**Applications in natural Language Processing**

**Statistics 1828013 Song Hye Jun**

< Table of Contents >

1. continuous bag-of-words(CBOW) & Continuous skip-gram

2. Skip-Gram + Negative Sampling

3. GloVe(Global Vectors for Word Representation)

4. FastText (Skip-gram + sub-word information (character n-grams))

5. summary

1. 📌 **Continuous Bag-of-Words & Continuous Skip-gram Model**
2. **Goal**

It has become important to train more complex models on larger datasets using distributed representations of words to artificial neural network-based models rather than simply learning large amounts of data. The main objective of this paper is to introduce a technique that can be used to learn high-quality word vectors on a huge dataset with billions of words. There are techniques for measuring the quality of the resulting vector representations: similar words tend to be close to each other, words can have multiple degrees of similarity and even similarities in word representations go beyond simple syntactic regularity. (ex : vector ("King") - vector ("Man") + vector ("Woman")). It is to maximize accuracy of these vector operations by developing new model architectures that preserve the linear regularities among words.

And the problem in artificial neural networks (NNLM, RNNLM) used before the new two model architectures is that the computational complexity is large and most computational complexities are caused by nonlinear hidden layers. Therefore, it is important to maximize accuracy while minimizing computational complexity by creating a simple model that can learn data more efficiently.

1. **Two Model Architectures**

**a) Continuous Bag-of-Words Model**

This structure is similar to NLM in that the weight matrix between the input and the projection layer is shared at all word positions but differs in that nonlinear hidden layers are eliminated. This architecture is called a bag-of -words models as the order of the past words does not affect the projection. (It is easy to understand that it is unordered set.) In addition, we also use future words together to create a log-linear classifier with four words from the past and four words from the future as input values. And the training criterion is whether the current word is classified correctly. The model is the Continuous Bag of Word (CBOW). It uses consecutive distributed contextual representations. Training complexity is as follows. Q = N X D + D + log2(V)

**b) Continuous Skip-gram Model**

The structure of this model is similar to CBOW, but it maximizes the classification of words based on other words in the same sentence. More specifically, each current word as an input is used to a log-linear classifier with a continuous projection layer and predicts words in a specific range before and after the current word. It is important to give less weight to the distant words by sampling less from those words in training examples. The training complexity of this architecture is proportional to Q = C × (D + D × log2(V )), where C is the maximum distance of the words. Thus, if we choose C = 10, for each training word we will select randomly a number R in range (1; C ), and then use R words from history and R words from the future of the current word as correct labels. This requires to do R × 2 word classifications.

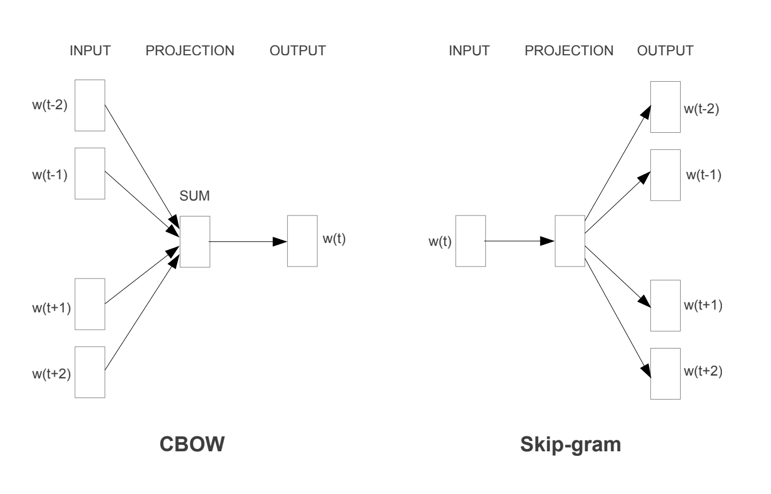


Figure 1: model architectures. The CBOW architecture predicts the current word based on the context which contains future and history words, and the Skip-gram predicts words nearby given the current word[1].

1. **Result**

The way to compare the quality of different versions of words vectors is as follows. By performing algebraic operations with the vector representation of words. For example to find the word which is similar in the same sense as biggest is similar to big is to compute vector X = vector(“biggest”) − vector(“big”) + vector(“small”). By training high dimensional word vectors on a large amount of data, the resulting vectors can be used to answer fine semantic relationships between words such as a city and the country it belongs to.

1. **Conclusion**

It is possible to train high quality word vectors using very simple model architectures, compared to the neural network models(feedforward and recurrent). Much lower computational complexity enables to compute very accurate high dimensional word vectors from a much larger data set.

1. **📌 The advance of Skip-Gram : Negative Sampling**
2. **Abstract**

Skip-gram model is efficient method for learning high-quality distributed vector representations that capture many syntactic and semantic word relationships. Subsampling of frequently used words enables speed up and more regular word expressions. And negative sampling is an alternative to instead of the hierarchical SoftMax.

1. **Goal**

The main objective of this paper is to improve the original Skip-gram model. By sub-sampling high-frequency words, it was possible to increase the speed of training by 2 to 10 times while increasing the accuracy of expression of low-frequency words. Additionally, in this paper, we demonstrate that the use of Noise Contrastive Estimation (NCE)\* for Skip-gram model training instead of the more complex hierarchical softmax used previously enables better representation for faster, more frequent words.

\* Noise Contrastive Estimation (NCE) : We consider the task of estimating, from observed data, a probabilistic model that is parameterized by a finite number of parameters. We are considering the situation where the model probability density function is unnormalized. That is, the model is only specified up to the partition function. The partition function normalizes a model so that it integrates to one for any choice of the parameters. However, it is often impossible to obtain it in closed form. Then, Maximum likelihood estimation can then not be used. So, we propose here a new objective function for the estimation of both normalized and unnormalized models. The basic idea is to perform nonlinear logistic regression to discriminate between the observed data and some artificially generated noise. With this approach, the normalizing partition function can be estimated like any other parameter. We prove that the new estimation method leads to a consistent (convergent) estimator of the parameters. For large noise sample sizes, the new estimator is furthermore shown to behave like the maximum likelihood estimator. In the estimation of unnormalized models, there is a trade-off between statistical and computational performance. We show that the new method strikes a competitive trade-off in comparison to other estimation methods for unnormalized models[3].

And there was a limitation that the expression of words was impossible to express the idiomatic phrases because individual words could not be combined. However, the use of vector representations for all idioms could further enhance the expression of skip-gram models. Another technique for expressing the meaning of sentences composed of combined words is to use recursive autonconders\*. This made it possible to use Phrase vector instead of word vector.

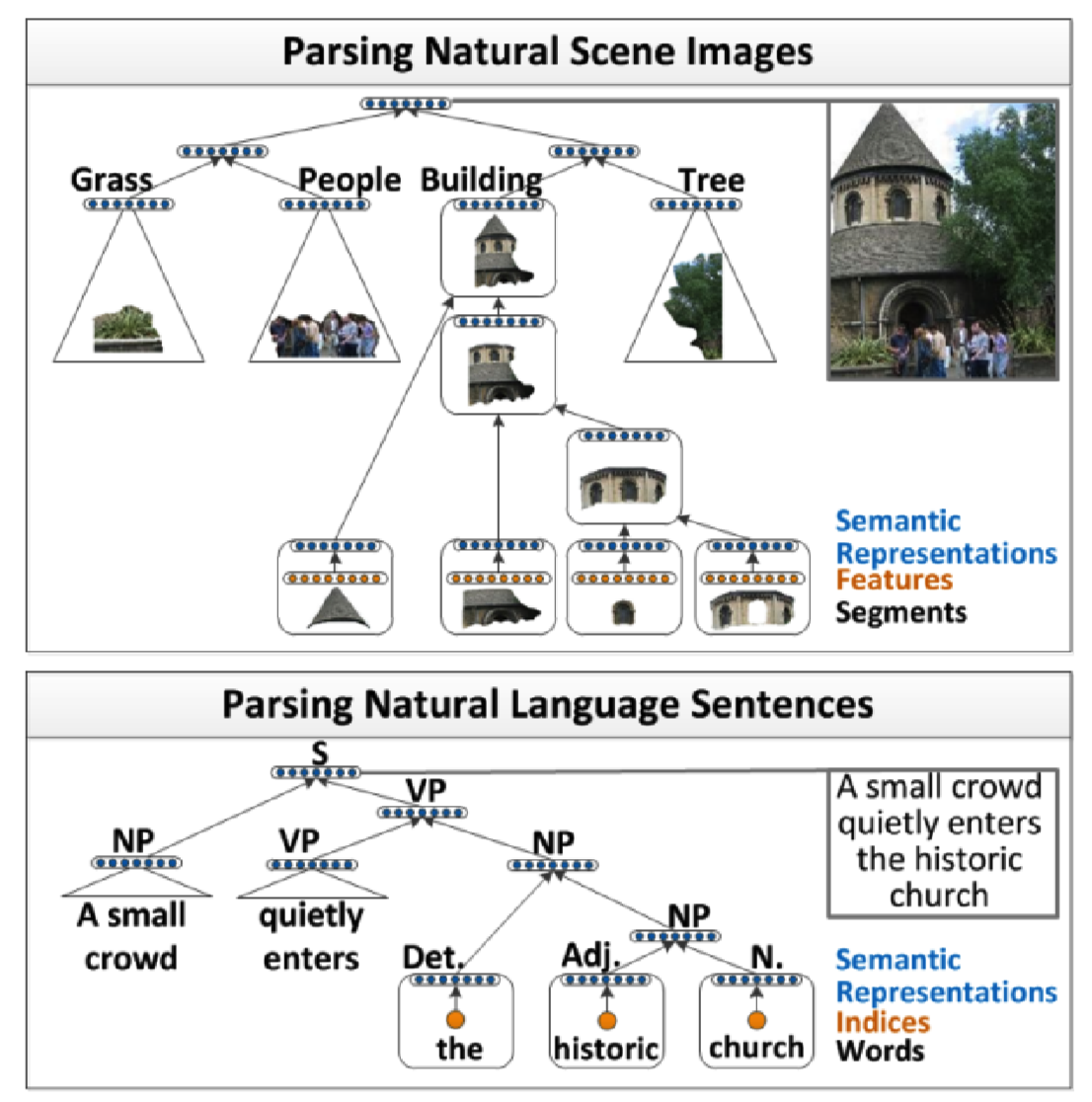


Figure 1. Illustration of our recursive neural network architecture which parses images and natural language sentences. Segment features and word indices (orange) are first mapped into semantic feature space (blue) and **then recursively merged by the same neural network** until they represent the entire image or sentence. Both mappings and mergings are learned [4].

\*autoencoders : We introduce a max-margin structure prediction architecture based on recursive neural networks that can successfully recover such structure both in complex scene images as well as sentences. The same algorithm can be used both to provide a competitive syntactic parser for natural language sentences from the Penn Treebank and to outperform alternative approaches for semantic scene segmentation, annotation, and classification.

In this paper, we introduce recursive neural networks (RNNs) for predicting recursive structure in multiple modalities.

Fig. 1 outlines our approach for both modalities. Images are oversegmented into small regions which often represent parts of objects or background. From these regions we extract vision features and then map these features into a “semantic” space using a neural network. Using these semantic region representations as input, our RNN computes (i) a score that is higher when neighboring regions should be merged into a larger region, (ii) a new semantic feature representation for this larger region, and (iii) its class label. Class labels in images are visual object categories such as building or street. The model is trained so that the score is high when neighboring regions have the same class label. After regions with the same object label are merged, neighboring objects are merged to form the full scene image. These merging decisions implicitly define a tree structure in which each node has associated with it the RNN outputs (i)-(iii), and higher nodes represent increasingly larger elements of the image.

The same algorithm is used to parse natural language sentences. Again, words are first mapped into a semantic space and then they are merged into phrases in a syntactically and semantically meaningful order. The RNN computes the same three outputs and attaches them to each node in the parse tree. The class labels are phrase types such as noun phrase (NP) or verb phrase (VP) [4].

Finally, we find that we can often produce meaningful results with the addition of simple vectors (ex. the vector (Germany) + vector (capital) was close to the vector (Berlin)). This combination suggests that the degree of understanding of unclear languages can be expressed through mathematical operations.

1. **The Skip-gram Model**

The training objective of the Skip-gram model is to find word representations that are useful for predicting the surrounding words in a sentence. In other words, It maximizes the average log probability where c is the size of the training context

텍스트이(가) 표시된 사진

자동 생성된 설명

The basic Skip-gram formulation defines p(wt+j |wt ) using the softmax function:

텍스트이(가) 표시된 사진

자동 생성된 설명

vw : Input vector representation of w, vw′ : output vector representation of w, W : # of words in the vocabulary

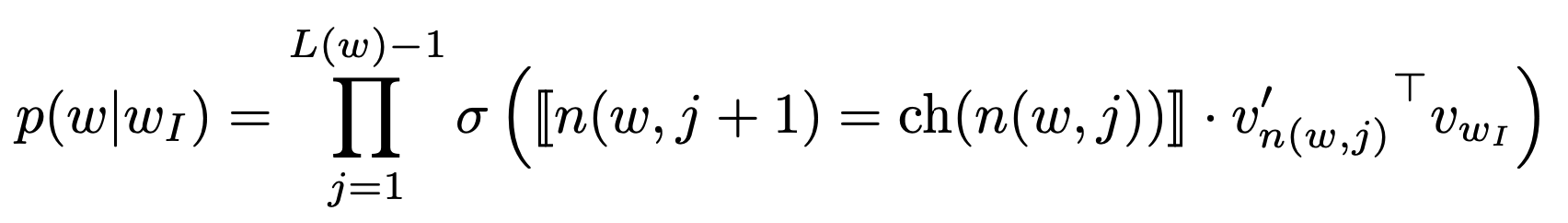
This formulation is impractical because the cost of computing ∇logp(wO | wI) is proportional to W.

1. **Hierarchical Softmax**

A computationally efficient approximation of the full softmax is the hierarchical softmax. The hierarchical softmax uses a binary tree representation of the output layer with the W words as its leaves and, for each node, explicitly represents the relative probabilities of its child nodes. More precisely, each word w can be reached by an appropriate path from the root of the tree.

n(w, j): the j-th node on the path from the root to L(w): the length of this path, Ex) n(w, 1) = root, n(w, L(w)) = w.

In addition, for any inner node n, let ch(n) be an arbitrary fixed child of n and let [[x]] be 1 if x is true and -1 otherwise. Then the hierarchical softmax defines p(wO |wI ) as follows:



It can be verified that . This implies that the cost of computing log p(wO |wI ) and ∇ log p(wO |wI ) is proportional to *L*(wO ), which on average is no greater than log W . Also, the hierarchical softmax formulation has one representation vw for each word w and one representation vn′ for every inner node n of the binary tree. A binary Huffman tree assigns short codes to the frequent words which results in fast training.

1. **Negative sampling**

An alternative to the hierarchical softmax is Noise Contrastive Estimation (NCE). NCE posits that a good model should be able to differentiate data from noise by means of logistic regression. The Skip-gram model is only concerned with learning high-quality vector representations, so we are free to simplify NCE as long as the vector representations retain their quality. We define Negative sampling (NEG) by the objective which is used to replace every log P (wO |wI ) term in the Skip-gram objective.

텍스트이(가) 표시된 사진

자동 생성된 설명

Thus the task is to distinguish the target word wO from draws from the noise distribution Pn(w) using logistic regression, where there are k negative samples for each data sample. The difference between the NEG and NCE is that NCE needs both samples and the numerical probabilities of the noise distribution, while NEG uses only samples. It is important to investigate a number of choices for Pn(w).

1. **Subsampling of Frequent Words**

Generally, the most frequent words usually provide less information value than the rare words. To counter the imbalance between the rare and frequent words, we used a simple subsampling approach that each word wi in the training set is discarded with probability computed by the formula.

**텍스트, 시계이(가) 표시된 사진

자동 생성된 설명**

It aggressively subsamples words whose frequency is greater than t while preserving the ranking of the frequencies. It accelerates learning and even significantly improves the accuracy of the learned vectors of the rare words.

1. **Empirical Results**

The Hierarchical Softmax (HS), Noise Contrastive Estimation(NCE), Negative Sampling(NEG), and subsampling of the training words are evaluated by using the analogical reasoning task. The task consists of analogies such as “Germany” : “Berlin” :: “France” : ?, which are solved by finding a vector x such that vec(x) is closest to vec(“Berlin”) - vec(“Germany”) + vec(“France”) according to the cosine distance. This specific example is considered to have been answered correctly if x is “Paris”. The task has two broad categories: the syntactic analogies (ex. “quick” : “quickly”) and the semantic analogies (ex. country - capital city). The results are followings: Negative Sampling outperforms the Hierarchical SoftMax on the analogical reasoning task and has even slightly better performance than the Noise Contrastive Estimation. The subsampling of the frequent words improves the training speed several times and makes the word representations significantly more accurate.

1. **Learning Phrases**

the phrase-based training corpus was constructed and then we trained several Skip-gram models using different hyperparameters. Using vector dimensionality 300 and context size 5, It already achieves good performance on the phrase dataset, and quickly compare the Negative Sampling and the Hierarchical Softmax, both with and without subsampling of the frequent tokens. The results show that while Negative Sampling achieves a respectable accuracy even with k = 5, using k = 15 achieves considerably better performance.

1. **Conclusion**

This work has several key contributions. It shows that how to train distributed representations of words and phrases with the Skip-gram model and demonstrate that these representations exhibit linear structure that makes precise analogical reasoning possible.

And It results in a great improvement in the quality of the learned word and phrase representations, especially for the rare entities. It was also found that the subsampling of the frequent words results in both faster training and significantly better representations of uncommon words. Another contribution of our paper is the Negative sampling algorithm, which is an extremely simple training method that learns accurate representations especially for frequent words.

A very interesting result of this work is that the word vectors can be somewhat “meaningfully” combined using just simple vector addition. Another approach for learning representations of phrases is to simply represent the phrases with a single token. Combination of these two approaches gives a powerful yet simple way how to represent longer pieces of text, while having minimal computational complexity.

1. **📌 GloVe: Global Vectors for Word Representation**
2. **Abstract**

Recent methods for learning vector space representations of words have succeeded in capturing fine-grained semantic and syntactic regularities using vector arithmetic, but the origin of these regularities has remained opaque.

It is needed to analyze and make explicit the model properties needed for such regularities to emerge in word vectors. The result is a new global log-bilinear regression model that combines the advantages of the two major model families in the literature: global matrix factorization and local context window methods. This model efficiently leverages statistical information by training only on the nonzero elements in a word-word co-occurrence matrix.

1. **Introduction**

The two main model families for learning word vectors are: 1) global matrix factorization methods, such as latent semantic analysis (LSA)and 2) local context window methods, such as the skip-gram model. Currently, both families suffer significant drawbacks. While methods like LSA efficiently leverage statistical information, they do relatively poorly on the word “analogy task”. Methods like skip-gram may do better on the analogy task, but they poorly utilize “the statistics” of the corpus since they train on separate local context windows instead of on global co-occurrence counts.

In this work, the model properties necessary to produce “linear directions of meaning” is analyzed and it is argued that global log-bilinear regression models are appropriate for doing so. And a specific weighted least squares model that trains on global word-word co-occurrence counts is proposed. and thus, it makes efficient use of statistics. The model produces a word vector space with meaningful substructure, as evidenced by its state-of-the-art performance of 75% accuracy on the word analogy dataset.

1. **Related work**
2. **Matrix Factorization Methods**

Matrix factorization methods for generating low-dimensional word representations have roots stretching as far back as LSA. These methods utilize low-rank approximations to decompose large matrices that capture statistical information about a corpus. In LSA, the matrices are of “term-document” type, i.e., the rows correspond to words or terms, and the columns correspond to different documents in the corpus.

In contrast, the Hyperspace Analogue to Language (HAL) for example, utilizes matrices of “term-term” type, i.e., the rows and columns correspond to words and the entries correspond to the number of times a given word occurs in the context of another given word. A main problem with HAL and related methods is that the most frequent words contribute a disproportionate amount to the similarity measure: the number of times two words co-occur with ‘the’ or ‘and’, for example, will have a large effect on their similarity despite conveying relatively little about their semantic relatedness. (summary: The model using matrix factorization method does poorly on the word similarity measure)

1. **Shallow Window-Based Methods**

Another approach is to learn word representations that aid in making predictions within local context windows. The skip-gram and continuous bag-of-words (CBOW) models of Mikolov et al. (2013a) propose a simple single-layer architecture based on the inner product between two words vectors. These models(skip-gram ivLBL, / CBOW, vLBL) demonstrated the capacity to learn linguistic patterns as linear relationships between the word vectors.

Unlike the matrix factorization methods, the shallow window-based methods suffer from the disadvantage that they do not operate directly on the co-occurrence statistics of the corpus. Instead, these models scan context windows across the entire corpus, which fails to take advantage of the vast amount of repetition in the data.

(summary : The model using shallow window based methods does poorly utilize the statistics of the corpus.)

1. **The GloVe Model**

The question remains as to how “meaning is generated from these statistics”, and how the resulting word vectors might represent that meaning. Thus, it is needed to construct a new model for word representation which we call GloVe, for Global Vectors, because the global corpus statistics are captured directly by the model.

*X*ij : the matrix of word-word co-occurrence counts, the number of times word *j* occurs in the context of word *i*. *X*i = :the number of times any word appears in the context of word *i*. *P*ij =*P*(*j* |*i* )= *X*i j /*X* i : the probability that word *j* appear in the context of word *i*.

**텍스트, 테이블이(가) 표시된 사진

자동 생성된 설명**

Table 1: Co-occurrence probabilities for target words *ice* and *steam* with selected context words[5]

There is a simple example that showcases how certain aspects of meaning can be extracted directly from co-occurrence probabilities.

Consider two words i and j that exhibit a particular aspect of interest; for concreteness, suppose we are interested in the concept of thermodynamic phase, for which we might take i = ice and j = steam. The relationship of these words can be examined by studying the ratio of their co-occurrence probabilities with various probe words, k. For words k related to ice but not steam, say k = solid, we expect the ratio Pik/Pjk will be large. For words k like water or fashion, that are either related to both ice and steam, or to neither, the ratio should be close to one. Table 1 shows these probabilities and their ratios for a large corpus, and the numbers confirm these expectations. Compared to the raw probabilities, the ratio is better able to distinguish relevant words (solid and gas) from irrelevant words (water and fashion) and it is also better able to discriminate between the two relevant words.

The above argument suggests that the appropriate starting point for word vector learning should be with ratios of co-occurrence probabilities rather than the probabilities themselves.

The most general model takes the form

텍스트이(가) 표시된 사진

자동 생성된 설명(1)

w: word vectors, w ̃ : separate context word vectors

We would like F to encode the information present the ratio Pik/Pjk in the word vector space. Since vector spaces are linear structures, the most natural way to do this is with vector differences. With this aim, it can be done to restrict the consideration to those functions F that depend only on the difference of the two target words, modifying Eqn. (1) to,

텍스트이(가) 표시된 사진

자동 생성된 설명(2)

And, the dot product of the arguments is taken,

텍스트이(가) 표시된 사진

자동 생성된 설명 (3)

It prevents F from mixing the vector dimensions in undesirable ways. Next, note that for word-word co-occurrence matrices, the distinction between a word and a context word is arbitrary and that we are free to exchange the two roles. (w ↔ w ̃, X ↔ ). Our final model should be invariant under this relabeling, but Eqn. (3) is not. However, the symmetry can be restored by two steps. First, It is required that F be a homomorphism between the groups (R,+) and (R>0, ×), i.e.,

**텍스트이(가) 표시된 사진

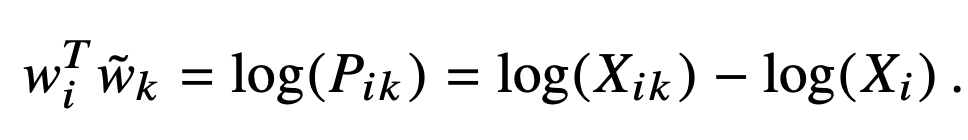
자동 생성된 설명(4)**

which, by Eqn. (3), is solved by,

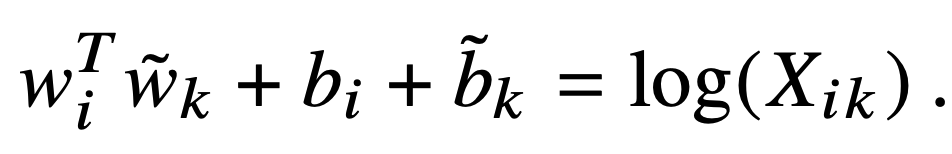
텍스트이(가) 표시된 사진

자동 생성된 설명(5)

The solution to Eqn. (4) is *F* = exp, or,

(6)

log(Xi) is independent of k so it can be absorbed into a bias bi for wi . Finally, adding an additional bias b ̃k for w ̃ k restores the symmetry

(7)

And It is needed to include an additive shift in the logarithm, log(*X*ik ) → log(1 + *X*ik ), which maintains the sparsity of X while avoiding the divergences. The idea of factorizing the log of the co-occurrence matrix is closely related to LSA and we will use the resulting model as a baseline in our experiments. A main drawback to this model is that it weighs all co-occurrences equally, even those that happen rarely or never.

It is proposed that a new weighted least squares regression model that addresses these problems. Casting Eqn. (7) as a least squares problem and introducing a weighting function f (*X*i j) into the cost function gives us the model

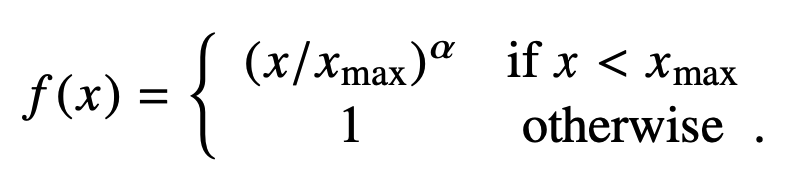
텍스트이(가) 표시된 사진

자동 생성된 설명(8)

where V is the size of the vocabulary.

The weighting function should obey the following : f (0) = 0, non-decreasing, relatively small for large values of x,

One class of functions can be parameterized as,

****(9)

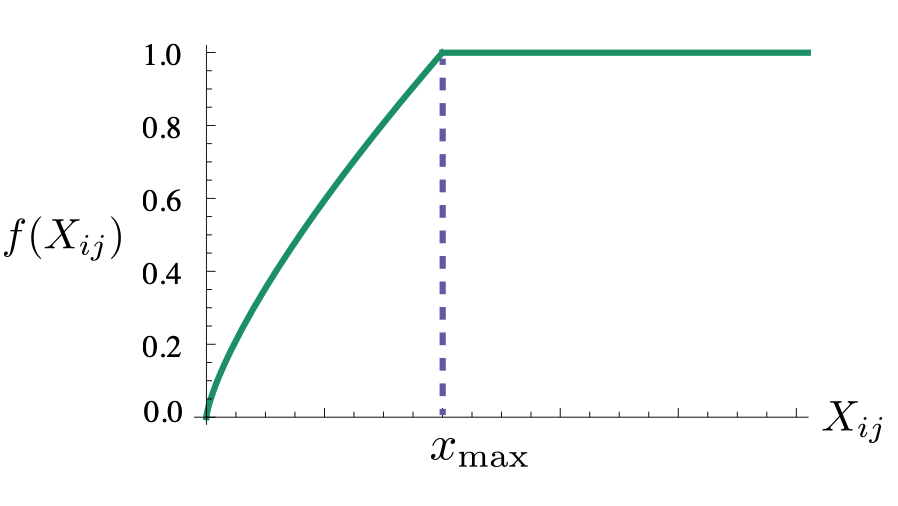
****

Figure 1: Weighting function *f* with α = 3/4 [5].

It was found that α = 3/4 gives a modest improvement over a linear version with α = 1.

1. **Result**

We present results on the word analogy task and the GloVe model performs significantly better than the other baselines, often with smaller vector sizes and smaller corpora. The results using the word2vec tool are somewhat better than most of the previously published results. This is due to several factors, including our choice to use negative sampling (which works better than the hierarchical softmax) and the number of negative samples.

And we present results on five different word similarity datasets. A similarity score is obtained from the word vectors by first normalizing each feature and then calculating the cosine similarity. And Spearman’s rank correlation coefficient between this score and the human judgments is computed.

And we show results on results on the NER (name entity recognition) task with the CRF-based model. The GloVe model outperforms all other methods on all evaluation metrics, except for the CoNLL test set, on which the HPCA method does slightly better. Thus, it is concluded that the GloVe vectors are useful in downstream NLP tasks.

1. **Conclusion**

Recently, considerable attention has been focused on the question of whether distributional word representations are best learned from count-based methods or from prediction-based methods. In this work, it is argued that the efficiency with which the count-based methods capture global statistics can be advantageous. It constructs a model that utilizes this main benefit of count data while simultaneously capturing the meaningful linear substructures prevalent (ex. word2vec) The result, GloVe, is a new global log-bilinear regression model for the unsupervised learning of word representations that outperforms other models on word analogy, word similarity, and named entity recognition tasks.

1. **📌 FastText**
2. **Abstract**

Popular models ignore the morphology of words, by assigning a “distinct vector” to each word. This is a limitation, especially for languages with large vocabularies and many rare words. In this paper,

A new approach based on the skip-gram model, where each word is represented as “a bag of character n-grams” is proposed. A vector representation is associated to each character n-gram; words being represented as the sum of these representations. This method is fast, allowing to train models on large corpora quickly and allows us to compute word representations for words that did not appear in the training data. By comparing to recently proposed morphological word representations, we show that vectors achieve state-of-the-art performance on these tasks.

1. **Introduction**

Recent techniques about learning embeddings represent each word of the vocabulary by a “distinct vector”, without parameter sharing. They ignore the internal structure of words, which is an important limitation for morphologically rich languages, such as Turkish or Finnish. These languages contain many word forms that occur rarely (or not at all) in the training corpus, making it difficult to learn good word representations. Because many word formations follow rules, it is possible to improve vector representations for morphologically rich languages by using character level information.

In this paper, it is proposed to learn representations for character n-grams, and to represent words as the sum of the n-gram vectors. Main contribution is to introduce an extension of the continuous skip-gram model, which considers sub-word information.

1. **Related work**
2. **Morphological word representations.**

Different approaches which use different composition functions to derive representations of words from morphemes rely on a morphological decomposition of words, while ours does not.

Similarly, Chen et al. (2015) introduced a method to jointly learn embeddings for Chinese words and characters. Cui et al. (2015) proposed to constrain morphologically similar words to have similar representations. Soricut and Och (2015) described a method to learn vector representations of morphological transformations, allowing to obtain representations for unseen words by applying these rules.

Word representations trained on morphologically annotated data were introduced by Cotterell and Schütze (2015). Closest to our approach, Schütze (1993) learned representations of character four-grams through singular value decomposition, and derived representations for words by summing the four-grams representations.

Very recently, Wieting et al. (2016) also proposed to represent words using character n-gram count vectors. However, the objective function used to learn these representations is based on paraphrase pairs, while our model can be trained on any text corpus.

1. **Character level features for NLP**

Character-level models for natural language processing discard the segmentation into words and aim at learning language representations directly from characters. A first class of such models are recurrent neural networks, applied to language modeling, text normalization, part-of-speech tagging and parsing. Another family of models are convolutional neural networks trained on characters, which were applied to part-of-speech tagging, sentiment analysis , text classification and language modeling. Sperr et al. (2013) introduced a language model based on restricted Boltzmann machines, in which words are encoded as a set of character n- grams. Finally, recent works in machine translation have proposed using sub-word units to obtain representations of rare words.

1. **Model**

A model to learn word representations with considering morphology is proposed. We model morphology by considering sub-word units and representing words by a sum of its character n-grams. And we present our sub-word model and eventually describe how to handle the dictionary of character n-grams.

* 1. **General model (Skip-gram + negative sampling )**

We start by briefly reviewing the continuous skip-gram model, from which our model is derived. Given a word vocabulary of size W, where a word is identified by its index w ∈ {1,...,W}, the goal is to learn a vectorial representation for each word w. Word representations are trained to *predict* wellwords that appear in its context. More formally, given a large training corpus represented as a sequence of words w1, ..., wT , the objective of the skip-gram model is to maximize the following log-likelihood:

**텍스트이(가) 표시된 사진

자동 생성된 설명**

For now, let us consider that we are given a scoring function s which maps pairs of (word, context) to scores in R.

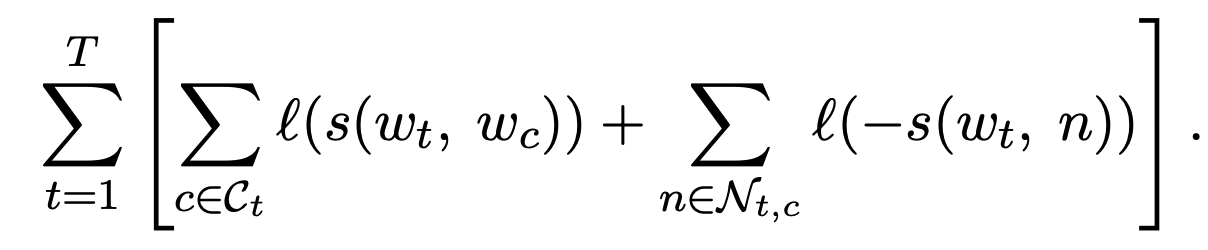
The problem of predicting context words can instead be framed as a set of independent binary classification tasks. Then the goal is to independently predict the presence (or absence) of context words. For the word at position t we consider all context words as positive examples and sample negatives at random from the dictionary.

For a chosen context position c, using the binary logistic loss, we obtain the following negative log-likelihood:

**텍스트, 손목시계이(가) 표시된 사진

자동 생성된 설명**

where Nt,c is a set of negative examples sampled from the vocabulary. By denoting the logistic loss function L: x → log(1 + e−x), we can re-write the objective as:

****

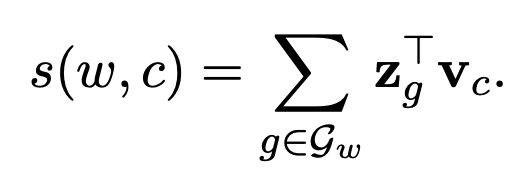
A natural parameterization for the scoring function s between a word wt and a context word wc is to use word vectors. Let us define for each word w in the vocabulary two vectors uw and vw in Rd. These two vectors are sometimes referred to as *input* and *output* vectors in the literature.

In particular, we have vectors uwt and vwc , corresponding, respectively, to words wt and wc. Then the score can be computed as the scalar product between word and context vectors as s(wt , wc ) = u⊤wt vwc . The model is the skip-gram model with negative sampling.

* 1. **Sub-word model**

we propose a different scoring function s, to consider the internal structure of words. Each word w is represented as a bag of character n-gram. We add special boundary symbols < and > at the beginning and end of words. We also include the word w itself in the set of its n-grams, to learn a representation for each word. Taking the word *where* and n = 3 as an example, it will be represented by the character n-grams: <wh, whe, her, ere, re> and the special sequence <where>. Note that the sequence <her>, corresponding to the word *her* is different from the tri-gram her from the word *where*.

Suppose that you are given a dictionary of n-grams of size G. Given a word w, let us denote by Gw ⊂ {1,...,G} the set of n-grams appearing in w. We associate a vector representation zg to each n-gram g. We represent a word by the sum of the vector representations of its n-grams. We thus obtain the scoring function:



This simple model allows sharing the representations across words, thus allowing to learn reliable representation for rare words.

1. **Results**
   1. **Human similarity judgement**

We first evaluate the quality of our representations on the task of word similarity. We do so by computing Spearman’s rank correlation coefficient between human judgement and the cosine similarity between the vector representations.

In order to provide comparable results, we propose by default to use null vectors for words that word representation can't not be obtained by (in CBOW, and Skip-Gram model). Since our model exploits sub-word information, we can also compute valid representations for out-of-vocabulary words. We do so by taking the sum of its n-gram vectors.

**테이블이(가) 표시된 사진

자동 생성된 설명**

**Table 1: Correlation between human judgement and similarity scores [6].**

When OOV words are represented using null vectors we refer to our method as sisg- and sisg otherwise (Sub-word Information Skip Gram).

First, by looking at Table 1, we notice that the proposed model (sisg), which uses sub-word information, outperforms the baselines on all datasets except the English WS353 dataset. Moreover, computing vectors for out-of-vocabulary words (sisg) is always at least as good as not doing so (sisg-). This proves the advantage of using sub-word information in the form of character n-grams.

* 1. **Word analogy tasks**

We evaluate our approach on word analogy questions, of the form A is to B as C is to D, where D must be predicted by the models. And we observe that morphological information significantly improves the syntactic tasks; our approach outperforms the baselines.

* 1. **Comparison with morphological representations**

We also compare our approach to previous work on word vectors incorporating sub-word information on word similarity tasks. The methods used are the recursive neural network of Luong et al. (2013), the morpheme cbow of Qiu et al. (2014) and the morphological transformations of Soricut and Och (2015). We also compare our approach to the log-bilinear language model introduced by Botha and Blunsom (2014). Using our model, we obtain representations of out-of- vocabulary words by summing the representations of character n-grams. We observe that our simple approach performs well relative to techniques based on sub-word information obtained from morphological segmentors.

1. **Conclusion**

we investigate a simple method to learn word representations by considering sub-word information. Our approach incorporates character n-grams into the skip-gram model. Because of its simplicity, our model trains fast and does not require any preprocessing or supervision. We show that our model outperforms baselines that do not consider sub-word information, as well as methods relying on morphological analysis.

1. **Summary**

1. CBOW & Skip-gram: These models makes training efficient because it does not perform matrix multiplication operations. And word representations using neural networks can be expressed as linear transformations as well as encoding words very clearly linguistically. (e.g., vector calculation vec ("Madrid") - vec ("Spain") + vec ("France") is as follows.)

* CBOW: we also use future words together to create a log-linear classifier with four words from the past and four words from the future as input values. And the training criterion is whether the current word is classified correctly.
* Skip-gram : each current word as an input is used to a log-linear classifier with a continuous projection layer and predicts words in a specific range before and after the current word.

2. Skip-gram + negative sampling : Skip-gram model is efficient method for learning high-quality distributed vector representations that capture many syntactic and semantic word relationships. Subsampling of frequently used words enables speed up and more regular word expressions. And negative sampling is an alternative to instead of the hierarchical SoftMax. It is demonstrated that these representations exhibit linear structure that makes precise analogical reasoning possible. Namely, the word vectors can be “meaningfully” combined using just simple vector addition.

* Negative sampling : the task is to distinguish the target word wO from draws from the noise distribution Pn(w) using logistic regression, where there are k negative samples for each data sample. The difference between the NEG and NCE is that NCE needs both samples and the numerical probabilities of the noise distribution, while NEG uses only samples.

3. GLoVe: The model using matrix factorization methods (ex. LSAs) represents words globally, but it does poorly on the word **similarity measure**. The model using local context window methods (ex. Skip models) represents words locally, but it does poorly utilize the statistics of the corpus because it can’t operate directly on **the co-occurrence statistics**.

The author of the paper comes up with a model that will borrow only the advantages of these two methods. It's Glove, the Global Vector. This model has global co-occurrence statistics information of matrix factorization and vector expression of skip gram. As a result, the word-word co-occurrence matrix learns only from nonzero values efficiently, creating meaningful vector spaces for words that make good use of statistics and perform better than conventional models.

4. FastText: A new approach is needed, rather than ignoring the **morphology** of words by assigning individual vectors to each word. This is very simple method to learn word representations by considering sub-word information. The approach incorporates character n-grams into the skip-gram model. We associate a vector representation zg to each n-gram g. We represent a word by the sum of the vector representations of its n-grams. We thus obtain the scoring function.

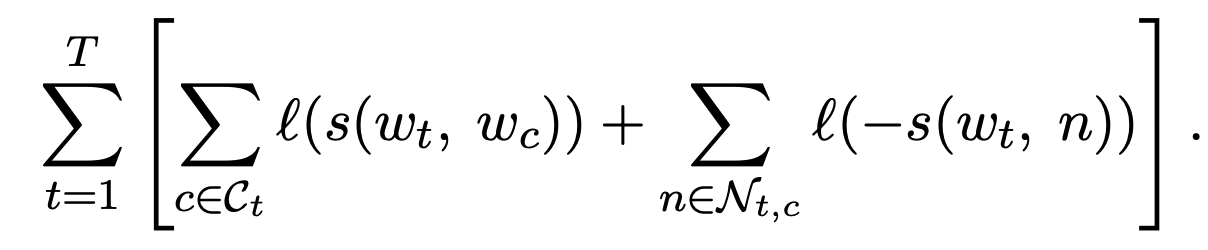
**## Difference between Skip-gram model and Fasttext**

The optimization criterion is the same, the difference is how the model gets the word vector. Fasttext optimizes the same criterion as the standard skip-gram model. More formally, given a large training corpus represented as a sequence of words w1, ..., wT , the objective of the skip-gram model is to maximize the following log-likelihood

**텍스트이(가) 표시된 사진

자동 생성된 설명**

with all the approximation tricks that make the optimization computationally efficient. In the end, they get this:

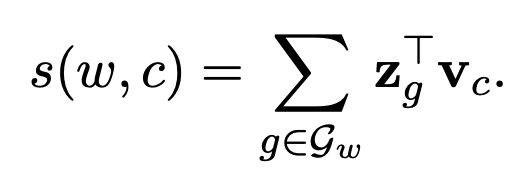
****

There is a sum over all words wc and approximate the denominator using some negative samples n.

The crucial difference is in the function s. In the original skip-gram model, it is a dot product of the two words embeddings.(The score can be computed as the scalar product of word and context vectors as s(wt ,wc ) = u⊤wt vwc )

However, in the FastText, the score function s  is redefined:

Suppose that you are given a dictionary of n-grams of size G. Given a word w, let us denote by Gw ⊂ {1,...,G} the set of n-grams appearing in w. We associate a vector representation zg to each n-gram g. We represent a word by the sum of the vector representations of its n-grams. We thus obtain the scoring function:



Word wt is represented as a sum of all n-grams zg the word consist of plus a vector for the word itself. You basically want to make not only the word, but also all its substrings probable in the given context window.

**## Additional description of Negative Sampling**

Negative sampling is a way for the skip-gram model to focus on some word sets rather than the entire word set during the learning process. For example, let's say the surrounding words that we're focusing on are cats and cute. Here, some non-peripheral words that are randomly selected from a set of words such as 'computer' and 'room' are imported. So for one central word, you create a set of words that are much smaller than the entire set of words and convert the last step into a binary classification problem. Labeling surrounding words positive and randomly sampled words negative results in a dataset for binary classification problems. This is more efficient than the simple skip-gram model, where multiple classification problems were solved with choices as large as the existing word set.

**REFERENCES**

[1] Efficient Estimation of Word Representations in Vector Space, Mikolov et al., ICLR Workshop 2013.

[2] Distributed Representations of Words and Phrases and their Compositionality, Mikolov et al., NeurIPS 2013.

[3] Michael U Gutmann and Aapo Hyva ̈rinen. Noise-contrastive estimation of unnormalized statistical mod- els, with applications to natural image statistics. *The Journal of Machine Learning Research*, 13:307, 2012.

[4] Richard Socher, Cliff C. Lin, Andrew Y. Ng, and Christopher D. Manning. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*, volume 2, 2011.

[5] GloVe: Global Vectors for Word Representation, Pennington et al., EMNLP 2014.

[6] Enriching Word Vectors with Sub-word Information, Bojanowski et al., TACL 2017.

Efficient estimation of word Representations in vector space review : <https://coshin.tistory.com/43>

Distributed Representations of Words and Phrases and their Compositionality review : <https://coshin.tistory.com/44?category=949609>

Glove review : <https://www.sallys.space/blog/2018/04/30/glove/>

FastText review : <https://deepdaiv.oopy.io/paper/fasttext#083621eb-9c08-4034-bd61-e8efede4daa4>

Negative sampling **:** [**https://wikidocs.net/69141**](https://wikidocs.net/69141)

What is the difference between fastext skipgram and word2vec skipgram? : [**https://stackoverflow.com/questions/55700359/whats-the-different-between-fasttext-skipgram-and-word2vec-skipgram**](https://stackoverflow.com/questions/55700359/whats-the-different-between-fasttext-skipgram-and-word2vec-skipgram)