Resources to Examine the Quality of Word Embedding Models Trained on n-Gram Data

Ábel Elekes   Adrian Englhardt   Martin Schäler   Klemens Böhm
Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
{abel.elekes, adrian.englhardt, martin.schaeler, klemens.Boehm}@kit.edu

Abstract

Word embeddings are powerful tools that facilitate better analysis of natural language. However, their quality highly depends on the resource used for training. There are various approaches relying on n-gram corpora, such as the Google n-gram corpus. However, n-gram corpora only offer a small window into the full text – 5 words for the Google corpus at best. This gives way to the concern whether the extracted word semantics are of high quality. In this paper, we address this concern with two contributions. First, we provide a resource containing 120 word-embedding models – one of the largest collection of embedding models. Furthermore, the resource contains the n-gramed versions of all used corpora, as well as our scripts used for corpus generation, model generation and evaluation. Second, we define a set of meaningful experiments allowing to evaluate the aforementioned quality differences. We conduct these experiments using our resource to show its usage and significance. The evaluation results confirm that one generally can expect high quality for n-grams with $n \geq 3$.

1 Introduction

Motivation. Word embedding approaches like Word2Vec (Mikolov et al., 2013b) or Glove (Pennington et al., 2014) are powerful tools for the semantic analysis of natural language. One can train them on arbitrary text corpora. Each word in the corpus is mapped to a d-dimensional vector. These vectors feature the semantic similarity and analogy properties, as follows. Semantic similarity means that representations of words used in a similar context tend to be close to each other in the vector space. The analogy property can be described by the example that ”man” is to ”woman” like ”king” to ”queen” (Mikolov et al., 2013b,a; Jansen, 2017).

These properties have been applied in numerous approaches (Mitra and Craswell, 2017) like sentiment analysis (Tang et al., 2014), irony detection (Reyes et al., 2012), out-of-vocabulary word classification (Ma and Zhang, 2015), or semantic shift detection (Hamilton et al., 2016a; Martinez-Ortiz et al., 2016). One prerequisite when creating high-quality embedding models is a good training corpus. To this end, many approaches use the Google n-gram corpus (Hellrich and Hahn, 2016; Pyysalo et al., 2013; Kim et al., 2014; Martinez-Ortiz et al., 2016; Kulkarni et al., 2016, 2015; Hamilton et al., 2016b). It also is the largest currently available corpus with historic data and exists for several languages. It incorporates over 5 million books from the last centuries split into n-grams (Michel et al., 2015). n-grams are text segments separated into pieces consisting of n words each. The *fragmentation* of a corpus is the size of its n-grams. To illustrate, a corpus of 2-grams is highly fragmented, one of 5-grams is moderately fragmented.

n-gram counts over time can be published even if the underlying full text is subject to copyright protection. Next, this format reduces the data volume very much. So it is important to know how good models built on n-gram corpora are.

While the quality of word embedding models trained on full-text corpora is fairly well known (Lebret and Collobert, 2015; Baroni et al., 2014), an assessment of models built on fragmented corpora is missing (Hill et al., 2014). The resource advertised in this paper is a set of such models, which should help to shed some light on the issue, together with some experiments.

Difficulties. An obvious benefit of making these models available is the huge runtime necessary to build them. However, evaluating them is not straightforward, for various reasons. First, drawing *general* conclusions on the quality of embedding models only based on the performance of specific approaches, i.e., examining the extrinsic suitability of models, is error-prone (Gladkova and Drozd, 2016; Schnabel et al., 2015). Consequently, to come to general conclusions one needs to investigate *general properties* of the embedding models itself, i.e., examine their intrinsic suitability. Properties of this kind are semantic similarity and analogy. For both properties, one can use well-known test sets that serve as comprehensive baselines. Second, there are various parameters which influence how the model looks like. Using n-grams as training corpus gives way to two new parameters, fragmentation, as just discussed, and minimum count, i.e., the minimum occurrence count of an n-gram in order to be considered when building the model. The latter is often used to filter error-prone n-grams from a corpus, e.g., spelling errors. While the effect of the other parameters on the models is known (Lebret and Collobert, 2015; Baroni et al., 2014), the one of these new parameters is not. We have to define meaningful experiments to quantify and compare the effects. Third, the full text, such as the Google Books corpus, is not openly available as reference in many cases. Hence, we need to examine how to compare results from other corpora, where the full text is available, referring e.g., to well-known baselines as the Wikipedia corpus.

Contribution. The resource provided here is a systematic collection of word embedding models trained on n-gram corpora, accessible at our project website[1]. The collection consists of 120 word embedding models trained on the Wikipedia and 1 Billion words data set. Its training has required more than two months of computing time on a modern machine. To our knowledge, it currently is one of the most comprehensive collection of its type. In order to make this resource re-usable and our experiments repeatable, we also provide the n-grammed versions of the Wikipedia and 1-Billion word datasets, which we used for training and the tools to create n-gram corpora from arbitrary text as well.

In addition, we describe some experiments to examine how much model quality changes when the training corpus is not full-text, but n-grams. The experiments quantify how much fragmentation (i.e., values of n) and minimum count reduce the average quality of the corresponding word embedding model, on common word similarity and analogical reasoning test sets.

To show the usefulness and significance of the experiments and to give general recommendations on which n-gram corpus to use as well as creating a baseline for comparison, we conduct the experiments on the full English Wikipedia dump and Chelba et al.'s 1-Billion word dataset. However, we recommend to conduct this examination for any corpus before using it as training resource, particularly if the corpus size differs from the ones of the baseline corpora by much:

1. What is the smallest number n for which an n-gram corpus is good training data for word embedding models?
2. How sensitive is the quality of the models to the fragmentation and the minimum count parameter?
3. What is the actual reason for any quality loss of models trained with high fragmentation or a high minimum count parameter?

Our results for the baseline test sets indicate that minimum count values exceeding a corpus-size-dependent threshold drastically reduce the quality of the models. Fragmentation in turn brings down the quality only if the fragments are very small. Based on this, one can conclude that n-gram corpora such as Google Books are valid training data for word embedding models.

2 Fundamentals and Notation

We first introduce word embedding models. Then we explain how to build word embedding models on n-gram corpora.

2.1 Background on Word Embedding Models

In the following, we review specific distributional models called word embedding models. Experiments have shown that word embedding models are superior to conventional distributional models (Baroni et al., 2014; Mikolov et al., 2013b). In this section, we briefly say how such models work, and which parameters influence their building process.

2.1.1 Building a Word Embedding Model

Word embedding models 'embed' words into a high-dimensional space, representing them as dense vectors of real numbers. Vectors close to

[1]http://dbis.ipd.kit.edu/2568.php

each other according to a distance function, often the cosine distance, represent words that are semantically related. Formally, a word embedding model is a function F which takes a corpus C as input, generates a dictionary D and associates any word in the dictionary $w \in D$ with a d-dimensional vector $v \in \mathbb{R}^d$. The dimension size parameter d sets the dimensionality of the vectors. win is the window size parameter. It determines the context of a word. For example, a window size of 5 means that the context of a specific word is any other word in its sentence, and their distance is at most 5 words. The training is based on word-context pairs $w \times c \in D \times D^{2\times \text{win}}$ extracted from the corpus. The parameter $epoch\_nr$ states how many times the training algorithm passes through the corpus. The model adds only words to the dictionary which are among the $dict\_size$ most frequent words of the corpus. Additionally, there are model specific parameters (θ) which change minor details in their algorithms. Having said this, one can define word embedding models as a function:

$$F(C, d, epoch\_nr, win, dict\_size, \theta) \in \mathbb{R}^d \quad (1)$$

There is related work that studies the impact of these parameters (Baroni et al., 2014; Hill et al., 2014; Elekes et al., 2017). Consequently, for all parameters mentioned except for the window size, we can rely on the results from the literature. This is because, as we outline shortly, win is highly relevant for building embedding models using n-gram corpora. Note that F has been defined to work on full text corpora so far, but not on n-grams. We explain how to adapt F in Section 2.3.

2.1.2 Realizations of Word Embedding Models

In this paper, we work with a well-known embedding model, Mikolov et al.'s Word2Vec model (Mikolov et al., 2013b). Word2Vec models use a neural-network based learning algorithm. They learn by maximizing the probability of predicting the word given the context (Continuous Bag of Words model, CBOW) (Mikolov et al., 2013c,b). Note that the results can be transferred to the skip-gram learning algorithm of Word2Vec as well as to Glove (Pennington et al., 2014; Levy and Goldberg, 2014). This is because very recent work has shown that the distribution of the cosine distances among the word vectors of all such models (after normalization) is highly similar (Elekes et al., 2017).

2.2 Generating Fragmented and Trimmed Corpora

To create fragmented n-gram corpora from raw text, we use a simple method described in the following. With a sliding window of size n passing through the whole raw text, we collect all the n-grams which appear in the corpus and store them in a dictionary, together with their match count. This means that we create datasets similar to the Google Books dataset, but from other raw text such as the Wikipedia dump. For every fragmented corpus, we create different versions of it, by trimming n-grams from the corpora with regard to different minimum count thresholds.

2.3 Building Word Embedding Models on N-grams

Distributional models conventionally are trained on full text corpora, by creating word-context pairs. It does not need to be a coherent text; it is sufficient if the sentences are meaningful. For n-gram corpora, this means the following. When creating the context of a word, we treat each n-gram as if it was a sentence.[2]

The word embedding models which we make available are trained on n-gram training corpora which are in the Google n-gram format. This format comprises 4 values: the n-gram, the match count, the book count and the year. For our purpose, only the first two values are relevant. When building a model with the n-gram versions, we deem every n-gram a sentence and use it as many times as it occurs in the raw text. We explain the impact of the window size and of the match count parameter in the following.

2.3.1 Window Size Parameter

The window size parameter win of the embedding model F depends on the fragmentation. For example, $win = 4$ is not meaningful when we work with 3-grams , because the maximum distance between two words in a 3-gram corpus is 2. The following Examples 1-3 illustrate how exactly the word-context pairs are generated on n-gram corpora depending on the size of the window.

Example 1. Let us look at the context of a specific word in a 5-gram corpus with $win = 4$. Let A B C D E F G H I be a segment of the raw text consisting of 9 words. In the raw text, the context

[2]Of course, if there is a punctuation mark in the n-gram ending a sentence, it splits the n-gram into several sentences.

of word E are words A, B, C, D, F, G, H, I. Now we create the 5-gram version of this segment and identify 5-grams which include word E. These are (A B C D E); (B C D E F); (C D E F G); (D E F G H) and (E F G H I). For word E on the 5-gram corpus, the contexts are words A, B, C, D; ...; words F, G, H, I. We can see that we have not lost any context words. But we also do not have all raw-text context words in one context, only fragmented into several ones.

Example 2. As extreme case, we consider a window size which is bigger than the size of the n-grams. In this setting, we naturally lose a lot of information. This is because distant words will not be in any n-gram at the same time. For example, look at the same text segment as in Example 1 with $win = 4$, but with a 3-gram variant of it. The raw-text context is the same as before for word E, but the fragmented contexts are words C, D; words D, F and words F, G.

Example 3. Another extreme case is when win is less than or equal to $\lfloor \frac{n-1}{2} \rfloor$. In this case at least one n-gram context will be the same as the full text context. This means that no information is lost. However, there also are fragmented contexts in the n-gram variant which can influence the training and, hence, model quality.

We point out that bigger window sizes do not necessarily induce higher accuracy on various test sets, as explained in (Levy et al., 2015).

2.3.2 Match Count Parameter

Another parameter to consider when building the models on the n-gram corpora is the match count of the n-grams. The intuition behind including only higher match-count n-grams in the training data is that they may be more valid segments of the raw text, as they appear several times in the same order. However, we naturally lose information by pruning low match-count n-grams. When we create the n-gram versions of our raw text corpora, we create different versions, by pruning the n-grams which do not appear in the corpus at least 2, 5, 10 times, respectively. These numbers are much smaller than the minimum count of 40 in the Google dataset. This is because the raw texts we use are much smaller than the Google Books dataset as well.

3 Experimental Setup

The experiments allow to to give general recommendations on which n-gram corpus to use to train word embedding models. In order to do this, we justify the selection of our text corpora and of the parameter values used. Finally, we explain the actual baseline test sets and say why this choice will yield general insights.

3.1 Raw Text Corpus Selection

In this study we work with two raw-text corpora. One is Chelba et al.'s 1-Billion word dataset (Chelba et al., 2013), the other one is a recent (November 1, 2016) Wikipedia dump, with the articles shuffled and sampled to contain approximately 1 billion words. Both corpora are good benchmark datasets for language modeling, with their huge size, large vocabulary and topical diversity (Chelba et al., 2013; Zesch and Gurevych, 2007). For both corpora we create their respective n-gram versions in the Google Books format, with n=2,3,5,8, cf. Section 2.3.

3.2 Baseline Test Sets

We see two properties of embedding models that makes them a worthwhile resource. First, word similarities, second, analogical reasoning. To illustrate, an analogical reasoning task is as follows: 'A is to B as C is to D', and the model has to guess Word D knowing the other three words. We have selected widely used word similarity and analogical reasoning baseline test sets for our evaluation.

3.2.1 Word Similarity

We use six test sets to evaluate word similarity. They contain word pairs with similarity scores, assigned by human annotators. The test sets are Finkelstein et al.'s WordSim353 (Finkelstein et al., 2001) partitioned into two test sets, WordSim Similarity and WordSim Relatedness (Zesch et al., 2008; Bruni et al., 2012); Bruni et al.'s MEN test set (Bruni et al., 2012); Hill et al.'s SimLex-999 test set (Hill et al., 2015); Rubenstein and Goodenough's RG-65 test set (Rubenstein and Goodenough, 1965); Radinsky et al.'s Mechanical Turk test set (Radinsky et al., 2011) and Luong et al.'s Rare Words test set (Luong et al., 2013). We evaluate the models with the conventional baseline evaluation method (Levy et al., 2015) (Baroni et al., 2014): We rank the word pairs of an evaluation test set by their similarity scores, based on cosine distance of the word vectors. Then we measure Spearman's correlation between this ranking and the one based on human annotators. This number is the score of the model on the test set.

3.2.2 Analogical Reasoning

We use two analogical reasoning test sets. The first one is MSR's analogy test set (Mikolov et al., 2013c), which contains 8000 syntactic analogy questions, such as "big is to biggest as good is to best". The other test set is Google's analogy test set (Mikolov et al., 2013b), which contains 19544 questions The models answer the questions with the following formula:

$$\text{argmax}_{d \in D \setminus \{a,b,c\}} \cos(d, b - a + c) \qquad (2)$$

Here $a, b, c, d \in D$ are the vectors of the corresponding word. The score of a model is the percentage of questions for which the result of the formula is the correct answer (d).

4 Experiment Questions

In order to make our experimental results more intuitive we attempt to explicitly answer the three following questions.

Question 1. What is the smallest number n for which an n-gram corpus is good for the training of embedding models?

Rationale behind Question 1. The size of any n-gram corpus highly increases with large n. Hence, it is important to know the smallest value that is expected to still yield good results.

Question 2. How does the minimum count parameter affect the quality of the models? How does this result compare to the effect caused by the fragmentation?

Rationale behind Question 2. Having answered the first question, we will be able to quantify the effect of the fragmentation. However, it is necessary to study the effect of the second parameter as well, in order to quantify the applicability of n-grams for embedding comprehensively. In other words, we want to compare the effects of both parameters; we will be able to give recommendations for both parameters.

Question 3. How does the quality loss of models trained on fragmented corpora of size n or with high minimum count parameter manifest itself in the embedding models?

Rationale behind Question 3. By answering Questions 1 and 2, we are able to quantify the effect of both parameters. We hypothesize that the parameters affect the quality of the models differently, and that we are able to observe this in the word vectors themselves. The rationale behind our hypothesis is that large minimum count values might eliminate various meaningful words from the vector space. Fragmentation in isolation however does not have the effect that a word is lost. Hence, a quality loss must manifest itself differently, which might be observable.

5 Experiment Results

We now turn to the questions just asked. We dedicate a subsection to each question. In the first two sections, we give an overview of the results using the Wikipedia corpus. The results for the 1-Billion word corpus are almost identical. For brevity, we do not show all results for this corpus.

5.1 Answering Question 1: Minimal Meaningful N-gram Size

We quantify the influence of the training-corpus fragmentation on the quality of the word-embedding models. We do not use a minimum count parameter in this section.

5.1.1 Results for the Wikipedia Corpus

Figure 1 shows the result for the models trained on the Wikipedia corpus.[3] The interpretation of the plots is as follows: We evaluate a specific model on every test set introduced in Section 3.2. We calculate the average scores for this model for both the similarity and analogy test sets. We do this for every trained model. We group the results by the window-size parameter of the models and plot the average values. So every plot shows the calculated average scores of such models which only differ in the fragmentation of their training corpus. As explained in Section 2.3.1, it is meaningless to train models on n-grammed corpora with window-size parameter $win < n$.

Figure 1 reveals that fragmentation does influence the quality of the models significantly. For the similarity test sets, fragmentation reduces the quality of the models for any value of the window-size parameter win almost linearly.

For the analogy test sets, the results are not as straightforward. Generally, the same observation holds as for the similarity test sets, namely that fragmentation reduces the quality of the models, however there are a few exceptions. It is true that the best models are trained on the 5 and 8-gram variants and the full text corpora for any window size. For models with smaller win however, the

[3] Note that we use different scales for the first and the second two subplots.

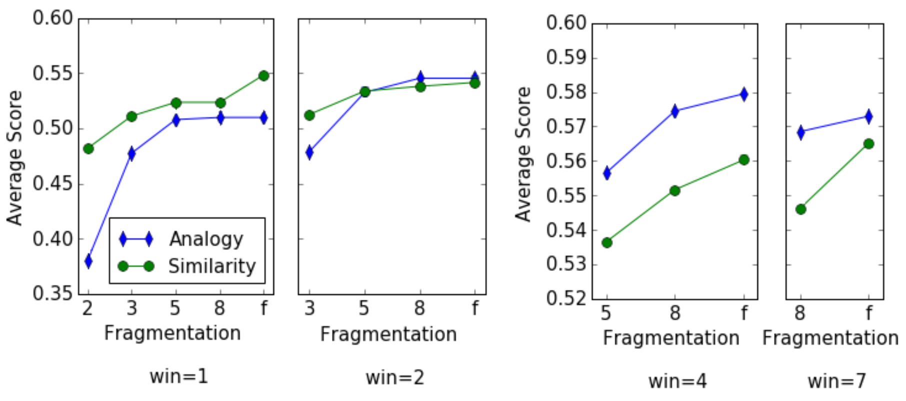

Figure 1: The average score of models trained on differently fragmented Wikipedia corpora on the analogy and similarity tasks

results do not always get better when the corpus is less fragmented. For example, the very best model for the MSR test set is trained on 8-grams, not the full text.

5.1.2 Generalization of Results

To generalize the results, we measure the overall average quality of the models trained on the differently fragmented corpora. Then we compare the results to ones computed on the full text. To this end, we calculate the averages of the previous results, grouped by training corpus fragmentation. To make the resulting numbers more intuitive, we do this in relative terms, compared to the results with the full text corpus. For example, on analogical reasoning tasks, the models trained on 3-grams are 7.5% worse than the models trained on full text with the same window size. Table 1 shows the results. The total column is the average of the similarity and analogy columns

Table 1: Average quality loss due to fragmentation compared to the full text on the Wikipedia corpus.

Wikipedia	total	similarity	analogy
2-gram	-20.2%	-14.4%	-26.0%
3-gram	-6.9%	-6.4%	-7.5%
5-gram	-2.8%	-3.6%	-1.9%
8-gram	-2.5%	-3.4%	-1.6%

5.1.3 Result Interpretation.

We conclude that embedding models built on fragmented corpora are worse than models based on full text, but the difference is not much. Embedding models trained on 2-gram corpora are 20.2% worse overall than models trained on full text, a significant drop. However, the 3-gram version is only 6.9% worse. The 5-gram version and the 8-gram version are almost tha same, they are only 2.8% and 2.5% worse, respectively. This answers Question 1, i.e., a 5-gram corpus is the most fragmented corpus which is almost as good as models trained on full text, and the 3-gram version is also not much worse. A general takeaway for other researchers when training embedding models on n-grams is that using at least 3-grams for training leads to good models and using at least 5-grams leads to almost identical models as training on full text.

5.1.4 Insight Confirmation Using a Different Text Corpus

So far, our results only rely on one corpus. To generalize, we verify our insights using another large corpus. We only list aggregated results to save space. We have calculated the numbers for all models trained on the 1-Billion word dataset and its fragmented versions. Table 2 shows that the numbers are generally very similar to the previous ones with Wikipedia. So the decline in quality most likely does not depend on the underlying text corpus, but on the fragmentation.

5.2 Answering Question 2: Effect of the Minimum Count Parameter

In the following, we aim at quantifying the influence of the minimum count parameter and investigate whether there is an interaction with the fragmentation. Our procedure is the same as in the prior section. First, we aggregate the raw results obtained from the Wikipedia corpus and *all* models built on it. Then we draw first conclusions and finally verify the insights using the second corpus.

Table 2: Average quality loss due to fragmentation compared to the full text on the 1-Billion word corpus.

1-Billion	total	similarity	analogy
2-gram	-19.3%	-15.0%	-23.6%
3-gram	-5.4%	-6.5%	-4.3%
5-gram	-1.8%	-2.9%	-0.8%
8-gram	-1.3%	-2.2%	-0.4%

5.2.1 Results for the Wikipedia Corpus

Figure 2 shows the average quality of models trained on the same n-gram corpus with different minimum count parameter. The results indicate that an increase of this parameter usually leads to significantly worse models. However, there is one exception, where the minimum count parameter is 2 and the corpus is the 2-grammed version of Wikipedia. For this case the models actually gets slightly better on the analogy task using the minimum count threshold. The reason is that we do not lose too many 2-grams with the thresholding in this case, and those which we do lose may bias the model on the analogy tasks. However, on the similarity tasks the models get slightly worse, which means we lose meaningful training data as well. The reduction in quality is even more severe with n-gram corpora with a big value of n, such as 5 or 8-grams. This is because these corpora have fewer high match count n-grams than the more fragmented 2 or 3-gram corpora. Summing up, for corpora of a size such as the Wikipedia dump, any threshold for the minimum count of the n-grams significantly reduces the quality of the embedding models. One exception is when the corpus is highly fragmented with the smallest minimum count parameter.

5.2.2 Generalization of the Results

We generalize the aforementioned observations by computing the average quality loss for the models, just as in Section 5.1.2. The numbers in Table 3 are for the Wikipedia dataset. With the smallest minimum count threshold already, the models get significantly worse. For the 1-Billion word dataset, the results again differ only slightly. This again confirms our hypothesis that the quality differences are not corpus-dependent.

5.2.3 Implications for the Google N-gram Corpus

So far, the question how to transfer the results from the Wikipedia and the 1-Billion corpus to the

Table 3: Average quality loss caused by the minimum count parameter parameter on Wikipedia.

Min. count	total	similarity	analogy
2	-23.6%	-19.2%	-28.0%
5	-56.5%	-47.9%	-65.1%
10	-72.3%	-64.0%	-78.6%

Google n-gram corpus remains open. We aim to answer whether we lose any meaningful information in the Google 5-gram corpus, because of the, at first sight, large threshold value of 40. We assume that even for such comprehensive text corpora, including the Wikipedia or the 1-Billion corpus, all the extracted n-grams are contained in the Google n-gram data as well, despite its large threshold value. We verify this hypothesis by comparing the number of existing 5-grams in the Google n-gram corpus (1.4 Billion) with those in the full Wikipedia (1.25 Billion). A systematic analysis of the data reveals that more than 99% of the 5-grams included in the Wikipedia corpus is included in the Google corpus as well. The ones which are not usually are typos or contain words which have not been present in the language until 2008 (the last year in the Google dataset). This holds for all n-gram corpora. This means that we do not lose any relevant information if we train our models on the Google n-gram dataset, despite its high minimum count threshold value. So it is a suitable training corpus for word-embedding models.

5.2.4 Result Interpretation

To conclude, we can now answer Question 2. We see that the minimum count parameter reduces model quality. This conclusion depends on the size of the corpus . For smaller corpora, the effect will be even more pronounced. For such sizes of the training data we do not recommend to use any minimum count threshold when training embedding models. In combination with the results from Section 5.1, we conclude that the Google Books dataset is valid training data for embedding models. In general, one can expect good results using the 5-grams as training data, but anything above 2-grams could be used.

5.3 Answering Question 3: the Reason for the Quality Loss

Regarding Question 3, we start with an explanation why the increase of the minimum count pa-

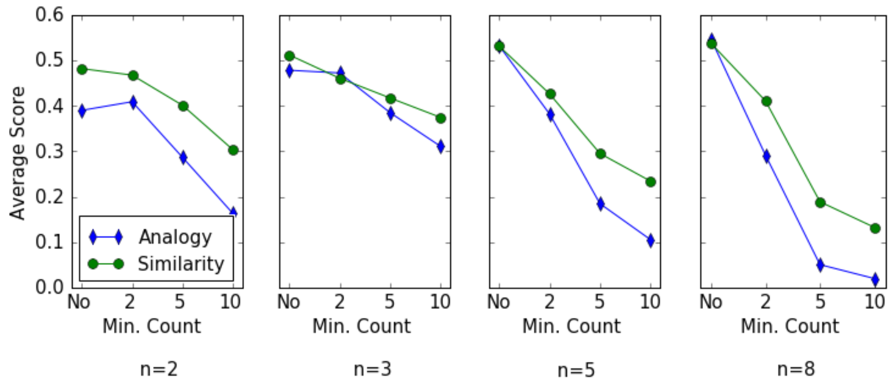

Figure 2: The average scores of models trained with different minimum count parameter on differently fragmented Wikipedia corpora on the Analogy and Similarity tasks

rameter decreases the quality of the embedding models. We have observed that in almost every case this has been a consequence of certain infrequent words of the evaluation test sets not occurring in sufficiently many n-grams. When analyzing the task specific results, we have seen that result quality on the rare words test set drops instantly even with the smallest threshold value. For other test sets on the other hand (WordSim353, RG-65 for instance) which include highly frequent words almost exclusively, results are not much worse. In summary, the reduced model quality generally is a consequence of the less frequent words not being trained sufficiently or even not at all. Therefore, they do not appear in the dictionaries of the model.

Table 4: Average movement in cosine distance of the word vectors with one extra iteration.

Corpus	2-gram	3-gram	5-gram	8-gram	Full text
Avg. movement	0.018	0.016	0.011	0.010	0.006

The fragmentation of the corpora causes a quality loss for a different reason. Every word of the evaluation test sets is included in the dictionary of every model, but fragmentation causes a mix-up of the word vectors, cf. Section 2.2. As explained in Example 1, each word is trained several times when fragmented corpora are used, and most of the time the context of the word, as considered by the algorithm during training, is not the full context. To quantify this effect, we have measured the average movement of a word vector when we iterate through the training data one extra time, after training the models. See Table 4; the numbers are average cosine distances. The lower the corpus quality is, the more the vectors move in the additional iteration. The results seem to confirm our intuition that, with bad corpora, vectors move in suboptimal directions to a higher extent, ultimately resulting in worse models.

6 Conclusions

In this paper we present a resource and corresponding experiments which allow to answer which differences in quality one can expect when training word embedding models on fragmented corpora, such as the Google n-gram corpus, compared to full-text. The resource contains all models, corpora and scripts we have used. The resource contains one of the largest collection of systematically pre-trained embedding models currently openly available. It also contains the fragmented versions of both corpora used in this paper and our scripts used to conduct the experiments. We present experiments to give recommendations on which n-gram versions to use for word embedding model training. An in-depth evaluation using our presented comprehensive resource confirms that one generally can expect good quality for n-grams with $n \geq 3$. In addition, we show that the minimum count parameter is highly corpus size dependent and should not be used for corpora with size similar to or smaller than the Wikipedia dump. Finally, our results show that the fragmentation (i.e., small values for n) and the minimum count parameter introduce different kinds of error.

In summary, our results indicate that one can train high-quality embedding models with n-grams if some (mild) prerequisites hold. This is particularly true for the Google n-gram corpus, which is a good corpus to this end.

References

M. Baroni et al. 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *ACL*. ACL.

E. Bruni et al. 2012. Distributional semantics in technicolor. In *Proc. Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 136–145. ACL.

C. Chelba et al. 2013. One billion word benchmark for measuring progress in statistical language modeling. *CoRR*, abs/1312.3005.

A. Elekes, M. Schäler, and K. Böhm. 2017. On the various semantics of similarity in word embedding models. In *JCDL*. IEEE.

L. Finkelstein et al. 2001. Placing search in context: The concept revisited. In *Proc. Int'l Conf. on World Wide Web (WWW)*, pages 406–414. ACM.

A. Gladkova and A. Drozd. 2016. Intrinsic evaluations of word embeddings: What can we do better? In *RepEval*. ACL.

W. Hamilton et al. 2016a. Cultural shift or linguistic drift? comparing two computational measures of semantic change. In *EMNLP*. ACL.

William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2016b. Diachronic word embeddings reveal statistical laws of semantic change. *arXiv preprint arXiv:1605.09096*.

J. Hellrich and U. Hahn. 2016. An assessment of experimental protocols for tracing changes in word semantics relative to accuracy and reliability. In *LaTeCH*.

F. Hill, R. Reichart, and A. Korhonen. 2015. Simlex-999: Evaluating semantic models with genuine similarity estimation. *Computer Linguistics*, 41(4):665–695.

F. Hill et al. 2014. Not all neural embeddings are born equal. *CoRR*, abs/1410.0718.

S. Jansen. 2017. Word and phrase translation with word2vec. *CoRR*, abs/1705.03127.

Y. Kim et al. 2014. Temporal analysis of language through neural language models. *CoRR*, abs/1405.3515.

Vivek Kulkarni, Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2015. Statistically significant detection of linguistic change. In *Proceedings of the 24th International Conference on World Wide Web*, pages 625–635. International World Wide Web Conferences Steering Committee.

Vivek Kulkarni, Bryan Perozzi, and Steven Skiena. 2016. Freshman or fresher? quantifying the geographic variation of language in online social media. In *ICWSM*, pages 615–618.

R. Lebret and R. Collobert. 2015. *Rehabilitation of Count-Based Models for Word Vector Representations*. Springer.

O. Levy and Y. Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *NIPS*. MIT Press.

O. Levy, Y. Goldberg, and I. Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *TACL*, 3.

T. Luong, R. Socher, and C. Manning. 2013. Better word representations with recursive neural networks for morphology. In *CoNLL*. ACL.

L. Ma and Y. Zhang. 2015. Using word2vec to process big text data. In *Proc. Int'l. Conf. on Big Data (Big Data)*, pages 2895–2897. IEEE.

C. Martinez-Ortiz et al. 2016. Design and implementation of shico: Visualising shifting concepts over time. In *HistoInformatics, DH*.

J.-B. Michel et al. 2015. Quantitative analysis of culture using millions of digitized books. *Science*, 331(6014):176–182.

T. Mikolov, Q. Le, and I. Sutskever. 2013a. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.

T. Mikolov et al. 2013b. Distributed representations of words and phrases and their compositionality. In *NIPS*. Curran Associates Inc.

T. Mikolov et al. 2013c. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

B. Mitra and N. Craswell. 2017. Neural text embeddings for information retrieval. In *WSDM*, pages 813–814. ACM.

J. Pennington et al. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543. ACL.

S. Pyysalo et al. 2013. Distributional semantics resources for biomedical text processing. In *LBM*.

K. Radinsky et al. 2011. A word at a time: Computing word relatedness using temporal semantic analysis. In *WWW*, pages 337–346. ACM.

A. Reyes et al. 2012. From humor recognition to irony detection: The figurative language of social media. *DKE*.

Herbert Rubenstein and John B Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

T. Schnabel et al. 2015. Evaluation methods for unsupervised word embeddings. In *EMNLP*.

Duyu Tang et al. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *ACL*, volume 1, pages 1555–1565.

T. Zesch and I. Gurevych. 2007. Analysis of the wikipedia category graph for nlp applications. In *NAACL-HLT*, pages 1–8. ACL.

T. Zesch et al. 2008. Using wiktionary for computing semantic relatedness. In *AAAI*, pages 861–866. AAAI Press.