D-GloVe: A Feasible Least Squares Model for Estimating Word Embedding Densities*

Shoaib Jameel and Steven Schockaert
School of Computer Science and Informatics,
Cardiff University.
{JameelS1, S.Schockaert}@cardiff.ac.uk

Abstract

We propose a new word embedding model, inspired by GloVe, which is formulated as a feasible least squares optimization problem. In contrast to existing models, we explicitly represent the uncertainty about the exact definition of each word vector. To this end, we estimate the error that results from using noisy co-occurrence counts in the formulation of the model, and we model the imprecision that results from including uninformative context words. Our experimental results demonstrate that this model compares favourably with existing word embedding models.

1 Introduction

Several vector space models for word meaning have already been proposed (Lund and Burgess, 1996; Landauer and Dumais, 1997; Turney and Pantel, 2010; Mikolov et al., 2013; Pennington et al., 2014). While there are considerable differences in how these vector space models are learned, most approaches represent words as vectors. However, a few authors have proposed models that represent words as regions or densities in a vector space (Erk, 2009; Vilnis and McCallum, 2015), motivated by the view that region or density based representations are better suited to model the diversity of word meaning, and can thus capture e.g. hyponymy in a natural way. In this paper, we also use densities in a low-dimensional vector space to represent word meaning. In contrast to previous work, however, we use densities for modelling our lack of knowledge about the precise meaning of a word. This allows us to use a more cautious representation for rare words, and leads to better confidence estimates in downstream tasks. Note that this use of densities is indeed fundamentally different from its use in previous work. For example, increasing the corpus size in our case will lead to more precise estimates (i.e. distributions with lower variance) while the models from (Erk, 2009; Vilnis and McCallum, 2015) may arrive at distributions with higher variance, reflecting the broader set of context windows that may be found.

Our approach is based on the GloVe model for word embedding (Pennington et al., 2014). In particular, we also associate two vectors with each word i: the vector w_i, which intuitively represents the meaning of word i, and the vector $\tilde{w}_j$, which intuitively represents how the occurrence of i in the context of another word j affects the meaning of that word. Moreover, we also use a least squares formulation to constrain these vectors such that $w_i \cdot \tilde{w}_j$ reflects the co-occurrence statistics of words i and j. In contrast to GloVe, however, we explicitly model two factors that contribute to the residual error of this model: (i) the fact that corpus statistics for rare terms are not reliable and (ii) the fact that not all words are equally informative. This has two key advantages. First, it allows us to formulate the underlying optimization problem as a feasible generalized least squares problem. As we show in our experiments, this leads to word embeddings (as vectors) that substantially outperform those obtained from the GloVe model, and other baselines, in standard word similarity and analogy tasks. Second, it allows us to explicitly represent our uncertainty about the precise definitions of the word vectors. Rather than using w_i for modelling the meaning of word i, we then consider a density which is defined by the residual error model.

Specifically, the residual error model allows us to naturally associate a univariate density with each context vector $\tilde{w}_j$, given a target word i. A natural geometric interpretation can be obtained by fixing the

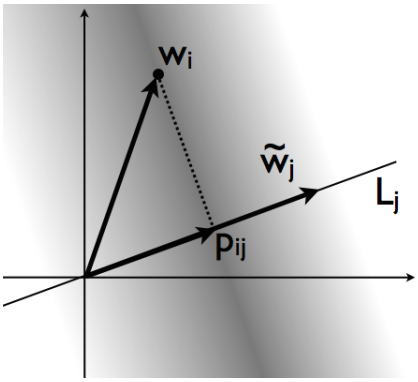

Figure 1: Density modelling the possible values of w_i, induced by a single context word $\tilde{w_j}$.

context vectors. The density associated with a given context word can then be viewed as a soft constraint on the possible values of the vector w_i, as illustrated in Figure 1. The variance of this density depends on the size of the corpus (larger corpora lead to more precise estimates) and on the informativeness of the context word, where densities associated with uninformative context words should have a high variance, reflecting the fact that they should not have a strong impact on the word embedding.

The remainder of this paper is structured as follows. In the next section, we give an overview of related work. Subsequently, in Section 3 we introduce our probabilistic model for word embedding and discuss its relationship with the GloVe model. Finally, we present our experimental results and conclusions.

2 Related work

Word embedding models construct vector space models of word meaning by relying on the distributional hypothesis, which states that similar words tend to appear in similar linguistic contexts (Harris, 1954). One class of methods relies on constructing a term-document (Landauer and Dumais, 1997) or, more commonly, a term-term (Lund and Burgess, 1996) co-occurrence matrix. Intuitively, we can think of the row vectors in these matrices as representing the contexts in which a given word occurs. Given the sparse and high-dimensional nature of these vectors, most approaches use some form of dimensionality reduction based on matrix factorization, such as singular value decomposition (SVD). An important factor in the performance of such methods is how co-occurrences are weighted, with Positive Pointwise Mutual Information (PPMI) generally considered to be a suitable choice (Bullinaria and Levy, 2007).

In the last few years, a number of neural network inspired methods have been proposed that formulate the problem of learning word embeddings as an optimization problem. The well-known skip-gram (SG) method (Mikolov et al., 2013), for example, aims to construct vectors, such that the log-probability that word c appears in the context of word w is proportional to $w \cdot c$. The related Continuous Bag of Words (CBOW) model uses a similar idea, but instead focuses on predicting the probability of a given target word, given its context. The GloVe model (Pennington et al., 2014), which our approach is based on, learns two word vectors w_i and $\tilde{w_j}$ and a bias b_i for each word i, using the following least squares regression formulation:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} f(x_{ij})(w_i \cdot \tilde{w}_j + b_i + \tilde{b}_j - \log x_{ij})^2$$

where x_{ij} is the number of times words w_i and $\tilde{w}_j$ co-occur, $\tilde{b}_j$ is the bias for the context word j, and n is the number of different words in the considered corpus. The function f weights the terms of the model to limit the effect of small co-occurrence counts, as these are deemed to be noisy. It is defined as $f(x_{ij}) = \left(\frac{x_{ij}}{x_{max}}\right)^{\alpha}$ if $x_{ij} < x_{\max}$ and $f(x_{ij}) = 1$ otherwise. The purpose of x_{max} is to prevent common words from dominating the objective function too much.

An interesting property of the representations learned by SG, CBOW and GloVe is that they capture similarity as well as analogies and related linear relationships. As a result, both word similarity and word analogy tasks are now commonly used to evaluate the quality of word embeddings. Finally, note

that while methods such as SG might seem like a radical departure from matrix factorization based methods, it was shown in (Levy and Goldberg, 2014) that SG implicitly finds a factorization of a shifted-PMI weighted term-term co-occurrence matrix. It has been observed that, compared to the factorization model underlying SG, SVD remains a useful choice for modeling word similarity, but it is less suited for discovering analogies (Levy and Goldberg, 2014).

The standard word embedding models have recently been improved in various ways. For example, some authors have proposed so-called multi-prototype representations, where the idea is to deal with ambiguity by learning several vectors for each word (Reisinger and Mooney, 2010; Huang et al., 2012; Liu et al., 2015; Neelakantan et al., 2015), intuitively by clustering the contexts in which the word appears, as a proxy for word senses, and learning one vector for each context. Other authors have shown how word embeddings can be improved by taking into account existing structured knowledge, e.g. from lexical resources such as WordNet or from knowledge graphs such as Freebase (Yu and Dredze, 2014; Xu et al., 2014; Faruqui et al., 2015).

While most word embedding models represent words as vectors, a few authors have explored the usefulness of region and density based representations. For example, in (Erk, 2009), two models are proposed to induce regions from context vectors. Among others, it is shown that these regions can be used to encode hyponym relations. In (Vilnis and McCallum, 2015), a model is proposed which represents words as Gaussian densities, and the usefulness of this model for discovering word entailment is demonstrated. As already mentioned in the introduction, while our model also represents words as densities, our densities model the uncertainty about the true location of a word vector, rather than modelling the diversity of the underlying concept. As a result, for instance, Kullback-Leibler divergence is meaningful for modelling word similarity in the approach from (Vilnis and McCallum, 2015), but would not be appropriate in our model. To the best of our knowledge, the model presented in this paper is the first that explicitly models the uncertainty associated with the word vectors. Note that while several probabilistic models have been proposed for word embedding (Maas and Ng, 2010; Li et al., 2015), these works model the probability that a document has been generated, rather than the probability that a given vector is the correct representation of a given word.

3 Our model

Similar to the GloVe model, we propose to learn word embeddings by solving the following weighted least squares problem:

$$\sum_{i=1}^{n} \sum_{j \in J_i} \frac{1}{\sigma_{ij}^2} (w_i \cdot \tilde{w}_j + \tilde{b}_j - s_{ij})^2 \tag{1}$$

where n is the number of words in the vocabulary, $J_i \subseteq \{1, ..., n\}$, and $\tilde{b}_j$ and s_{ij} are constants. In particular, the GloVe model can be recovered by choosing $J_i = \{j \,|\, x_{ij} > 0\}$, $\sigma_{ij}^2 = \frac{1}{f(x_{ij})}$ and $s_{ij} = \log x_{ij} - b_i$. However, as we explain below, these choices are sub-optimal. First, in Section 3.1 we propose to use a Dirichlet-Multinomial language modeling approach and choose s_{ij} as the expectation of $\log P(j|i)$ in this model. Section 3.2 then explains how suitable estimates for σ_{ij}^2 can be obtained. The importance of these estimates is twofold: they should improve the quality of the word vectors that we obtain by solving (1) and they will enable us to precisely model our uncertainty about the exact location of each word vector. This latter point is discussed in more detail in Section 3.3, which explains how we can evaluate the likelihood that a vector is the correct representation of a given word.

3.1 Dealing with imperfect corpus statistics

Let us write $s_{ij}^{glove} = \log x_{ij} - b_i$. The idea behind the derivation of the GloVe model in (Pennington et al., 2014) is that s_{ij}^{glove} is an estimation of $\log P(j|i)$, with $P(j|i)$ the probability of seeing word j in the context of word i. Rather than fixing $b_i = \log \sum_l x_{il}$, in line with this view, it is assumed that $\log \sum_l x_{il}$ is absorbed in the bias term b_i. One of the main advantages of this choice is that it makes the model

symmetric w.r.t. the role of the target vectors w_i and context vectors $\tilde{w_j}$; e.g. in some experiments it was observed that using the average of w_i and $\tilde{w_j}$ can lead to a small increase in performance.

In our model, s_{ij} will be chosen as an estimation of $\log P(j|i)$. This will enable a more elegant modeling of the residual errors, and offer a more principled way of dealing with sparse frequency counts. It also leads to a clearer geometric interpretation. In particular, let us write p_{ij} for the orthogonal projection of w_i on the line $L_j = \{p \,|\, p = \lambda \cdot \tilde{w}_j, \lambda \in \mathbb{R}\}$ (see Figure 1), then $w_i \cdot \tilde{w}_j = \|\tilde{w}_j\| \cdot \|p_{ij}\|$. This allows us to write the residual error as $e_{ij} = \|\tilde{w}_j\| \cdot \|p_{ij}\| - s_{ij} + \tilde{b}_j$. We can think of $\|p_{ij}\|$ as the coordinate of word i in a one-dimensional word embedding, which is constrained by the model to correspond to a linear function of s_{ij}. The relation between this one-dimensional embedding and the full embedding is determined by $\|\tilde{w}_j\|$ and $\tilde{b}_j$, which only depend on the context word j. Another way to look at this geometric interpretation is that each context word j acts as a soft constraint on the possible choices of w_i, which is illustrated by the shaded area in Figure 1.

Clearly $\frac{x_{ij}}{\sum_l x_{il}}$ only gives us a reliable estimate of $P(j|i)$ if the number of occurrences of i is sufficiently large. This problem is well known and can be alleviated by various smoothing techniques (Zhai and Lafferty, 2004). In this paper, we will adopt Bayesian smoothing. In addition to smoothing the frequency counts, this will give us a way to estimate variance. In particular, we assume that for each target word i there is a multinomial distribution from which all words that appear in the context of i are drawn. A standard approach is to assume that the parameters of that multinomial distribution are drawn from a Dirichlet distribution. Specifically, let x_{ij} be the number of times word j appears in the context of word i in the considered corpus, as before, and let $x_i = \sum_l x_{il}$. Note that x_i is the total number of tokens that occur in the context of i. The probability that among these there are y_1 occurrences of word 1, y_2 occurrences of word 2, etc. is given by

$$P(\mathbf{y} \,|\, \alpha) = \frac{x_i B(\sum_j \alpha_j, x_i)}{\prod_{y_j > 0} y_j B(\alpha_j, y_j)}$$

where $\mathbf{y} = (y_1, ..., y_n)$, $\alpha = (\alpha_1 + x_{i1}, ..., \alpha_n + x_{in})$ and B is the Beta function. In the experiments, we will use the overall corpus statistics to set the parameters of the Dirichlet prior, i.e. we will choose $\alpha_i = \lambda \cdot \frac{n_i}{\sum_j n_j}$, where n_i is the total number of occurrences of word i in the corpus and $\lambda > 0$ is a parameter that will be chosen based on tuning data.

Using this Dirichlet-Multinomial model, we can set s_{ij} as the expectation of $\log \frac{Y_{ij}}{x_i}$, where the random variable Y_{ij} represents the number of occurrences of word j in the context of word i. We estimate this expectation using a Taylor expansion:

$$s_{ij} = E[\log Y_{ij}] - \log x_i \approx \log E[Y_{ij}] - \frac{\mathit{Var}[Y_{ij}]}{(2 \cdot E[Y_{ij}]^2)} - \log x_i$$

where

$$E[Y_{ij}] = \frac{x_i \alpha_j^*}{\sum_l \alpha_l^*} \qquad \mathit{Var}[Y_{ij}] = \Big(\frac{x_i \alpha_j^*}{\sum_l \alpha_l^*}\Big)\Big(1 - \frac{\alpha_j^*}{\sum_l \alpha_l^*}\Big)\Big(\frac{x_i + \sum_l \alpha_l^*}{1 + \sum_l \alpha_l^*}\Big)$$

with $\alpha_j^* = \alpha_j + x_{ij}$. This choice of s_{ij} has two advantages over s_{ij}^{glove}. First, by smoothing the raw frequency counts, we obtain more reliable estimates for rare terms. Second, context words j for which $x_{ij} = 0$ are completely ignored in the GloVe model. From the point of view of the proposed geometric interpretation, this means that valuable information is ignored. In particular, if we want $\|p_{ij}\|$ to reflect how strongly context word j is related to word i, we should require that it is small when $x_{ij} = 0$. On the other hand, evaluating (1) for every pair (i, j) is not feasible as it would make the complexity of the model quadratic. Therefore, we let J_i contain all indices j for which $x_{ij} > 0$, as well as a random sample of indices for which $x_{ij} = 0$. In our experiments, we choose the sample size such that the number of indices for which $x_{ij} > 0$ is equal to the number of indices for which $x_{ij} = 0$.

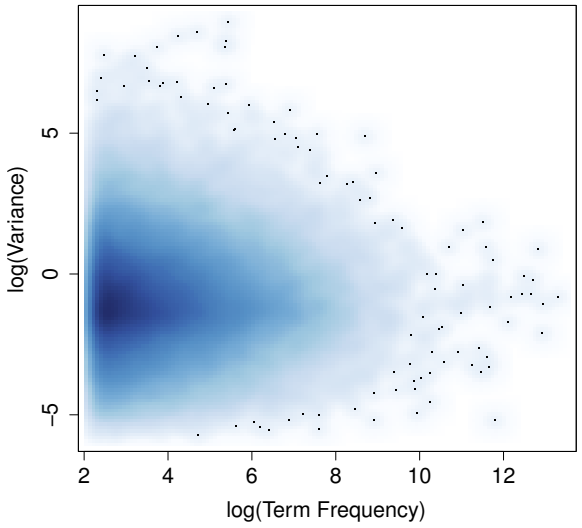

Figure 2: Scatter plot comparing term frequency with $\mathit{Var}[e_j^{\mathit{info}}]$.

3.2 Estimating the variance of residual errors

The choice of $f(x_{ij})$ as the weight for the term corresponding to target word i and context word j reflects the implicit assumption that $\frac{1}{f(x_{ij})}$ is a reasonable estimate of the variance σ_{ij}^2 of the residual error $e_{ij}^{\mathit{glove}} = w_i \cdot \tilde{w}_j + b_i + \tilde{b}_j - \log x_{ij}$. As we will see, this assumption is rather questionable.

A standard technique for selecting the weights in weighted least squares problems, called feasible generalized least squares, consists in estimating the variance of the residual errors in an initial solution (e.g. obtained using standard least squares). This allows us to reformulate the objective function by deriving appropriate weights from the estimated variances σ_{ij}^2. Solving the resulting optimization problem in turn allows us to obtain better estimates of the variances. This process is repeated for a fixed number of times, or until the estimated variances converge.

In our model, we will follow this strategy to estimate the variances σ_{ij}^2 from the observed residual errors e_{ij}. This requires us to make assumptions about which factors affect these variances, as we can clearly not estimate σ_{ij}^2 from e_{ij} alone. We will assume that e_{ij} is the sum of two independent errors, viz. $e_{ij} = e_{ij}^{\mathit{count}} + e_j^{\mathit{info}}$. Intuitively e_{ij}^{count} is the error that results from using unreliable co-occurrence statistics and e_j^{info} is the error that results when the target word j is uninformative. Again using a Taylor expansion, we can estimate $\mathit{Var}[e_{ij}^{\mathit{count}}]$ as follows:

$$\mathit{Var}[e_{ij}^{\mathit{count}}] = \mathit{Var}[\log Y_{ij}] \approx \frac{\mathit{Var}[Y_{ij}]}{E[Y_{ij}]^2}$$

where $E[Y_{ij}]$ and $\mathit{Var}[Y_{ij}]$ are evaluated as before. Furthermore, we can estimate $\mathit{Var}[e_j^{\mathit{info}}]$ from the observed residual errors, as follows:

$$\frac{\sum\{e_{ij}^2 : j \in J_i\} - \sum\{\mathit{Var}[e_{ij}^{\mathit{count}}] : j \in J_i\}}{|\{e_{ij}^2 : j \in J_i\}|} \tag{2}$$

This allows us to estimate σ_{ij}^2 as $\mathit{Var}[e_{ij}^{\mathit{count}}] + \mathit{Var}[e_j^{\mathit{info}}]$.

Figure 2 shows the relationship between $\mathit{Var}[e_j^{\mathit{info}}]$ and the number of occurrences of the context word j in the text collection (for a subset of Wikipedia[1]). As can be seen from the figure, the correlation

[1] http://mattmahoney.net/dc/text8.zip

Table 1: Examples illustrating the weak correlation between term frequency and informativeness, measured in terms of the variance $Var[e_j^{info}]$.

Frequent and informative

	Term Frequency	$Var[e_j^{info}]$
one	411764	1.39
time	21412	0.720
states	14916	0.259
united	14494	0.282
city	12275	0.221
university	10195	0.632
french	8736	0.270
two	192644	0.815
american	20477	1.26
government	11323	1.54

Frequent and uninformative

	Term Frequency	$Var[e_j^{info}]$
in	372201	58.08
was	112807	43.16
or	68945	57.10
his	62603	47.05
also	44358	44.87
their	31523	84.35
used	22737	31.80
these	19864	25.96
e	11426	45.75
without	5661	30.38

Infrequent and uninformative

	Term Frequency	$Var[e_j^{info}]$
wendell	40	29.38
actuality	42	29.75
ebne	54	30.17
christology	45	31.04
mico	45	33.38
utilised	30	21.52
reopened	54	21.32
generalizes	24	19.07
flashing	49	19.83
eitc	27	20.77

Infrequent and informative

	Term Frequency	$Var[e_j^{info}]$
psycho	56	0.05
quantization	56	0.25
residue	56	0.02
inert	54	0.98
imap	54	0.19
batsman	52	0.68
bilinear	52	0.18
crucified	50	0.08
germanium	50	0.11
lactose	50	0.45

between these two quantities is very weak, e.g. high-frequency words can be very informative. For example, the words 'family' and 'service' are frequent in Wikipedia but were still found to be highly informative context words (i.e. $Var[e_j^{info}]$ is low for these words), while stop words such as 'were' and 'is' are found to be uninformative (i.e. $Var[e_j^{info}]$ is high for these words). Similarly, there are low-frequency words which are found to be uninformative, such as 'ga', 'scoula' and 'niggle' while other low-frequency words were found to be highly informative, such as 'compactness' and 'nasdaq'. Table 1 shows a number of additional examples of words with high/low frequency and high/low variance.

3.3 Evaluating likelihood

Explicitly modelling the residual error allows us to associate a density with each word. For example, the density shown in Figure 1 intuitively captures the evidence about the embedding of the word i that is provided by the context word j. In this section, we will assume that each target word is associated with a random vector. Note that the residual error e_{ij} then is a random variable. We will evaluate the likelihood that e_{ij} takes a given value by evaluating the likelihood that e_{ij}^{count} takes a given value s and that e_{ij}^{info} takes the value $r - s$. Let $S_{ij} \subseteq \mathbb{R}$ be the set of possible values that e_{ij}^{count} can take, i.e.:

$$S_{ij} = \{E[\log Y_{ij}] - \log P(Y_{ij} = k) : 0 \leq k \leq x_i\}$$

With each target word i and context word j we can associate the density f_{ij} defined for $r \geq 0$ as:

$$f_{ij}(r) = \frac{1}{|S_{ij}|} \sum_{s \in S_{ij}} P(Y_{ij} = k) \cdot f_{ij}^{info}(r - s) \tag{3}$$

Here $f_{ij}(r)$ is the likelihood that the residual error e_{ij} takes the value r, while $f_{ij}^{info}(s)$ is the likelihood that e_{ij}^{info} takes the value s. The variance σ_{ij}^{info} of f_{ij}^{info} is given by (2). If we furthermore assume that e_{ij}^{info} is normally distributed, we obtain:

$$f_{ij}^{info}(r - s) = \mathcal{N}(r - s, 0, \sigma_{ij}^{info})$$

If we treat each context word as an independent source of evidence, we obtain the following density g_i, modelling our knowledge about the possible choices of a word vector for word i ($w_i \in \mathbb{R}^s$):

$$g_i(w_i) = \prod_{j \in J_i} f_{ij}(w_i \cdot \tilde{w}_j - s_{ij} + \tilde{b}_j) \tag{4}$$

Note that we assume that the context vectors $\tilde{w}_j$ are given.

4 Evaluation

In this section we compare our method with existing word embedding models on a range of standard benchmark tasks.

4.1 Methodology

Corpora We have used the following text collections: Wikipedia[2] (1,335,766,618 tokens), the English Gigaword corpus[3] (1,094,733,691 tokens), a concatenation of the Wikipedia and Gigaword corpora (2,430,500,309 tokens), UMBC[4] (2,714,554,484 tokens) and ClueWeb-2012 Category-B[5] (6,030,992,452 tokens). Note that the first three text collections have also been used in (Pennington et al., 2014). We adopted a straightforward text preprocessing strategy. In particular, following (Pennington et al., 2014), we have removed punctuations, lower-cased the tokens, removed HTML/XML tags, and conducted sentence segmentation. For the ClueWeb collection, we used the preprocessing implementation of the reVerb tool[6], which was specifically designed to process ClueWeb, and only considered terms which occur at least 100 times in the collection, to offset the larger size of this collection. This led to a vocabulary size of 283,701 words. For the other collections, we used our own code, which is available along with the rest of our implementation[7], and used the NLTK library[8] for sentence segmentation. As these collections are smaller than the ClueWeb collection, we considered all words that appear at least 10 times. The resulting vocabulary sizes are 1,252,101 words for Wikipedia, 469,052 words for the Gigaword corpus, 1,524,043 words for Wikipedia+Gigaword and 541,236 words for UMBC. When counting word co-occurrence statistics, we do not cross sentence boundaries. Similar to GloVe, words in the context windows in our model were weighted using the harmonic function. For the baseline models, we used the context word weighting scheme from their original implementations.

Baseline methods and variants We consider the following state-of-the-art word embedding baselines: the Skip-Gram (SG) and Continuous-Bag-of-Words (CBOW) models from (Mikolov et al., 2013), GloVe (Pennington et al., 2014), and the Gaussian word embedding model (Gauss) from (Vilnis and McCallum, 2015). In all cases, we have used existing implementations of these models[9][10][11]. Furthermore, we have considered several variants of our model to better understand what components are responsible for the improvements over GloVe. In the standard version, we estimate the similarity between words w_i and w_j by evaluating the likelihood $g_i(w_j)$, as defined in (4). In variant DG-ZC we instead use cosine similarity, as in the GloVe model. In variant DG-C we also use the cosine similarity and in addition set J_i as in the GloVe model (i.e. we disregard pairs (i, j) for which $x_{ij} = 0$). Variant DG-UfL differs from the standard model by not considering the error term e_j^{info}.

Evaluation tasks We have evaluated the models on traditional word analogy and word similarity tasks (Levy et al., 2015). In particular, we have used an existing Google Word analogy dataset which we obtained from the GloVe project[12]. In addition, we have used the Microsoft Word analogy dataset[13] as well as twelve existing word similarity datasets[14]. The aim of these evaluation tasks has been explained in detail in (Levy et al., 2015). A new evaluation task for word embedding has recently been proposed

[2] We used the dump from November 2nd, 2015.
[3] https://catalog.ldc.upenn.edu/LDC2011T07
[4] http://ebiquity.umbc.edu/resource/html/id/351
[5] http://lemurproject.org/clueweb12/
[6] http://reverb.cs.washington.edu/
[7] https://github.com/bashthebuilder/pGlove
[8] http://www.nltk.org/
[9] https://code.google.com/archive/p/word2vec/
[10] http://nlp.stanford.edu/projects/glove/
[11] https://github.com/seomoz/word2gauss
[12] http://nlp.stanford.edu/projects/glove/
[13] https://bitbucket.org/omerlevy/hyperwords/src
[14] https://github.com/mfaruqui/retrofitting

Table 2: Comparison with baseline methods on standard word embedding evaluation tasks.

	Gsem	Gsyn	MSR	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	Outlier	
	Acc			Spearman's ρ												Acc	OPP
	Wikipedia																
SG	71.6	**64.2**	**68.6**	0.658	0.773	0.784	**0.645**	**0.708**	**0.456**	0.500	**0.415**	**0.435**	0.773	0.655	0.731	70.3	93.8
CBOW	74.2	62.4	66.2	0.644	0.768	0.740	0.532	0.622	0.419	0.341	0.361	0.343	0.707	0.597	0.693	73.4	95.3
Gauss	61.3	53.3	43.8	0.593	0.632	0.681	0.409	0.506	0.256	0.392	0.337	0.416	0.649	0.601	0.644	04.6	40.0
GloVe	80.2	58.0	50.3	0.595	0.755	0.746	0.515	0.577	0.318	0.533	0.382	0.354	0.690	0.652	0.724	58.8	92.6
D-GloVe	**81.4**	59.1	59.6	**0.670**	**0.789**	**0.789**	0.560	0.658	0.401	**0.540**	0.413	0.391	**0.780**	**0.656**	**0.749**	**73.5**	**96.1**
	Gigaword																
SG	61.5	**63.2**	**67.5**	0.676	0.628	0.594	**0.550**	**0.614**	**0.446**	0.408	**0.422**	**0.408**	**0.691**	0.621	**0.696**	74.9	84.1
CBOW	50.2	58.1	64.8	0.615	0.568	0.600	0.416	0.518	0.405	0.259	0.347	0.343	0.625	0.520	0.610	74.1	84.0
Gauss	38.2	45.1	40.1	0.600	0.474	0.548	0.413	0.507	0.326	0.223	0.307	0.204	0.504	0.473	0.567	56.0	66.2
GloVe	64.4	59.6	55.8	0.600	0.669	**0.599**	0.511	0.535	0.336	0.486	0.327	0.255	0.593	0.606	0.668	74.2	83.2
D-GloVe	**65.5**	61.5	58.9	**0.697**	**0.673**	**0.599**	0.521	0.555	0.394	**0.499**	0.387	0.289	0.663	**0.622**	**0.696**	**75.2**	**86.0**
	Gigaword+Wikipedia																
SG	74.4	**69.6**	**69.3**	0.678	0.712	0.794	**0.659**	**0.719**	**0.459**	0.511	**0.518**	**0.437**	0.731	0.631	0.733	82.8	91.6
CBOW	72.2	61.2	66.2	0.633	0.699	0.681	0.528	0.592	0.419	0.321	0.405	0.359	0.688	0.561	0.662	80.1	90.1
Gauss	56.1	53.2	51.9	0.601	0.583	0.619	0.421	0.518	0.311	0.318	0.332	0.319	0.581	0.557	0.617	40.1	55.9
GloVe	78.8	66.9	58.6	0.608	0.741	0.735	0.598	0.581	0.388	0.578	0.399	0.357	0.711	0.616	0.719	81.8	89.6
D-GloVe	**86.8**	67.2	66.3	**0.689**	**0.749**	**0.799**	0.606	0.589	**0.459**	**0.589**	0.482	0.401	**0.743**	**0.640**	**0.742**	**85.2**	**92.5**
	UMBC																
SG	62.0	65.8	**68.7**	0.619	0.778	0.753	**0.594**	**0.620**	**0.355**	0.572	0.390	0.367	**0.684**	0.664	0.735	76.2	86.2
CBOW	73.5	**67.4**	65.3	0.619	0.768	0.733	0.586	0.616	0.345	0.577	0.347	0.352	**0.684**	0.658	0.723	76.1	85.3
Gauss	56.7	64.4	55.2	0.614	0.764	0.742	0.583	0.608	0.342	0.571	0.362	0.344	0.674	0.652	0.717	45.9	59.2
GloVe	65.9	66.5	65.2	0.618	0.770	0.731	0.587	0.613	0.344	0.572	0.374	0.363	0.679	0.660	0.723	75.9	85.2
D-GloVe	**77.5**	66.7	65.3	**0.620**	**0.796**	**0.756**	0.591	0.618	**0.355**	**0.592**	**0.394**	**0.368**	**0.684**	**0.667**	**0.736**	**77.1**	**87.1**
	ClueWeb12-B																
SG	37.3	58.9	**87.5**	0.674	0.725	0.713	**0.632**	0.680	0.463	**0.384**	0.389	0.388	0.730	0.643	0.718	86.7	98.1
CBOW	50.0	**61.7**	**87.5**	0.636	0.702	0.704	0.514	0.612	0.422	0.329	0.362	0.367	0.691	0.612	0.668	86.4	97.9
Gauss	39.5	49.0	72.1	0.611	0.664	0.670	0.416	0.520	0.261	0.314	0.339	0.312	0.669	0.599	0.647	75.8	81.7
GloVe	48.9	51.7	85.2	0.651	0.724	0.720	0.621	0.681	0.421	0.321	0.356	0.361	0.700	0.619	0.678	79.8	97.1
D-GloVe	**56.7**	60.4	87.0	**0.675**	**0.744**	**0.736**	0.629	**0.683**	**0.533**	0.383	**0.390**	**0.389**	**0.731**	**0.653**	**0.724**	**86.8**	**98.2**

Table 3: Results for different variants of our model on the Wikipedia collection.

	Gsem	Gsyn	MSR	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	Outlier	
	Acc			Spearman's ρ												Acc	OPP
D-GloVe	81.4	59.1	59.6	0.670	0.789	0.789	0.560	0.658	0.401	0.540	0.413	0.391	0.780	0.656	0.749	73.5	96.1
DG-ZC	80.9	58.8	51.8	0.659	0.781	0.786	0.521	0.589	0.320	0.533	0.383	0.370	0.779	0.661	0.747	66.1	95.0
DG-C	80.8	58.3	51.5	0.659	0.781	0.784	0.518	0.581	0.321	0.533	0.382	0.361	0.778	0.661	0.740	62.8	94.9
DG-UfL	79.9	56.2	50.1	0.615	0.763	0.758	0.491	0.568	0.311	0.509	0.376	0.349	0.709	0.645	0.704	61.8	93.7

in (Camacho-Collados and Navigli, 2016), which we have also considered, using the evaluation script provided by the authors[15]. The aim of this task is to find the outlier in a given set of words. We refer to (Camacho-Collados and Navigli, 2016) for a detailed explanation of the task and the considered evaluation metrics.

Parameter tuning We select the parameters for each of the methods using a 25% validation set and report results on the remaining 75% of each evaluation set. The parameters were tuned separately for each of the evaluation tasks. For CBOW and SG, we chose the number of negative samples from a pool of {1, 5, 10, 15}. For GloVe, we selected the x_{max} value from {10, 50, 100} and α from {0.1, 0.25, 0.5, 0.75, 1}. For the Gaussian word embedding approach, we used the spherical Gaussian with KL-divergence model. For our model, we selected the Dirichlet prior constant λ from {0.0001, 0.001, 0.01, 0.1, 1000, 2000, 5000, 8000}. For all models, the number of dimensions was chosen from {100, 300}, the size of the context windows was chosen from {2, 5, 10}, and the number of iterations was fixed as 50. In our model, we re-estimate the variances σ_{ij}^2 every five iterations.

[15] http://lcl.uniroma1.it/outlier-detection/

Table 4: Results for high-frequency and low-frequency words for Wikipedia (left) and UMBC (right).

Most frequent	S4	S5	S6	S8	S9
SG	0.504	0.452	0.598	0.711	0.596
D-GloVe	**0.530**	**0.560**	**0.650**	**0.773**	**0.724**
Least frequent	S4	S5	S6	S8	S9
SG	**0.623**	**0.445**	**0.400**	**0.560**	**0.559**
D-GloVe	0.579	0.328	0.200	0.182	0.245

Most frequent	S4	S5	S6	S10
SG	0.511	0.256	0.591	0.287
D-GloVe	**0.575**	**0.354**	**0.626**	**0.333**
Least frequent	S4	S5	S6	S10
SG	**0.661**	**0.372**	**0.100**	**0.331**
D-GloVe	0.605	0.312	0.091	0.313

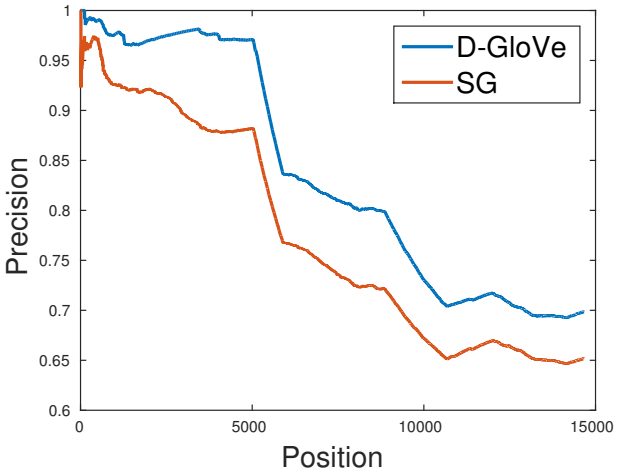

Figure 3: Confidence ranking plot for the Google word analogy test set.

4.2 Results

We present our main results in Table 2. The word similarity datasets are indexed as: S1: EN-MTurk-287, S2: EN-RG-65, S3: EN-MC-30, S4: EN-WS-353-REL, S5: EN-WS-353-ALL, S6: EN-RW-STANFORD, S7-EN-YP-130, S8-EN-SIMLEX-999, S9-EN-VERB-143, S10-EN-WS-353-SIM, S11: EN-MTurk-771, and S12: EN-MEN-TR-3k. We can see from the results that our model consistently outperforms GloVe. To further understand what components are responsible for this improvement, Table 3 shows the result for some variants of our model, showing that each of the proposed adaptations of the GloVe model contributes to the overall result. Compared to the other baselines, the results in Table 2 show that our model performs substantially better for the semantic instances of the Google analogy datasets (Gsem), while it is outperformed by SG (and in some cases CBOW) for the syntactic instances (Gsyn) and for the Microsoft dataset (MSR), which contains only syntactic instances. Our model also outperforms the baselines for the outlier detection task. For the similarity test instances, the performance is mixed. What is noticeable is that our model performs comparatively better for large corpora (e.g. ClueWeb) and worse for smaller corpora (e.g. Gigaword).

In Table 4 we present a more detailed analysis of the similarity test sets for which our model performs worse than SG. In particular, the table shows the results of a modified test set that only considers the 30% most frequent terms and a modified test set that only considers the 30% least frequent terms. These results clearly show that our model outperforms SG for high-frequency terms and that it is outperformed by SG for low-frequency terms. Dirichlet-Multinomial model are indeed known to struggle with low-frequency terms (Sridhar, 2015), which can e.g. be addressed by the use of asymmetric Dirichlet priors (Wallach et al., 2009). Note that this observation also explains why our model performs comparatively better for larger corpora and why it performs worse for syntactic analogy instances (given that such instances tend to contain low-frequency terms). While this can be seen as a limitation of our model, the fact that our model treats low-frequency terms in a cautious way may actually be advantageous in downstream applications.

An advantage of our approach is that the likelihood based formulation naturally allows us to estimate the confidence that a given prediction is correct. In Figure 4.2 we show the accuracy for the k analogy

instances of the Google dataset about which our model was most confident, for varying values of k. Similarly, the figure also shows the accuracy of the predictions made by SG, ranked in terms of cosine similarity. As can be seen, our model is better able to identify those instances that it can answer correctly (e.g. the accuracy of the top 5000 instances remains close to 1).

5 Conclusions

We have proposed a new word embedding model in which each word is represented as a density, obtained by associating with each word i and each context word j a univariate density. These univariate densities are in turn obtained by explicitly modelling the residual error of the considered least squares optimization function. Our experiments reveal that the model consistently outperforms the GloVe model, on which it is based. The proposed model also outperforms skip-gram, and other baselines, for high-frequency terms. For low-frequency terms, our model takes a rather cautious approach, which means that it is often outperformed by skip-gram in standard evaluation settings.

6 Acknowledgments

This work was supported by ERC Starting Grant 637277. This work was performed using the computational facilities of the Advanced Research Computing@Cardiff (ARCCA) Division, Cardiff University.

References

[Bullinaria and Levy2007] John A Bullinaria and Joseph P Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods*, 39:510–526.

[Camacho-Collados and Navigli2016] Jose Camacho-Collados and Roberto Navigli. 2016. Find the word that does not belong: A framework for an intrinsic evaluation of word vector representations. In *Proceedings of the ACL Workshop on Evaluating Vector Space Representations for NLP*.

[Erk2009] Katrin Erk. 2009. Representing words as regions in vector space. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 57–65.

[Faruqui et al.2015] Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard H. Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615.

[Harris1954] Zellig Harris. 1954. Distributional structure. *Word*, 10:146–162.

[Huang et al.2012] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 873–882.

[Landauer and Dumais1997] Thomas K Landauer and Susan T Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104:211–240.

[Levy and Goldberg2014] Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *Proceedings of the Annual Conference on Neural Information Processing Systems*, pages 2177–2185.

[Levy et al.2015] Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

[Li et al.2015] Shaohua Li, Jun Zhu, and Chunyan Miao. 2015. A generative word embedding model and its low rank positive semidefinite solution. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1599–1609.

[Liu et al.2015] Yang Liu, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. 2015. Topical word embeddings. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2418–2424.

[Lund and Burgess1996] Kevin Lund and Curt Burgess. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, & Computers*, 28:203–208.

[Maas and Ng2010] Andrew L Maas and Andrew Y Ng. 2010. A probabilistic model for semantic word vectors. In *Proceedings of the NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.

[Mikolov et al.2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 27th Annual Conference on Neural Information Processing Systems*, pages 3111–3119.

[Neelakantan et al.2015] Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. 2015. Efficient non-parametric estimation of multiple embeddings per word in vector space. *arXiv:1504.06654*.

[Pennington et al.2014] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

[Reisinger and Mooney2010] Joseph Reisinger and Raymond J Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117.

[Sridhar2015] Vivek Kumar Rangarajan Sridhar. 2015. Unsupervised topic modeling for short texts using distributed representations of words. In *Proceedings of the 2015 Conference of theNorth American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 192–200.

[Turney and Pantel2010] P. D. Turney and P. Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

[Vilnis and McCallum2015] Luke Vilnis and Andrew McCallum. 2015. Word representations via Gaussian embedding. In *Proceedings of the International Conference on Learning Representations*.

[Wallach et al.2009] Hanna M Wallach, David M Mimno, and Andrew McCallum. 2009. Rethinking LDA: Why priors matter. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*, pages 1973–1981.

[Xu et al.2014] Chang Xu, Yalong Bai, Jiang Bian, Bin Gao, Gang Wang, Xiaoguang Liu, and Tie-Yan Liu. 2014. RC-NET: A general framework for incorporating knowledge into word representations. In *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management*, pages 1219–1228.

[Yu and Dredze2014] Mo Yu and Mark Dredze. 2014. Improving lexical embeddings with semantic knowledge. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 545–550.

[Zhai and Lafferty2004] Chengxiang Zhai and John Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems*, 22:179–214.