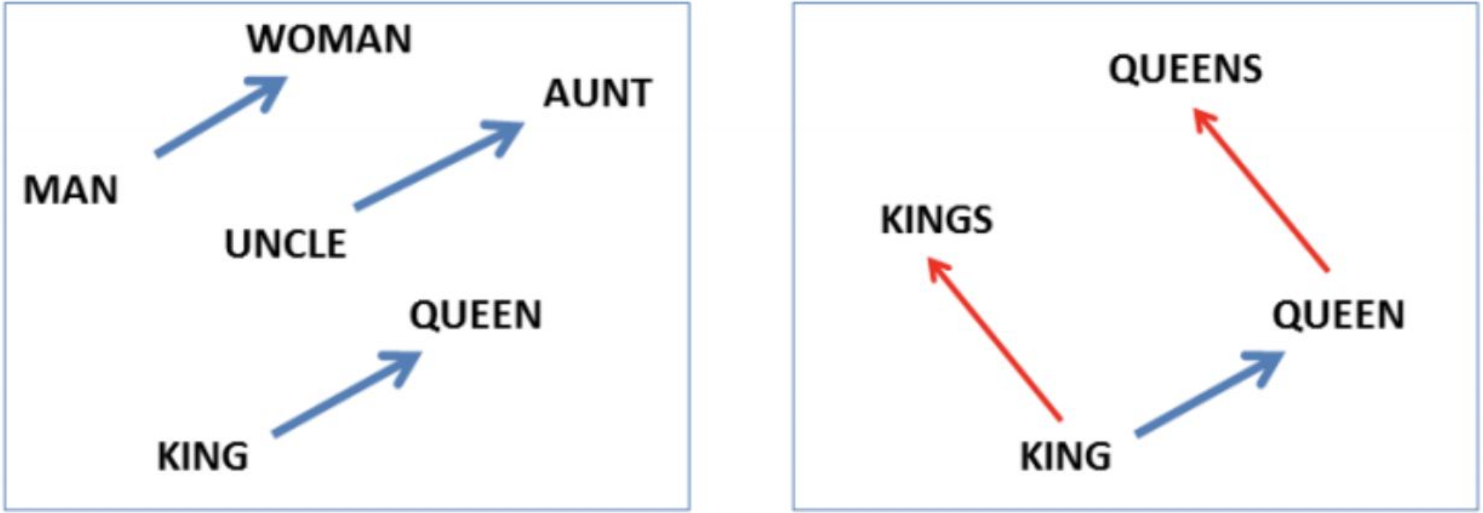
Goals of the Paper

The main goal of this paper is to introduce techniques that can be used for learning high-quality word vectors from huge data sets with billions of words, and with millions of words in the vocabulary … We use recently proposed techniques for measuring the quality of the resulting vector representations, with the expectation that not only will similar words tend to be close to each other, but that words can have multiple degrees of similarity … it was shown for example that vector(”King”) - vector(”Man”) + vector(”Woman”) results in a vector that is closest to the vector representation of the word Queen … we try to maximize accuracy of these vector operations by developing new model architectures that preserve the linear regularities among words. We design a new comprehensive test set for measuring both syntactic and semantic regularities1, and show that many such regularities can be learned with high accuracy. Moreover, we discuss how training time and accuracy depends on the dimensionality of the word vectors and on the amount of the training data.

- 수십억 단어로 질 좋은(high-quality) 단어 벡터를 학습하는 방법
- 유사단어 간에는 거리가 가까운 경향이 있고, 단어는 다양한 유사도를 가진다

Insight campus SeSAC

의미 보존



(Mikolov et al., NAACL HLT, 2013)

Insight campus SeSAC

Model architecture

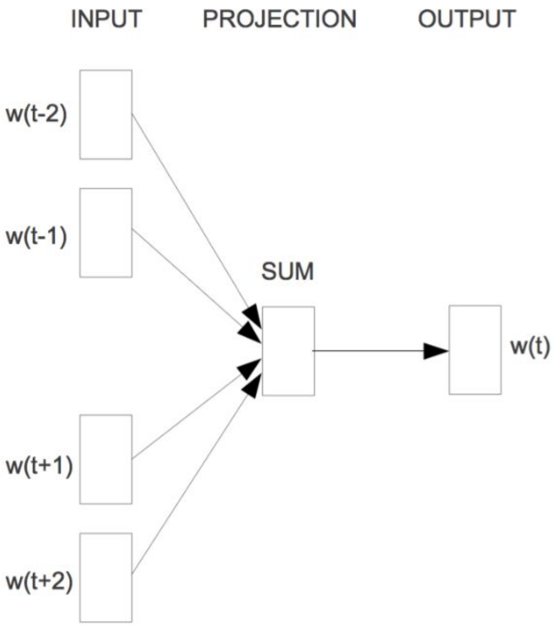
In this paper, we focus on distributed representations of words learned by neural networks, as it was previously shown that they perform significantly better than LSA for preserving linear regularities among words [20, 31]; LDA moreover becomes computationally very expensive on large data sets…For all the following models, the training complexity is proportional to

$$O = E \times T \times Q$$

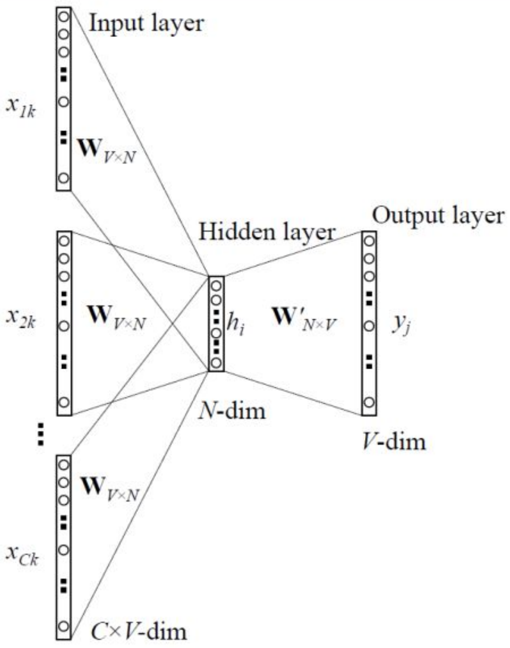
where E is number of the training epochs, T is the number of the words in the training set and Q is defined further for each model architecture. Common choice is $E = 3 - 50$ and T up to one billion. All models are trained using stochastic gradient descent and backpropagation

- LSA보다 뛰어난 선형 정규성(Linear regularity)
- LDA는 데이터 양이 많을 수록 많은 연산을 필요로함
- Word2vec의 복잡도는 $O = E \times T \times Q$ (E : Epoch, T : 트레이닝셋 단어갯수, Q : 모델마다 별도 설정)

Continuous Bag-of-Words Model



CBOW



CBOW Architecture

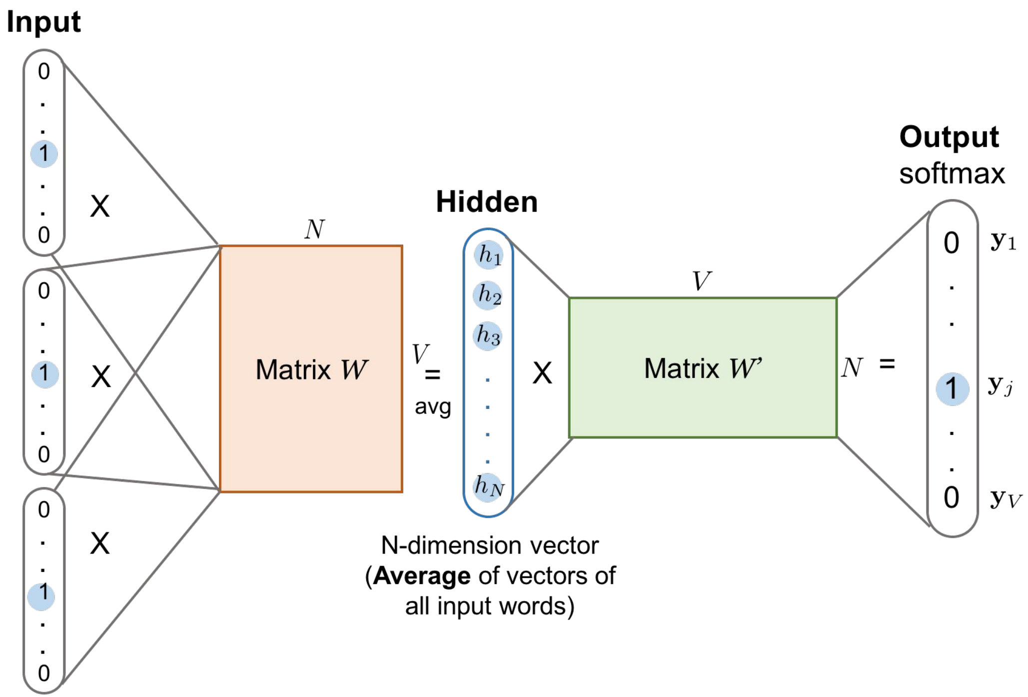
Continuous Bag-of-Words Model

빈칸에 어떤 단어가 들어갈 수 있을까?

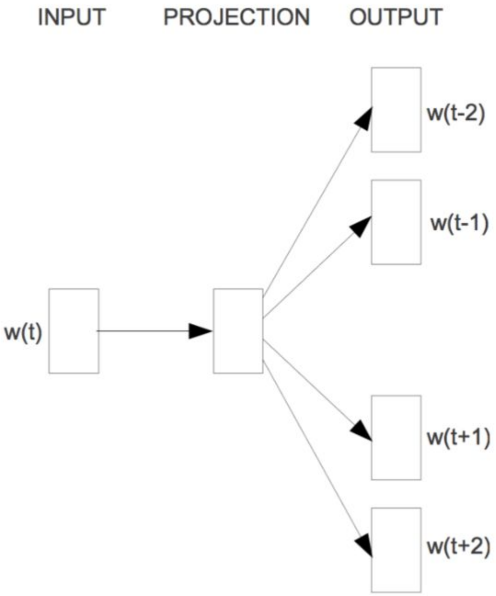
Continuous Bag-of-Words Model

Sliding window (size = 5)	Target word	Context
[The man who]	the	man, who
[The man who passes]	man	the, who, passes
[The man who passes the]	who	the, man, passes, the
[man who passes the sentence]	passes	man, who, the, sentence
...	...	...
[sentence should swing the sword]	swing	sentence, should, the, sword
[should swing the sword]	the	should, swing, sword
[swing the sword]	sword	swing, the

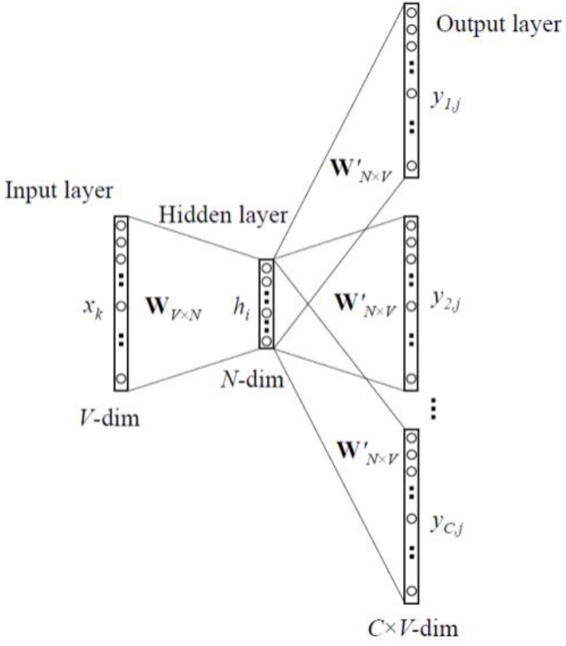
Continuous Bag-of-Words Model



Skip-gram Model



Skip-gram



Skip-gram Architecture

Conclusion

We observed that it is possible to train high quality word vectors using very simple model architectures, compared to the popular neural network models (both feedforward and recurrent). Because of the much lower computational complexity, it is possible to compute very accurate high dimensional word vectors from a much larger data set... To find a word that is similar to small in the same sense as biggest is similar to big, we can simply compute vector X = vector(”biggest”) −vector(”big”) + vector(”small”)....Finally, we found that when we train high dimensional word vectors on a large amount of data, the resulting vectors can be used to answer very subtle semantic relationships between words, such as a city and the country it belongs to, e.g. France is to Paris as Germany is to Berlin. Word vectors with such semantic relationships could be used to improve many existing NLP applications, such as machine translation, information retrieval and question answering systems, and may enable other future applications yet to be invented.

- 낮은 연산 복잡도
- 높은 정확도

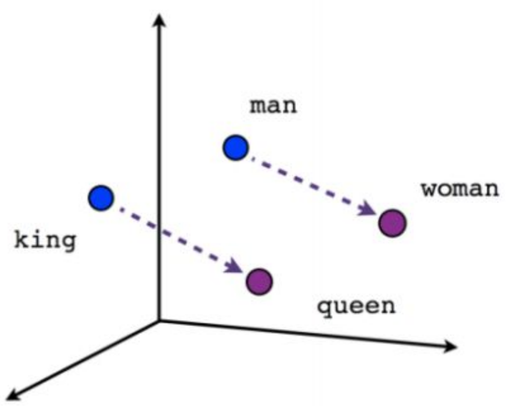
Comparison

Table 1: *Examples of five types of semantic and nine types of syntactic questions in the Semantic-Syntactic Word Relationship test set.*

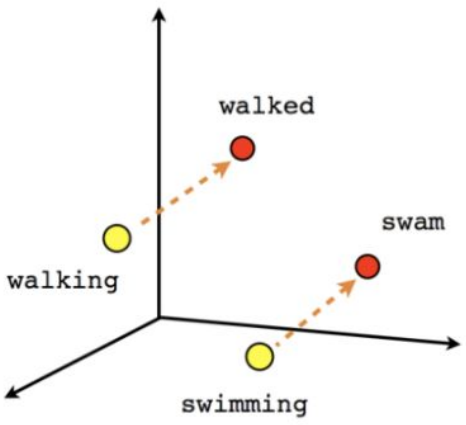
Type of relationship	Word Pair 1		Word Pair 2	
Common capital city	Athens	Greece	Oslo	Norway
All capital cities	Astana	Kazakhstan	Harare	Zimbabwe
Currency	Angola	kwanza	Iran	rial
City-in-state	Chicago	Illinois	Stockton	California
Man-Woman	brother	sister	grandson	granddaughter
Adjective to adverb	apparent	apparently	rapid	rapidly
Opposite	possibly	impossibly	ethical	unethical
Comparative	great	greater	tough	tougher
Superlative	easy	easiest	lucky	luckiest
Present Participle	think	thinking	read	reading
Nationality adjective	Switzerland	Swiss	Cambodia	Cambodian
Past tense	walking	walked	swimming	swam
Plural nouns	mouse	mice	dollar	dollars
Plural verbs	work	works	speak	speaks

- 8869개 의미적 문항과 10675개의 문법적 문항으로 테스트

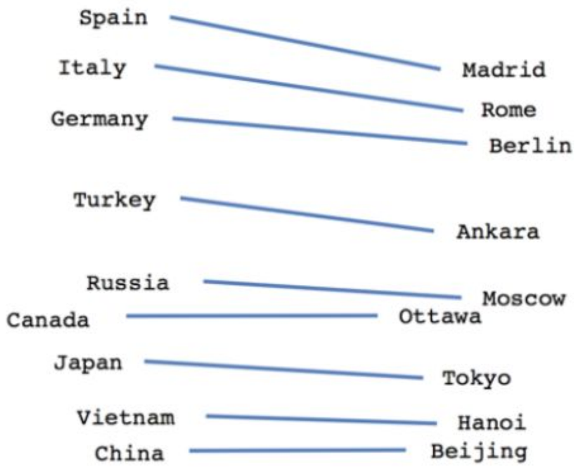
Comparison



Male-Female



Verb tense



Country-Capital

https://tensorflowkorea.gitbooks.io/tensorflow-kr/content/g3doc/tutorials/word2vec/

Comparison

Table 2: *Accuracy on subset of the Semantic-Syntactic Word Relationship test set, using word vectors from the CBOW architecture with limited vocabulary. Only questions containing words from the most frequent 30k words are used.*

Dimensionality / Training words	24M	49M	98M	196M	391M	783M
50	13.4	15.7	18.6	19.1	22.5	23.2
100	19.4	23.1	27.8	28.7	33.4	32.2
300	23.2	29.2	35.3	38.6	43.7	45.9
600	24.0	30.1	36.5	40.8	46.6	50.4

Table 3: *Comparison of architectures using models trained on the same data, with 640-dimensional word vectors. The accuracies are reported on our Semantic-Syntactic Word Relationship test set, and on the syntactic relationship test set of [20]*

Model Architecture	Semantic-Syntactic Word Relationship test set		MSR Word Relatedness Test Set [20]
	Semantic Accuracy [%]	Syntactic Accuracy [%]	
RNNLM	9	36	35
NNLM	23	53	47
CBOW	24	64	61
Skip-gram	55	59	56

Insight campus SeSAC

Comparison

Table 4: *Comparison of publicly available word vectors on the Semantic-Syntactic Word Relationship test set, and word vectors from our models. Full vocabularies are used.*

Model	Vector Dimensionality	Training words	Accuracy [%]		
			Semantic	Syntactic	Total
Collobert-Weston NNLM	50	660M	9.3	12.3	11.0
Turian NNLM	50	37M	1.4	2.6	2.1
Turian NNLM	200	37M	1.4	2.2	1.8
Mnih NNLM	50	37M	1.8	9.1	5.8
Mnih NNLM	100	37M	3.3	13.2	8.8
Mikolov RNNLM	80	320M	4.9	18.4	12.7
Mikolov RNNLM	640	320M	8.6	36.5	24.6
Huang NNLM	50	990M	13.3	11.6	12.3
Our NNLM	20	6B	12.9	26.4	20.3
Our NNLM	50	6B	27.9	55.8	43.2
Our NNLM	100	6B	34.2	**64.5**	50.8
CBOW	300	783M	15.5	53.1	36.1
Skip-gram	300	783M	**50.0**	55.9	**53.3**

Comparison

Table 5: *Comparison of models trained for three epochs on the same data and models trained for one epoch. Accuracy is reported on the full Semantic-Syntactic data set.*

Model	Vector Dimensionality	Training words	Accuracy [%]			Training time [days]
			Semantic	Syntactic	Total	
3 epoch CBOW	300	783M	15.5	53.1	36.1	1
3 epoch Skip-gram	300	783M	50.0	55.9	53.3	3
1 epoch CBOW	300	783M	13.8	49.9	33.6	0.3
1 epoch CBOW	300	1.6B	16.1	52.6	36.1	0.6
1 epoch CBOW	600	783M	15.4	53.3	36.2	0.7
1 epoch Skip-gram	300	783M	45.6	52.2	49.2	1
1 epoch Skip-gram	300	1.6B	52.2	55.1	53.8	2
1 epoch Skip-gram	600	783M	56.7	54.5	55.5	2.5

Comparison

Table 6: *Comparison of models trained using the DistBelief distributed framework. Note that training of NNLM with 1000-dimensional vectors would take too long to complete.*

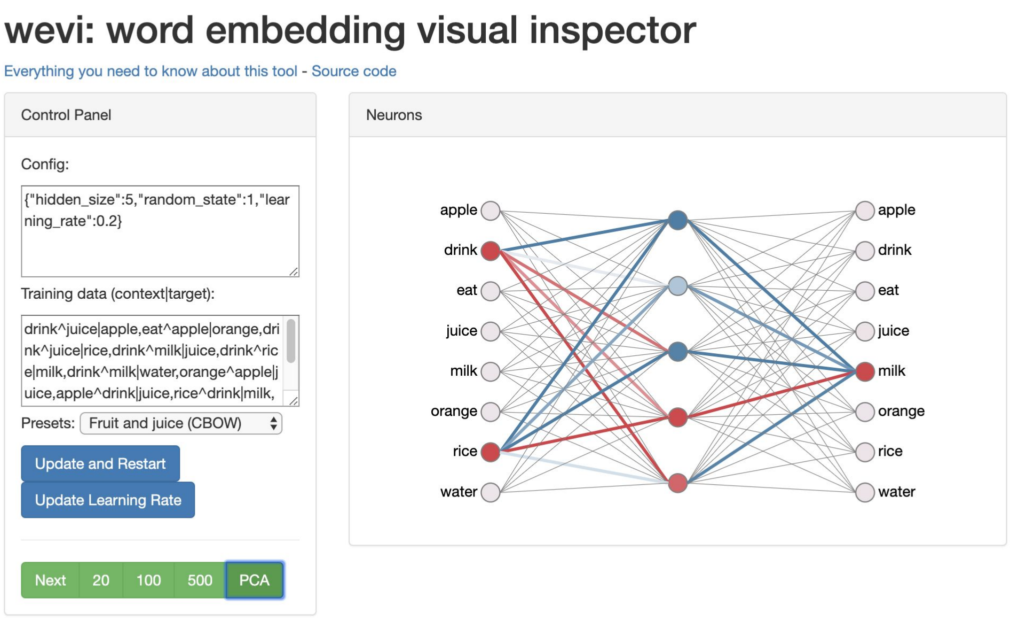
Model	Vector Dimensionality	Training words	Accuracy [%]			Training time [days x CPU cores]
			Semantic	Syntactic	Total	
NNLM	100	6B	34.2	64.5	50.8	14 x 180
CBOW	1000	6B	57.3	68.9	63.7	2 x 140
Skip-gram	1000	6B	66.1	65.1	65.6	2.5 x 125

Table 7: *Comparison and combination of models on the Microsoft Sentence Completion Challenge.*

Architecture	Accuracy [%]
4-gram [32]	39
Average LSA similarity [32]	49
Log-bilinear model [24]	54.8
RNNLMs [19]	55.4
Skip-gram	48.0
Skip-gram + RNNLMs	**58.9**

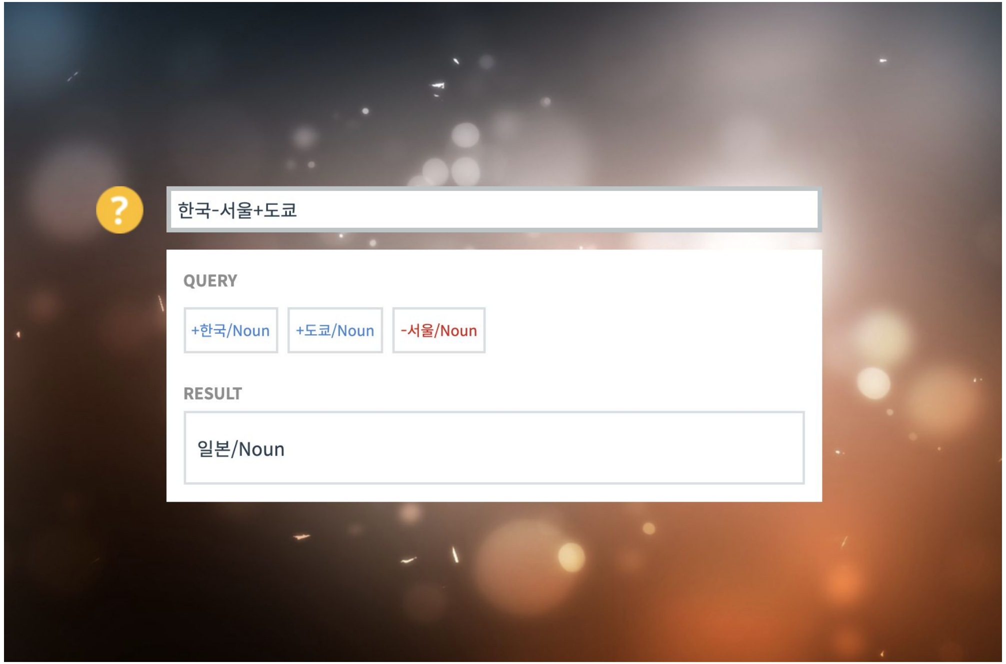
Word2Vec 시연

https://ronxin.github.io/wevi/



한글 Word2Vec

http://word2vec.kr/search/?query=%ED%95%9C%EA%B5%AD-%EC%84%9C%EC%9A%B8%2B%EB%8F%84%EC%BF%84



감사합니다.

daniel@fins.ai