The estimation of stability of semantic space generated by word embedding algorithms

Amirzhan Sanzhar
Institute of Information and Computational Technologies
Almaty, Kazakhstan
amir.sanzhar@gmail.com

Alexander Pak
Institute of Information and Computational Technologies
Almaty, Kazakhstan
aa.pak83@gmail.com

Jaxylykova Assel Bulatovna
Institute of Information and Computational Technologies
Almaty, Kazakhstan
aselya17.89@mail.ru

***Abstract*—Vector representation of words plays a major role in the natural language processing. As a fundamental unit, the vector representation is further used to solve applied problems: text classification, textual entailment, named-entity recognition. In recent times was created several models to produce word embedding vectors which uses different approaches. However, they suffer from accidents while training, such as random initialization of weights, the random order of the examples. Therefore, it is impossible to reproduce the result, and repeated experiments using the same dataset and algorithms lead to various close results. In this work, we presented methods for estimating the ”dissimilarity” of the semantic spaces built by the algorithms of word embeddings and give mathematical intuition about influence of various randomness on the structure of semantic spaces.**

***Index Terms*—embeddings, semantic, word2vec, Kullback-Leibler divergence, diffusion distance**

I. Introduction

Natural language processing is one of the key application field in artificial intelligence. This processing allows to perform a wide range of practical tasks from the classification of texts to the construction of dialogue systems. In the last few years, solutions to the problems of natural language processing based on neural networks to distributed representations of words have been proposed. However, each of the following approaches has its own advantages and disadvantages. In classical models of word processing features used words encoded by ”one-hot” method, each word in the dictionary is represented as a vector whose size is equal to the number of words in the dictionary. All elements of a vector except one are zero, and an element in the position corresponding to the word number in the dictionary is one. The proposed approach has a number of drawbacks as there are very sparse vector representations. In consequence of which such ideas are not able to catch the similarities between the words. To improve the ”one-hot” method we used the distributive hypothesis. The distributive hypothesis states that words with similar meanings tend to occur in a similar context. According to this hypothesis, each word can be represented as a distributed representation, a vector of real numbers. Such a vector, being an element of Euclidean space for some dimension R^d, is fed to the input of models. The assumption of this idea is that the geometric relations in space R^d will correspond to the semantic relations between words. For example, the nearest neighbors of a word in this space will be its synonyms or other words similar in meaning to the subject. One of the modern methods for obtaining a distributed representation of words are Word2Vec models. The idea was proposed by Mikolov and his co-authors in work[1] with two different neural network architectures: in the form of a continuous bag of words (CBOW) and in the form of a skip-gram architecture[2]. The main purpose of these methods is to store as much information as possible in the vector of the word while maintaining a smaller dimension. The CBOW model calculates the conditional probability of the target word from the context words that surround it in the fixed window. While, the skip-gram model performs the reverse operation: predicts the surrounding context words by a given central target word. Comparing these two architectures, we can note that the CBOW model is trained faster than the skip-gram, gives greater accuracy for frequent words, in turn, the skip-gram model is more suitable for training on small data, gives greater quality of training for rare words. As the embedding dimension increases, the prediction accuracy also increases to saturation at some point, which is chosen as the optimal embedding dimension. Word2Vec model maximizes the logarithmic likelihood of occurrence of the context for the central word and calculates word vectors using stochastic gradient descent. The functionality of Word2Vec models can be represented as:

$$\sum_{t=1}^{T} \log p(w_t|C_t), \tag{1}$$

, where w_t - vector of the center word, C_t - set of input vectors.

The main advantage of the distributed representation is its ability to capture the similarity between words. Measurement of similarity between vectors is possible, for example, using cosine distance. But Word2Vec model takes into account only local distribution, not uses global statistics as frequency.

Alternative method presented by Sotcher[3] uses a statistical approach. Word embeddings obtained by using singular value decomposition method from co-occurrence matrix. Key idea is to approximate the logarithm of the co-occurrence by multiplying the vectors of words. However, this method is computationally expensive. Thus, using statistical data, you can get a vector for each word from the dictionary. The functionality of the GloVe model:

$$\sum_{i,j} f(X_{i,j})(w_i^T \tilde{w}_j - \log X_{i,j})^2, \tag{2}$$

, where $X_{i,j}$ - co-occurrence matrix, $f(x)$ - the weighting function, $w, \tilde{w}$ - word vectors.

This algorithm uses global word statistics to construct a vector representation of words, although it does not use valuable information about local contexts and distributions.

One more method for obtaining a word embedding vector is FastText model created by a research team from Facebook Research[4]. The main difference is that Word2Vec algorithm builds a vector representation for the word as a whole, whereas FastText searches for vectors for n-grams that are inside each word. Consequently, each word is the sum of n-grams vectors. This approach has a number of advantages like finding the best vector representation for rare words, and most importantly can build a vector for a word from its n-grams even if the word is not in the dictionary, while Word2Vec and GloVe models cannot do it. But on an intuitive level, this approach has a drawback, since it builds a vector representation for n-grams, rather than for statistically significant morphemes that convey the basic semantic and grammatical properties of the word.

All these models have one more common drawback - assigning only one vector to each word. This means that if the word has several different meanings at once, which is very often the case, the contexts of different meanings will be averaged, so the existing models of nested words do not take into account the phenomenon of *homonymy*, when several meanings correspond to the same word[5][6].

Randomness in model training leads to a lack of reproducibility, as re-training using the same data sets and model parameters leads to different "similar" results[7][8]. Therefore, it is impossible to match vectors from different trained models. Moreover, to expand the dictionary, you have to re-train the model with new examples, which is a resource-intensive process. Therefore, methods of stabilization of training models, today have a high applied importance.

In this paper, for numerical results was used Word2Vec model.

II. Stability

We propose the following definition of stability of semantic spaces: $p(||x|| < r) > p_{cutoff}$, where one should understand as differences between two representation of the same word in two different word2vec models $x = w_{1,i} - w_{2,i}$; $w_{k,i}$ is k model, r is a radius of n-dimensional sphere, which bounds the word in it; p_{cutoff} is the level of confidence.

III. Metric and Divergence

The Diffusion distance[9] and the Kullbak-Leibler Divergence were chosen as a metric to assess the stability of the Word2Vec model.

A. Diffusion distance

The Diffusion distance simulates the difference between two histograms as a temperature field and considers the diffusion process. Then the integration of a norm on the diffusion field in time is used as a measure of dissimilarity between histograms. For the efficiency of calculations, a Gaussian pyramid is used to discretize the continuous diffusion process. The diffusion distance is then defined as the sum of the norms over all the layers of the pyramid. Consider 1-D distributions $h_1(x)$ and $h_2(x)$. Instead of calculating $d(x) = h_1(x) - h_2(x)$, consider it as isolated temperature field $T(x,t)$ at time $t = 0$, $T(x,0) = d(x)$. The temperature in an isolated field obeys the heat diffusion equation:

$$\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2} \tag{3}$$

Initial condition $T_0(X)$:

$$T(x,0) = T_0(x) = d(x) \tag{4}$$

Heat diffusion equation has unique solution:

$$T(x,t) = T_0(x) * \phi(x,t) \tag{5}$$

where $\phi(x,t)$ is the Gaussian filter

$$\phi(x,t) = \frac{1}{(2\pi)^{1/2}t}\exp(-\frac{x^2}{2t^2}) \tag{6}$$

When t increases $T(x,t)$ becomes zero everywhere, because the mean of the difference field is zero. Hence, $T(x,t)$ can be interpreted as a process of histogram value exchange, which makes $h_1(x)$ and $h_2(x)$ equivalent. Measure of this process can be used as value of difference between two histograms. Dissimilarity measure between $h_1(x)$ and $h_2(x)$ is defines as:

$$\hat{K}(h_1,h_2) = \int_0^{\bar{t}} k(|T(x,t)|)dt \tag{7}$$

, where $\bar{t}$ is a integration constant upper bound. $k(.)$ is a norm, and L_1 norm used due to its performance and cheap calculation process.

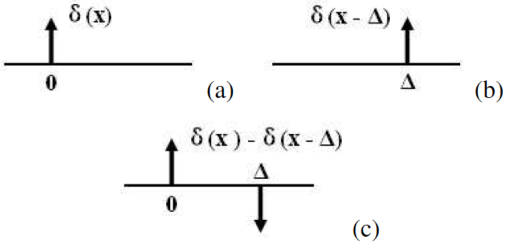

Fig. 1. . Difference between two histograms with shift. (a) h_1. (b) h_2. (c) $d = h_1 - h_2$, adopted from [9].

As mentioned in [9], assume $h_1(x) = \delta(x)$ and $h_2(x) = \delta(x - \Delta)$. This means that histogram is shifted by $\Delta \geqslant 0$. Therefore, $T_0 = \delta(x) - \delta(x - \Delta)$. The diffusion process becomes:

$$T(x,t) = (\delta(x) - \delta(x-\Delta)) * \phi(x,t) = \phi(x,t) - \phi(x-\Delta,t) \tag{8}$$

Direct computation of $\hat{K}$ is expensive. Instead, [9] offered an alternative distance function based on the Gaussian pyramid. The Gaussian pyramid is a natural and efficient discretization of the continuous diffusion process $T(x,t)$. Diffusion distance $\hat{K}(h_1,h_2)$ calculates as:

$$K(h_1,h_2) = \sum_{l=0}^{L} k(|d_l(x)|) \tag{9}$$

, where $d_0(x) = h_1(x) - h_2(x)$, $d_l = [d_{l-1}(x) * \phi(x,\sigma)] \downarrow_2$ $l = 1,\dots,L$ are different layers of the pyramid. $\downarrow_2$ denotes half size down sampling. L - is the number of pyramid layers, σ - standard deviation for the Gaussian filter ϕ. As long as $k(.)$ is a metric, as metric was chosen L_1 norm, equation (9) becomes:

$$K(h_1,h_2) = \sum_{l=0}^{L} |d_l(x)| \tag{10}$$

As alternative method for measuring dissimilarity between histograms, we can use divergence to check difference between two distributions.

B. Kullback-Leibler divergence

Divergence function $D[z : y], z, y \in S$ should satisfy the following conditions[10]:

1) $D[z : y] \geqslant 0$, where $z \neq y$
2) $D[z : y] = 0$, when and only when $z = y$
3) For small dz,

$$D[z + dz : z] \approx \frac{1}{2}\sum g_{ij} dz_i dz_j \tag{11}$$

, where S is a manifold consisting of probability distributions parameterized by z. $z = (z_1,\dots,z_n)$ - local coordinate system.

In general, a divergence is not symmetric with respect to z and y so that:

$$D[y : z] \neq D[z : y] \tag{12}$$

Example of symmetric divergence is the square of the Euclidean distance:

$$D[z : y] = \frac{1}{2}\sum |z_i - y_i|^2 \tag{13}$$

Kullback-Leibler divergence is a case of f-divergence[10]:

$$D_f[p : q] = \sum p_i f(\frac{q_i}{p_i}) \tag{14}$$

Measure the distance between two probability distributions $q(x)$ and $p(x)$ over the same x is called divergence the Kullback-Leibler. The concept is closely related to the relative entropy, and is an asymmetric measure of the difference between the two probability distributions. Kullback-Leibler divergence (KL) of $q(x)$ in $p(x)$ denotes as $D_{KL}(p(x), q(x))$ and denotes the amount of lost information $q(x)$ in the approximation of $p(x)$ [10].

Define $q(x)$ and $p(x)$ as two probability distributions of a random discrete quantity x. Sum of probabilities $p(x)$ and $q(x)$ equals 1, and for every $x \in X$, $p(x) > 0$ and $q(x) > 0$. Then $D_{KL}(p(x), q(x))$:

$$D_{KL}(p(x)||q(x)) = -\sum_{x \in X} p(x) \ln \frac{q(x)}{p(x)} \tag{15}$$

However, it is possible to make Kullback-Leibler divergence symmetric by following method:

$$S(P,Q) = D_{KL}(P,Q) + D_{KL}(Q,P) \tag{16}$$

$S(P,Q)$ becomes symmetric but it is not yet metric. It should be noted, that in our experiments distribution generates by the same approach and differences between two distributions are too small. As we know, if distinction is small (from the third condition), then it can be used as metric, because it locally proportional to Fisher information metric. Hence, Kullback-Leibler divergence used as metric.

IV. Dataset

For numerical results, was used standard corpus text8 of English Wikipedia dump on Mar. 3, 2006, which had been used in the many studies. text8 dataset consist 17,005,208 words and 253,855 unique words. Corpus was already prepocessed and prepared for training process.

V. Experiments

In numerical experiments, Word2Vec model was used to evaluate the stability of the model for distributed word representation. The reasons for the instability of learning are random initialization of word embedding vectors, and the order in which these examples are processed. Therefore, three types of randomness were tested:

- the procedure with different order of examples and with the same initialization vectors;
- with random initialization and identical order of examples;
- with different order of examples and random initialization of vectors.

For all models used Word2Vec default hyperparameters, where word embedding size is 200 [11] [12].

For each of these three cases of randomness was trained 20 couples of Word2Vec models. So, totally for each randomness case was generated 40 models.

Models were compared based on the distance between words vectors embeddings, than histograms were built in the following way:

1) for each randomness case train 20 couple of embeddings models;
2) define m_1 and m_2 as models from one iteration;
3) for each word w_i in vocabulary:
calculate L_2 norm between word embeddings of m_1 and m_2;

4) collect L_2 norms in the same iteration and build histogram;

After that, calculate differences between all generated histograms, within particular randomness case, using Diffusion distance and Kullbak-Leibler divergence. Since we made several experiments, was built boxplot for each randomness case in ”Fig.2” and ”Fig.3”.

Moreover, the estimation of ”dissimilarity” in terms of L_2 norms, we should measure cosine angle as differences in directions. Algorithm for above estimation is following:

1) use trained coulpe of Word2Vec models m_1 and m_2;
2) calculate cosine angle of the same word between m_1 and m_2 models;
3) collect cosine angle values and build histogram;

In ”Fig.4” showed boxplot of cosine angle values for each randomness type.

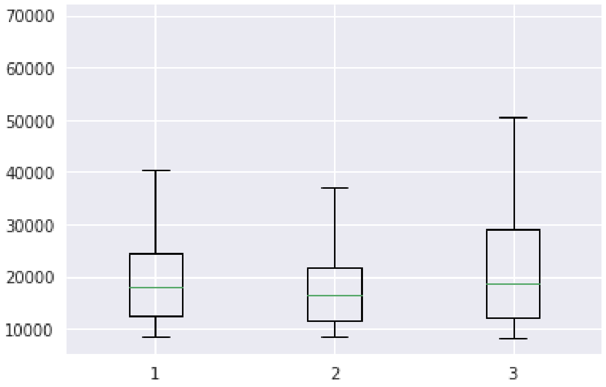

Fig. 2. . The range of values of Diffusion metric (Y-axis) for each of the randomness case (X-axis). 1) model with different order of samples. 2) model with random vector initialization. 3) with random vector initialization and different order of examples.

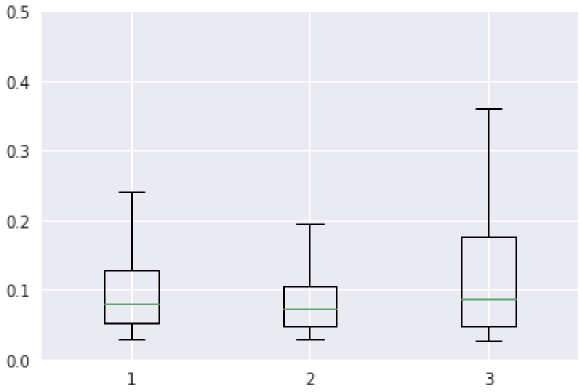

Fig. 3. The range of values of Kullback-Leibler divergence (Y-axis) for each of the randomness case (X-axis). 1) model with different order of samples. 2) model with random vector initialization. 3) with random vector initialization and different order of examples.

VI. Discussion

According to obtained results, in our opinion different order in which these examples are processed has less contribution to randomness, compared to random initialization of word embedding vectors. These results intuitively interpreted better by Diffusion distance metric, and this result may be used to try stabilize training process of models.

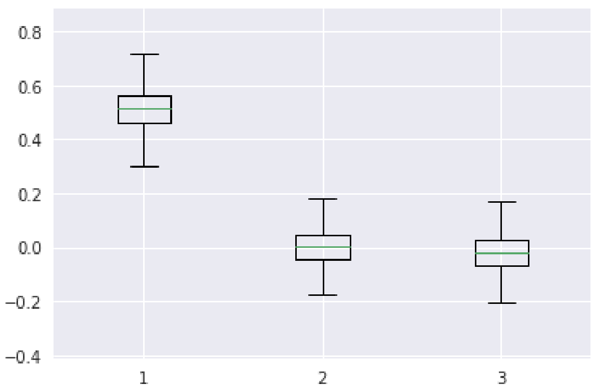

Fig. 4. The range of values of cosine angle (Y-axis) for each of the randomness case (X-axis). 1) model with different order of samples. 2) model with random vector initialization. 3) with random vector initialization and different order of examples.

VII. Conclusion

In this paper, was considered three types of randomness during training process of word embedding models. Dissimilarity of produced word embedding vectors in each type of randomness measured by metrics: Diffusion distance and Kullback-Leibler divergence. Training dataset is the database of medical texts from public sources in Russian language. Diffusion distance as a metric better suited, because it is intuitively understandable and shows how much work needed to make two histograms similar. In the future work will be tested other word embedding algorithms and will be proposed methods for stabilization of training models.

Acknowledgment

We gratefully acknowledge financial support of Institute of Information and Computational Technologies. In addition, thankful to Nikolai Makarenko for useful advice and interest in the work. (Grant AP05132760, Kazakhstan).

References

[1] Tomas Mikolov et al. “Efficient Estimation of Word Representations in Vector Space”. In: *CoRR* abs/1301.3781 (2013). arXiv: 1301.3781. URL: http://arxiv.org/abs/1301.3781.

[2] Ian H Witten et al. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.

[3] Jeffrey Pennington, Richard Socher, and Christopher D Manning. “Glove: Global Vectors for Word Representation.” In: *EMNLP*. Vol. 14. 2014, pp. 1532–1543.

[4] Armand Joulin et al. “Bag of Tricks for Efficient Text Classification”. In: *CoRR* abs/1607.01759 (2016). arXiv: 1607.01759. URL: http://arxiv.org/abs/1607.01759.

[5] Dirk Weissenborn. “Reading Twice for Natural Language Understanding”. In: *arXiv preprint arXiv:1706.02596* (2017).

[6] Omer Levy and Yoav Goldberg. “Dependency-Based Word Embeddings”. In: *ACL*. 2014.

[7] Johannes Hellrich and Udo Hahn. “Don’t Get Fooled by Word Embeddings-Better Watch their Neighborhood”. In: *DH*. 2017.

[8] Johannes Hellrich and Udo Hahn. “Bad Company - Neighborhoods in Neural Embedding Spaces Considered Harmful”. In: *COLING*. 2016.

[9] Haibin Ling and K. Okada. “Diffusion Distance for Histogram Comparison”. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*. Vol. 1. June 2006, pp. 246–253. DOI: 10.1109/CVPR.2006.99.

[10] Shun-ichi Amari. “Divergence function, information monotonicity and information geometry”. In: *Workshop on Information Theoretic Methods in Science and Engineering (WITMSE)*. Citeseer. 2009.

[11] Franziska Horn. “Context encoders as a simple but powerful extension of word2vec”. In: *arXiv preprint arXiv:1706.02496* (2017).

[12] Jie Shen and Cong Liu. “Improved Word Embeddings with Implicit Structure Information”. In: *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 2016, pp. 2408–2417.