# Probabilistic approach for embedding arbitrary features of text

Anna Potapenko

National Research University Higher School of Economics
anna.a.potapenko@gmail.com

**Abstract.** Topic modeling is usually used to model words in documents by probabilistic mixtures of topics. We generalize this setup and consider arbitrary features of the positions in a corpus, e.g. “contains a word”, “belongs to a sentence”, “has a word in the local context”, “is labeled with a POS-tag”, etc. We build sparse probabilistic embeddings for positions and derive embeddings for the features by averaging of those. Importantly, we interpret the EM-algorithm as an iterative process of intersection and averaging steps that reestimate position and feature embeddings respectively. With this approach, we get several insights. First, we argue that a sentence should not be represented as an average of its words. While each word is a mixture of multiple senses, each word occurrence refers typically to just one specific sense. So in our approach, we obtain sentence embeddings by averaging position embeddings from the E-step. Second, we show that Biterm Topic Model (Yan et. al, 2013) and Word Network Topic Model (Zuo et. al, 2016) are equivalent with the only difference of tying word and context embeddings. We further extend these models by adjusting representation of each sliding window with a few iterations of EM-algorithm. Finally, we aim at consistent embeddings for hierarchical entities, e.g. for word-sentence-document structure. We discuss two alternative schemes of training and generalize to the case where the middle level of the hierarchy is unknown. It provides a unified formulation for topic segmentation and word sense disambiguation tasks.

**Keywords:** Topic models · Word embeddings · EM-algorithm.

## 1 Introduction

Recently there have been proposed a lot of frameworks for embedding short pieces of text: [6, 8, 2] to name a few. Many of them suggest that a sentence embedding is represented as an average of its word embeddings. It can be done on top of pre-trained word embeddings [2] or enforced during training [8]. However, averaging implies bringing all uncertainty of individual word senses into a fairly smooth combination. For instance, consider the clear concept conveyed by the phrase ”support vector machine”. Ideally, this embedding should be much more specific (e.g. sparse) then the embeddings of its constituent words.

To resolve this inconsistency, let us start with a question: ”What are the smallest units of language, such that all other units could be expressed using them?”. Arora [1] argues that those units are so called *atoms of discourse*, that represent *what is talked* *about in the current moment*. They show that each word can be expressed as a linear combination of several (e.g. $k = 5$) atoms, and that every atom can be interpreted as a sense of a word, thus providing promising results for polysemy. They have 2000 atoms in their experiments, which makes the atoms rather generic, thus leaving it unclear whether the model has enough granularity.

We are interested in investigating those smallest units for several reasons: 1) it would shed some light on the structure of the embedding space, 2) it could be used for inducing more sparsity and interpretability of the embeddings, 3) it would give the right building blocks to embed more complicated unites, such as sentences.

Our claim is that these units should be *individual word occurrences* in the corpus. Then the embeddings for other entities will naturally come out as some aggregates. E.g. *word embedding* is an average of all occurrences of this word, while *sentence embedding* is an average of all word-occurrences within this sentence. With this approach, both words and sentences are made of the *word-occurrence* embeddings, and thus neither of them is a “sub-embedding” of the other.

In this work we develop a framework for consistent embedding of arbitrary text features. The framework is based on topic modeling and provides probabilistic embeddings with interpretable components.

## 2 EM-algorithm as intersection-averaging in embeddings space

Probabilistic Latent Semantic Analysis (PLSA, [4]) explains observed words in documents with a mixture of latent topics. Training is performed with EM-algorithm, where E-step computes posterior probabilities of topics for each word in each document, and M-step performs maximum likelihood updates for the parameters, namely for probabilities of words in topics and for probabilities of topics in documents. In this section we provide a new interpretation of this algorithm as an iterative intersection-averaging procedure in the embedding space of arbitrary text features.

Let us denote word embeddings with $\phi_w$, document embeddings with $\theta_d$, and embeddings for positions in the corpus with $\psi_i$ ($i$ runs from 1 to the length of the corpus $N$). We define the embedding space as a probability simplex $\{x \in \mathbb{R}^k : \sum_{j=1}^k x_j = 1;\ \forall j\, x_j \geq 0\}$. Thus we can interpret each dimension as a *topic*, and elements of the vectors as topic probabilities. The following two lemmas represent formulas for the EM-algorithm in our notation, providing a new intuition behind the updates.

**Lemma 1.** *E-step performs* a soft intersection of topics *in word and document embeddings related to position* $i$*:*

$$\psi_i = \frac{1}{Z_i}\, \tau \circ \phi_{w_i} \circ \theta_{d_i}, \tag{1}$$

*where* $\tau = \left(\frac{1}{p(t_1)}, \dots, \frac{1}{p(t_k)}\right)$ *is a vector of inverted topic probabilities in the corpus,* $\circ$ *denotes an element-wise product, and* $Z_i$ *is a partition function.*

As a corollary, it’s seen that position embeddings $\psi_i$ are more sparse (specific) then the corresponding feature embeddings $\phi_{w_i}$ and $\theta_{d_i}$. This also follows an intuition, since positions are more specific objects then words or documents. In future, we will denote the weighted element-wise product (1) as a $mult$ operation.

**Lemma 2.** *M-step performs* averaging *of position embeddings:*

$$\phi_w = \frac{1}{\sum_i [w_i = w]} \sum_{i:w_i=w} \psi_i; \quad \theta_d = \frac{1}{\sum_i [d_i = d]} \sum_{i:d_i=d} \psi_i, \tag{2}$$

*where squared brackets denote the indicator function.*

As a corollary, it's seen that feature embeddings $\phi_{w_i}$ and $\theta_{d_i}$ are more smooth (generic) then the corresponding position embeddings. Let us denote element-wise averaging (2) as a $mean$ operation.

***Note about equivalence relations.*** Let us consider two *equivalence relations* for the set $S = \{1, 2, \ldots, N\}$ of all positions in the corpus. The first relation $\mathcal{W}$ considers two positions equivalent if they hold the same words. The second relation $\mathcal{D}$ considers two positions equivalent if they are contained in the same document:

$$i \sim_{\mathcal{W}} j \text{ if } w_i = w_j; \qquad i \sim_{\mathcal{D}} j \text{ if } d_i = d_j.$$

Then $S/\sim_{\mathcal{W}}$ is the vocabulary and $S/\sim_{\mathcal{D}}$ is the set of all documents. The M-step of the algorithm computes embeddings for every equivalence class. The E-step computes embeddings for every equivalence class with respect to the *intersection* of the defined relations $\sim_{\mathcal{W}} \cap \sim_{\mathcal{D}}$. It means that the choice of the equivalence relations $\mathcal{W}$ and $\mathcal{D}$ *defines* what are the smallest unites distinguished by the algorithm.

***Arbitrary text features.*** Word-in-document unit might be not fine-grained enough, so one can consider *arbitrary features of position* $i$ and straightforwardly incorporate them to the M-step instead of the document feature $d_i = d$. Here are some examples:

1. Local context: $c_i = c$, where $c_i = \{w_{i-h}, \ldots, w_{i+h}\}$ or some other local neighbourhood of the position $i$.
2. Sliding window: $j - h \leq i \leq j + h$, where $h$ is the window size.
3. Arbitrary annotation: $\text{tag}_i = \text{tag}$ by some system of tagging, e.g. POS-tags, grammar roles, types of named entities, WordNet nodes, etc.
4. Additional modality: $a_i = a$, e.g. authors of each token in call center dialogues.

Some of these features are commonly used in NLP tasks, e.g. as an input to CRFs. In this work we claim that they can also be freely used in topic modeling.

## 3 Incorporating local contexts and hierarchies

A straightforward way of using local context features (1) in the EM-algorithm, would require to learn parameters for all unique $2h$-words contexts, which is not feasible. Thus one might suggest to reparametrize context embeddings as an average of individual contexts. This will amount to Word Network Topic Model (WNTM), proposed in [12]. It was initially considered as a modification of PLSA for modeling word co-occurrences in short text analysis.

One could go further and notice that in this case $\Phi$ and $\Theta$ matrices have the same dimension and represent *word and context embeddings* respectively. According to modern literature [10, 5], it might be beneficial to tie input and output embeddings, thus reducing the parameter space.

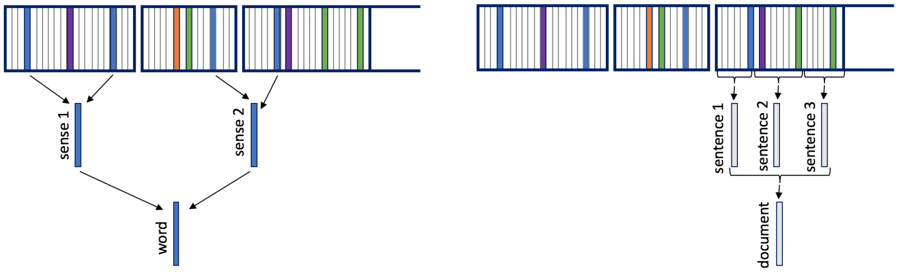

**Fig. 1.** Consistent embeddings for hierarchical structures.

**Lemma 3.** *Condition $\Theta = \Phi$ set in initialization of Word Network Topic Model is preserved with EM iterations.*

We omit the proof due to space limitations and provide the final update rules:

$$\text{E-step: } \psi_i = mult(\phi_{w_i}, \underset{v \in c_i}{mean}\, \phi_v); \qquad \text{M-step: } \phi_u = \underset{i : w_i = u}{mean}\, \psi_i. \tag{3}$$

Remarkably, these update rules represent EM-algorithm for Biterm Topic Model (BTM) [11], which was initially learned with Gibbs Sampling. BTM was proposed before WNTM and studied independently. Here we highlight the close relatedness of two models and confirm this by their equal performance in the experiments section.

***Fitting window embeddings.*** Incorporating sliding window feature (2) from the list above, would amount to the following EM-updates:

$$\text{E-step: } \psi_i = mult(\phi_{w_i}, \theta_i); \qquad \text{M-step: } \theta_i = \underset{i-h \le j \le i+h}{mean}\, \psi_j,\ \phi_u = \underset{i : w_i = u}{mean}\, \psi_i. \tag{4}$$

With a naive approach, one would need to learn $\theta$ vectors for every position in the corpus. With a modification of *online EM-algorithm* [9], this can be done on the fly. For this, we propose to compute $\theta$ vectors as a moving average and never store them. Updates (4) suggest to embed local context by averaging specific word position embeddings $\psi_i$. BTM updates (3) are remarkably similar, but average ambiguous word embeddings $\phi_u$ instead. We argue that this is an important difference and prove this empirically for a more simple case. Namely, we compare sentence embeddings obtained by averaging of word embeddings vs fitted position embeddings.

***Embedding hierarchical structures.*** To build consistent representations of hierarchical text features, let us consider a *word* embedding as an average of the *word sense* embeddings, which are the averages of the corresponding *word-occurrence* embeddings (see Fig. 1). Similarly, a document embedding is an average of its paragraph embeddings, which are the averages of the word-occurrence embeddings. With this approach, both *word sense induction* and *topic segmentation* can be viewed as a task of determining the middle level of the hierarchy. To provide hierarchical consistency, we sketch two alternative models. Likelihood maximization for a corpus of word-sentence-document triplets under conditional independence assumptions $p(w, s, d|t) = p(w|t)p(s|t)p(d|t)$ leads to a generalized E-step that computes an element-wise product of word, sentence, and document embeddings:

$$\psi_i = \frac{1}{Z_i}\ \alpha^2 \circ \phi_{w_i} \circ \zeta_{s_i} \circ \theta_{d_i} \tag{5}$$

An alternative is to view data as two independent corpora of word-sentence and sentence-document pairs. This leads to two separate EM-procedures that share sentence embeddings. One can show that the update for position embeddings amounts exactly to (5), but the iteration scheme differs.

## 4 Experiments

***On equivalence of BTM and WNTM.*** Table 1 shows an equal performance of BTM and WNTM on a range of word similarity tasks. Both models were trained on Wikipedia 2016-01-13 dump with open-source topic modeling library BigARTM[1]. Word similarity was obtained as the dot product of $\phi$ vectors. Refer to [9] for any details, since we fully follow their setup. Fitting window embeddings with (4) is left for future work.

***Sentence embeddings though positions.*** We closely follow the setup from [3] to evaluate sentence embeddings for semantic textual similarity tasks, namely SICK-2014 and STS-2014. For SICK dataset we train a linear model on top of the embedding features with SentEval tool[2]. With the topic modeling framework [7], we prepare averaging of word embeddings (BOW) and averaging of position embeddings (fitted). Position embeddings are obtained with 10 passes of EM-algorithm for the target sentences given fixed $\phi$ vectors. In both cases, we use word vectors pre-trained on Wikipedia. The important take-away from table 2 is that position fitting improves performance on all datasets, thus justifying our argument on consistent sentence representations. Our approach produces sentence embeddings that perform on par with word2vec baseline, being at the same time more interpretable and sparse as shown in [9].

## 5 Conclusions

In this work we interpreted EM-algorithm for learning a topic model as a process of intersection-averaging steps in the embedding space. This gave us a number of new insights: (1) how to use topic modeling for embedding arbitrary text features; (2) how to obtain consistency between representation of word occurrences, word senses, words, sentences, and documents; (3) how to generalize and improve local context topic models. In the experiments, we confirmed that sentence embeddings should be built from *word-in-document embeddings* rather than from more ambiguous *word embeddings*, and also showed the equivalent performance of WNTM and BTM. However, many other interesting outcomes of our theory, such as building hierarchies of embeddings or representing syntactic role features, were left for future work.

[1] bigartm.org

[2] github.com/facebookresearch/SentEval

**Table 1.** Spearman correlation of BTM and WNTM embeddings on word similarity tasks.

| Model | WS-353 Sim | WS-353 Rel | WS-353 All | SimLex<br>Hill et al. | MEN<br>Bruni et. al | RareWords<br>Luong et al. | Radinsky<br>M. Turk |
|---|---|---|---|---|---|---|---|
| BTM | **0.68** | **0.59** | **0.61** | 0.24 | 0.65 | 0.32 | 0.54 |
| WNTM | 0.67 | 0.58 | 0.60 | 0.24 | **0.66** | **0.33** | **0.55** |

**Table 2.** Sentence embeddings performance for unsupervised tasks. We report Person/Spearman correlations for SICK relatedness and STS-2014 datasets and accuracy for SICK entailment.

| Model | STS-2014 | | | | | | SICK | |
|---|---|---|---|---|---|---|---|---|
| | Forum | News | Headlines | Images | Tweets | Average | Rel | Ent |
| BOW (ours) | 0.41/ 0.42 | 0.70/ 0.62 | 0.60/ 0.53 | 0.76/ 0.71 | 0.68/ 0.63 | 0.64/ 0.60 | 0.77/ 0.70 | 76.27 |
| Fitted (ours) | **0.45/ 0.46** | **0.70/** 0.62 | 0.61/ 0.55 | **0.76/** 0.71 | 0.68/ 0.62 | **0.65/** 0.62 | 0.78/ **0.71** | **76.96** |
| BOW (w2v) | 0.39/ 0.46 | 0.67/ **0.66** | **0.64/ 0.60** | 0.76/ **0.72** | **0.70/ 0.69** | 0.65/ **0.65** | **0.79/** 0.69 | 75.62 |

***Acknowledgements.*** The research was supported by Russian Foundation for Basic Research (17-07-01536).

## References

1. Arora, S., Li, Y., Liang, Y., Ma, T., Risteski, A.: Linear algebraic structure of word senses, with applications to polysemy. CoRR **abs/1601.03764** (2016)
2. Arora, S., Liang, Y., Ma, T.: A simple but tough-to-beat baseline for sentence embeddings. In: International Conference on Learning Representations (2017)
3. Conneau, A., Kiela, D., Schwenk, H., Barrault, L., Bordes, A.: Supervised learning of universal sentence representations from natural language inference data. In: Proceedings of EMNLP. pp. 670–680. Association for Computational Linguistics (2017)
4. Hofmann, T.: Probabilistic latent semantic analysis. In: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence. pp. 289–296. UAI'99, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1999)
5. Inan, H., Khosravi, K., Socher, R.: Tying word vectors and word classifiers: A loss framework for language modeling. CoRR **abs/1611.01462** (2016)
6. Kiros, R., Zhu, Y., Salakhutdinov, R., Zemel, R.S., Torralba, A., Urtasun, R., Fidler, S.: Skip-thought vectors. In: Proceedings of the 28th International Conference on Neural Information Processing Systems. pp. 3294–3302. NIPS'15, MIT Press, Cambridge, MA, USA (2015)
7. Kochedykov D., Apishev M., G.L.V.K.: Fast and modular regularized topic modelling. In: Proceeding Of The 21St Conference Of FRUCT Association. ISMW. p. 182193 (2017)
8. Pagliardini, M., Gupta, P., Jaggi, M.: Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. In: Proceedings of NAACL (2018)
9. Potapenko, A., Popov, A., Vorontsov, K.: Interpretable probabilistic embeddings: Bridging the gap between topic models and neural networks. In: Filchenkov, A., Pivovarova, L., Žižka, J. (eds.) AINL. pp. 167–180. Springer International Publishing, Cham (2018)
10. Press, O., Wolf, L.: Using the output embedding to improve language models. In: Proceedings of ACL: Volume 2, Short Papers. pp. 157–163. ACL (2017)
11. Yan, X., Guo, J., Lan, Y., Cheng, X.: A biterm topic model for short texts. In: Proceedings of WWW. pp. 1445–1456 (2013)
12. Zuo, Y., Zhao, J., Xu, K.: Word network topic model: a simple but general solution for short and imbalanced texts. Knowl. Inf. Syst. **48**(2), 379–398 (2016)