Tweaks and Tricks for Word Embedding Disruptions

Amir Hazem[1] **Nicolas Hernandez**[1]

[1] LS2N - UMR CNRS 6004, Université de Nantes, France
{Amir.Hazem,Nicolas.Hernandez}@univ-nantes.fr

Abstract

Word embeddings are established as very effective models used in several NLP applications. If they differ in their architecture and training process, they often exhibit similar properties and remain vector space models with continuously-valued dimensions describing the observed data. The complexity resides in the developed strategies for learning the values within each dimensional space. In this paper, we introduce the concept of disruption which we define as a side effect of the training process of embedding models. Disruptions are viewed as a set of embedding values that are more likely to be noise than effective descriptive features. We show that dealing with disruption phenomenon is of a great benefit to bottom-up sentence embedding representation. By contrasting several in-domain and pre-trained embedding models, we propose two simple but very effective tweaking techniques that yield strong empirical improvements on textual similarity task.

1 Introduction

Word embedding models are now a standard in many NLP applications. If the choice of the most appropriate model is not always straightforward, context word representation is at the core of each model and the performance is closely related to how well the context is exploited. In this paper, we introduce the notion of disruption, a phenomenon caused by the training process of word embedding models. Disruptions are a set of extreme embedding values that are more likely to be noise than reliable features. We consider this phenomenon as a negative side effect closely related to the data and to the training optimization decisions. If we observe the Gaussian distribution of word embedding dimensional-values, we notice a set of high positive and negative values of which the percentage varies from one embedding model to another. In the context of vector-space models where each word is represented as a point in the N-dimensional Euclidean space, disruptions may drastically affect word's position and so create unbalanced embedding values. If normalization tends to smooth this effect, our experiments reveal that detecting disruptions and adjusting them is far more efficient than a standard normalization. We show that dealing with disruptions is of a substantial benefit to bottom-up sentence embedding representation.

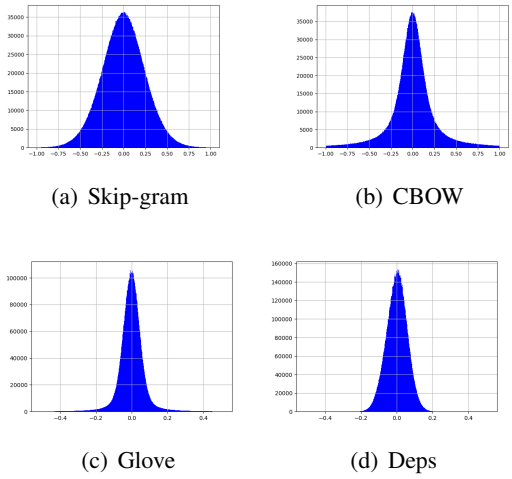
A bottom-up sentence representation is a weighted sum of the embedding vectors of its constituent words. This simple approach turned out to be very competitive in many NLP applications (Wieting et al., 2016; Arora et al., 2017) and outperformed several advanced RNNs and LSTM-based models of which the performance heavily depends on the quality and the large size of the training data set (Socher et al., 2011; Le and Mikolov, 2014; Kiros et al., 2015; Pagliardini et al., 2018). By contrast to sophisticated approaches, bottom-up sentence embedding models are less constrained and more easy to acquire. The core of the bottom-up model is the word embedding unit. An efficient sentence representation is then closely related to the quality of the used word embedding model. We state that the additive process of bottom-up sentence representation amplifies disruption's negative impact and propose to manage this phenomenon by introducing two tweaking techniques. Our approaches take into account and reduce the effect of disruptions in order to improve bottom-up sentence embedding representation. We evaluate bottom-up sen-

tence embeddings using StackExchange Philosophy data set over several in-domain [1] and pre-trained embedding models on question to question similarity task and show significant improvements. The used pre-trained models are skipgram (Sg) (Mikolov et al., 2013), Glove (Pennington et al., 2014), dependency relation (Deps) (Levy and Goldberg, 2014), and the character Skipgram (ChSg) and character CBOW (ChC)[2] (Bojanowski et al., 2016).

2 Word Embedding Disruptions

Word embedding models are vector-spaces in which words are represented as points in an N-dimensional Euclidean space. Regardless of the complexity of the embedding models, the main concern remains weights estimation. At the end of the training process, the obtained model is expected to map and to efficiently represent the training data set. This supposes that each dimension has a degree of representativeness of a given word. Which means that each single dimensional value can potentially affect the word's position in the N-dimensional space. This also means that extreme values that we call disruptions might have a bigger impact on the word's position. This phenomenon is amplified by the mathematical properties and the additive process of bottom-up representation. If we consider for instance a 3-dimensional space in which one dimensional value is drastically high, the position in the 3-D space will be attracted by this dimension. This is not problematic on its own if the 3-D model well maps the data. However, if it is not the case, this might weaken the quality of the word's embedding vector.

Multiple reasons lend support to the idea that disruptions are more likely to be side effects of the training process rather than being discriminative values. The first reason comes from the characteristics of the training data set. Regardless of the size which in most cases greatly affects the quality of the embedding models, not all words are equally distributed and even using down-sampling and other sophisticated techniques to reduce the size impact, infrequent words will always be under-characterized at least for embedding models not involving character n-gram modeling. The second reason comes from the architecture and the training procedure of word embeddings (Nematzadeh et al., 2017). CBOW model for instance, predicts the target word based on a mean average weights of its context words. While, skip-gram maximizes the average log-probability of each word's context (Mikolov et al., 2013). Also, the process of weights computation is often based on batches which leads to a mean error minimization instead of a specific attention to each single training example. This optimization process might lead to some computation decisions that create margin values, potentially not relevant and so creates dimension disruptions. Even without using batches, and in order to allow efficient training, evaluating the normalization factor of the Softmax (Mikolov et al., 2013) for instance, introduces approximations. We don't claim that one of the above cited reasons is the main cause of disruptions, however we hypothesize that several parameters may lead to training side effects that lead to disruption values. If it is difficult to remove disruptions, we propose two tweaking techniques to reduce their effects.



(a) Skip-gram (b) CBOW

(c) Glove (d) Deps

Figure 1: 300 dimensional word embedding distributions on the StackExchange Philosophy data set.

Figure 1 illustrates the distribution values of some well-known state-of-art embedding models. We first observe that all the embedding models follow a Gaussian distribution with a mean around zero. The main differences concern the standard deviation, minimum and maximum values and the density. Also, Table 1 reports the statistics of in-domain and pre-trained embeddings. We observe that each model shows different characteris-

[1] In-domain embedding models are embeddings trained on the in-domain philosophy corpus.

[2] ChC is not available, and is only used as in-domain.

	In-domain embedding models					Pre-trained embedding models						
	Sg	CBOW	Glove	ChC	ChSG	W2V	Glove6B	Glove42B	Bow5C	Bow5W	Deps	ChSG
μ	-0.01	-0.003	-0.0008	0.001	-0.011	-0.003	-0.003	0.005	-0.0002	-0.0007	0.0004	0.007
σ	0.35	0.47	0.12	0.77	0.17	0.13	0.38	0.29 2	0.05	0.05	0.05	0.295
min	-2.07	-5.51	-3.06	-8.01	-2.55	-4.06	-3.06	-5.09	-0.31	-0.358	-0.34	-10.1
max	2.20	7.57	2.57	9.98	2.26	4.18	3.52	3.25	0.32	0.354	0.29	13.6
disrupt (%)	7.56	17.3	7.11	15.8	22.1	8.1	18.3	17.1	36.6	37.9	38.4	21.5
Cove (%)	100	100	100	100	100	52.1	55.2	66.1	52.0	52.0	51.9	56.4

Table 1: Statistics over several in-domain and pre-trained embedding models on the Philosophy data set. The mean value over the entire set of embeddings (μ), its corresponding standard deviation (σ), minimum (min) and maximum (max) values, the percentage of disruptions ($disrupt$) and vocabulary coverage of the embedding models on the Philosophy data set ($Cove$).

tics which can be comparable in some cases (Sg and ChSg) or totally different in other cases such as Sg and CBOW. Also, we see that substantial dimensional values that we define as disruptions are beyond the standard deviation.

3 Tweaks Approach

In order to reduce disruption impact on bottom-up sentence representation, we introduce the mean correction trick. A tweaking process that adjusts extreme values and center them around the mean embedding value. Let's consider N the number of embedding dimensions, V the corpus vocabulary size and S the set of disruptions with $S \in N \times V$. All the S values are replaced by the mean.

Algorithm 1 Mean Tweak approach

Require: $Emb = SG, CBOW, Glove, ...$
Require: $\mu \leftarrow Mean(Emb)$
Require: $\sigma \leftarrow StandardDeviation(Emb)$
Require: $\alpha \in]min, \mu - \sigma]$
Require: $\beta \in [\mu + \sigma, max[$

function TWEAK(Emb_w,μ,α, β)
  for $i \in Dim(Emb_w)$ **do**
    if $Emb_w[i] \notin [\alpha, \beta]$ **then**
      $Emb_w[i] \leftarrow \mu$
    end if
  end for
  return Emb_w
end function

Algorithm 1 represents the mean tweak approach where the mean value is computed over the entire set of embedding models. By contrast, per dimension approach computes one mean value per dimension. In our experiments α and β were computed empirically on a development set, however, their values are often around the $\mu + \sigma$ for max and μ - σ for min intervals. In the dimensional space this procedure tends to center the words position in the space and to reduce the impact of disruptions. From the bottom-up sentence representation side, this can be interpreted as ignoring some dimensions in the additive process. As can be seen in Table 1, the mean is often around zero and this value will not have any effect on the position of the sentence in the space when we sum-up the words of a given sentence. We consider two ways of mean computation. The first is computed over the entire set of word embeddings and the second one is the computation of a mean per dimension. We contrast both techniques in our experiments.

4 Data Description

The data set was extracted from the philosophy community question answering forum StackExchange. Basically, each post is composed of a pair of questions and one or several answers and comments. Our data set contains 5.7k posts and 1.1M tokens. We took 10% of questions for dev and 10% for test set (575 questions).

Philosophy question pair example:
Q1: were there any pre-enlightenment philosophical consideration of the naturalistic fallacy or relate concept?
Q2: naturalistic fallacy was described and named by g.e. Moore at the beginning of the 20th century. but have there been pre-enlightenment philosopher who have treat the concept?

5 Experiments and Results

Similarly to (Arora et al., 2017), we evaluate the performance of the bottom-up approach on the questions similarity task. This consists in, first, computing for each question, an average sum of its embedding words and then compute the cosine similarity on the entire set of questions to

Approach	In-domain embedding models					Pre-trained embedding models						
	Sg	CBOW	Glove	ChC	ChSG	W2V(Sg)	Glove6B	Glove42B	Bow5C(CBOW)	Bow5W(Sg)	Deps	ChSG
Baseline	63.6	40.8	46.5	36.8	49.6	49.6	51.0	50.1	53.1	52.0	50.3	54.5
+Norm	60.3	52.2	54.0	48.0	50.9	50.2	51.9	51.9	52.6	51.8	49.8	55.3
+MeanAll	**63.9**	56.1	58.7	56.4	**59.7**	54.1	55.2	57.3	**59.2**	55.8	56.4	57.8
+MeanDim	63.8	**63.1**	**61.5**	**59.0**	59.0	**55.0**	58.1	**59.1**	58.4	57.8	**56.9**	59.4
+MeanALL+Norm	60.7	53.4	54.6	55.3	**59.7**	54.3	55.8	56.7	59.0	55.7	56.8	57.9
+MeanDim+Norm	60.9	60.1	59.2	58.5	58.3	54.7	**58.4**	59.0	57.6	**58.0**	56.3	**59.5**

Table 2: Results (MAP%) of bottom-up sentence representation on the test question-to-question similarity task using the Philosophy data set, 5 in-domain 300 dimensions embedding models (CBOW and Skipgram (Mikolov et al., 2013)), Glove (Pennington et al., 2014), Character n-gram CBOW (ChC) and character Skip-gram (ChSG) (Bojanowski et al., 2016) and 7 pre-trained 300 dimensions models (W2V(sg) trained on googlenews (Mikolov et al., 2013), Glove6B trained on wikipedia and Gigaword, Glove42B trained on Common Crawl (Pennington et al., 2014), Bow5C(CBOW), Bow5W(Sg) and Deps trained on wikipedia (Levy and Goldberg, 2014) and Character skipgram (ChSG) trained on wikipedia (Bojanowski et al., 2016)).

rank the target candidates. The mean average precision (MAP) is used for evaluation. We compare the raw bottom-up approach (*baseline*) to the tweaks that we introduced. As a preprocessing step, we apply the L2 norm ($+Norm$) and our two tweaking techniques that is: $+MeanAll$ for a mean computation over the entire corpus and $+MeanDim$, for a per dimension mean adjustment. We also contrast the use of the L2 norm on the top of our tweaking techniques that we refer to as $+MeanAll + Norm$ and $+MeanDim + Norm$.

Table 2 reports the evaluation of in-domain and pre-trained embeddings of the bottom-up approach. Overall, we see that whether using in-domain or pre-trained embeddings, the baseline bottom-up approach gains significant improvements while using the proposed tweaking techniques. The gain depends on the model but the margin is in most cases very important especially for in-domain CBOW and ChC models where we notice a rise of about 20 points of Map score. We also observe improvements in all pre-trained embeddings while the gain is not as important as the one observed in the in-domain sets. This can be explained by the vocabulary coverage of pre-trained embeddings which is often between 50 and 60% as shown in Table 1. If in most cases a per dimension tweak ($MeanDim$) shows better results than $MeanAll$, we observe some margin improvements using L2 norm on the top of our tweaking techniques. If L2 norm on its own improves the performance of the baseline, the results clearly show that managing disruptions is more efficient than L2 Norm.

Figure 2 contrasts different embedding dimension size of the CBOW model[3]. The figure shows the same tendency of improvements regardless of the change in dimension size. Also, it shows that our tweaking techniques are very effective to improve question similarity using a bottom-up model.

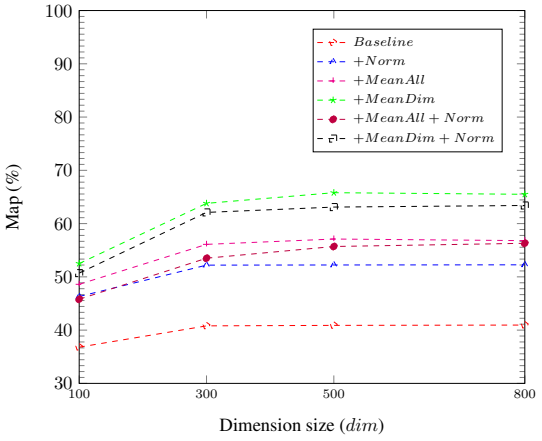


Figure 2: Contrasting several embedding dimension size of CBOW on Philosophy data set.

6 Conclusion

We introduced disruption phenomenon, a negative side effect of word embedding training process. We consider the resulting set of extreme positive and negative dimensional-values as noise and as not reliable descriptive features. To reduce the effect of disruptions on bottom-up sentence representation, we proposed two tweaking techniques. Our procedure aims at adjusting the disrupted values by smoothing them around the mean value

[3]Other models are not presented for a matter of space. Except the skipgram in-domain model, all the 10 other embedding models show the same tendency as CBOW.

of embedding dimensions. Our results over in-domain and pre-trained models showed significant improvements for question similarity task.

7 Acknowledgments

The current work was supported by the ANR 2016 PASTEL (ANR-16-CE33-0007) project[4].

References

Sanjeev Arora, Liang Yingyu, and Ma Tengyu. 2017. A simple but tough to beat baseline for sentence embeddings. In *Proceedings of the 17th International Conference on Learning Representations (ICLR'17)*. pages 1–11.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *CoRR* abs/1607.04606. http://arxiv.org/abs/1607.04606.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. Skip-thought vectors. *arXiv preprint arXiv:1506.06726* .

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. *CoRR* abs/1405.4053.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 2: Short Papers*. pages 302–308. http://aclweb.org/anthology/P/P14/P14-2050.pdf.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119.

Aida Nematzadeh, Stephan C. Meylan, and Thomas L. Griffiths. 2017. Evaluating vector-space models of word representation, or, the unreasonable effectiveness of counting words near other words. In *CogSci*.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. In *NAACL 2018 - Conference of the North American Chapter of the Association for Computational Linguistics*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. pages 1532–1543. http://www.aclweb.org/anthology/D14-1162.

Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. 2011. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. In *Advances in Neural Information Processing Systems 24*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Towards universal paraphrastic sentence embeddings. *Internationa Conference on Learning Representations, CoRR* abs/1511.08198.

[4] http://www.agence-nationale-recherche.fr/?Projet=ANR-16-CE33-0007.