Can Network Embedding of Distributional Thesaurus be Combined with Word Vectors for Better Representation?

Abhik Jana
IIT Kharagpur
Kharagpur, India
abhik.jana@iitkgp.ac.in

Pawan Goyal
IIT Kharagpur
Kharagpur, India
pawang@cse.iitkgp.ac.in

Abstract

Distributed representations of words learned from text have proved to be successful in various natural language processing tasks in recent times. While some methods represent words as vectors computed from text using predictive model (Word2vec) or dense count based model (GloVe), others attempt to represent these in a distributional thesaurus network structure where the neighborhood of a word is a set of words having adequate context overlap. Being motivated by recent surge of research in network embedding techniques (DeepWalk, LINE, node2vec etc.), we turn a distributional thesaurus network into dense word vectors and investigate the usefulness of distributional thesaurus embedding in improving overall word representation. This is the first attempt where we show that combining the proposed word representation obtained by distributional thesaurus embedding with the state-of-the-art word representations helps in improving the performance by a significant margin when evaluated against NLP tasks like word similarity and relatedness, synonym detection, analogy detection. Additionally, we show that even without using any handcrafted lexical resources we can come up with representations having comparable performance in the word similarity and relatedness tasks compared to the representations where a lexical resource has been used.

1 Introduction

Natural language understanding has always been a primary challenge in natural language processing (NLP) domain. Learning word representations is one of the basic and primary steps in understanding text and nowadays there are predominantly two views of learning word representations. In one realm of representation, words are vectors of distributions obtained from analyzing their contexts in the text and two words are considered meaningfully similar if the vectors of those words are close in the euclidean space. In recent times, attempts have been made for dense representation of words, be it using predictive model like Word2vec (Mikolov et al., 2013) or count-based model like GloVe (Pennington et al., 2014) which are computationally efficient as well. Another stream of representation talks about network like structure where two words are considered neighbors if they both occur in the same context above a certain number of times. The words are finally represented using these neighbors. Distributional Thesaurus is one such instance of this type, which gets automatically produced from a text corpus and identifies words that occur in similar contexts; the notion of which was used in early work about distributional semantics (Grefenstette, 2012; Lin, 1998; Curran and Moens, 2002). One such representation is JoBimText proposed by Biemann and Riedl (2013) that contains, for each word, a list of words that are similar with respect to their bigram distribution, thus producing a network representation. Later, Riedl and Biemann (2013) introduced a highly scalable approach for computing this network. We mention this representation as a DT network throughout this article. With the emergence of recent trend of embedding large networks into dense low-dimensional vector space efficiently (Perozzi et al., 2014; Tang et al., 2015; Grover and Leskovec, 2016) which are focused on capturing different properties of the network like neighborhood structure, community structure, etc., we explore representing DT network in a dense vector space and evaluate its useful application in various NLP tasks.

There has been attempt (Ferret, 2017) to turn distributional thesauri into word vectors for synonym extraction and expansion but the full utilization of DT embedding has not yet been explored. In this paper, as a main contribution, we

Proceedings of NAACL-HLT 2018, pages 463–473
New Orleans, Louisiana, June 1 - 6, 2018.

investigate the best way of turning a Distributional Thesaurus (DT) network into word embeddings by applying efficient network embedding methods and analyze how these embeddings generated from DT network can improve the representations generated from prediction-based model like Word2vec or dense count based semantic model like GloVe. We experiment with several combination techniques and find that DT network embedding can be combined with Word2vec and GloVe to outperform the performances when used independently. Further, we show that we can use DT network embedding as a proxy of WordNet embedding in order to improve the already existing state-of-the-art word representations as both of them achieve comparable performance as far as word similarity and word relatedness tasks are concerned. Considering the fact that the vocabulary size of WordNet is small and preparing WordNet like lexical resources needs huge human engagement, it would be useful to have a representation which can be generated automatically from corpus. We also attempt to combine both WordNet and DT embeddings to improve the existing word representations and find that DT embedding still has some extra information to bring in leading to better performance when compared to combination of only WordNet embedding and state-of-the-art word embeddings. While most of our experiments are focused on word similarity and relatedness tasks, we show the usefulness of DT embeddings on synonym detection and analogy detection as well. In both the tasks, combined representation of GloVe and DT embeddings shows promising performance gain over state-of-the-art embeddings.

2 Related Work

The core idea behind the construction of distributional thesauri is the distributional hypothesis (Firth, 1957): "You should know a word by the company it keeps". The semantic neighbors of a target word are words whose contexts overlap with the context of a target word above a certain threshold. Some of the initial attempts for preparing distributional thesaurus are made by Lin (1998), Curran and Moens (2002), Grefenstette (2012). The semantic relation between a target word and its neighbors can be of different types, e.g., synonymy, hypernymy, hyponymy or other relations (Adam et al., 2013; Budanitsky and Hirst, 2006) which prove to be very useful in different natural language tasks. Even though computation of sparse count based models used to be inefficient, in this era of high speed processors and storage, attempts are being made to streamline the computation with ease. One such effort is made by Kilgarriff et al. (2004) where they propose Sketch Engine, a corpus tool which takes as input a corpus of any language and corresponding grammar patterns, and generates word sketches for the words of that language and a thesaurus. Recently, Riedl and Biemann (2013) introduce a new highly scalable approach for computing quality distributional thesauri by incorporating pruning techniques and using a distributed computation framework. They prepare distributional thesaurus from Google book corpus in a network structure and make it publicly available.

In another stream of literature, word embeddings represent words as dense unit vectors of real numbers, where vectors that are close together in euclidean space are considered to be semantically related. In this genre of representation, one of the captivating attempt is made by Mikolov et al. (2013), where they propose Word2vec, basically a set of two predictive models for neural embedding whereas Pennington et al. (2014) propose GloVe, which utilizes a dense count based model to come up with word embeddings that approximate this. Comparisons have also been made between count-based and prediction-based distributional models (Baroni et al., 2014) upon various tasks like relatedness, analogy, concept categorization etc., where researchers show that prediction-based word embeddings outperform sparse count-based methods used for computing distributional semantic models. In other study, Levy and Goldberg (2014) show that dense count-based methods, using PPMI weighted co-occurrences and SVD, approximates neural word embeddings. Later, Levy et al. (2015) show the impact of various parameters and the best performing parameters for these methods. All these approaches are completely text based; no external knowledge source has been used.

More recently, a new direction of investigation has been opened up where researchers are trying to combine knowledge extracted from knowledge bases, images with distributed word representations prepared from text with the expectation of getting better representation. Some use

Knowledge bases like WordNet (Miller, 1995), FreeBase (Bollacker et al., 2008), PPDB (Ganitkevitch et al., 2013), ConceptNet (Speer et al., 2017), whereas others use ImageNet (Frome et al., 2013; Kiela and Bottou, 2014; Both et al., 2017; Thoma et al., 2017) for capturing visual representation of lexical items. There are various ways of combining multiple representations. Some of the works extract lists of relations from knowledge bases and use those to either modify the learning algorithms (Halawi et al., 2012; Wang et al., 2014; Tian et al., 2016; Rastogi et al., 2015) or post-process pre-trained word representations (Faruqui et al., 2015). Another line of literature prepares dense vector representation from each of the modes (text, knowledge bases, visual etc.) and tries to combine the vectors using various methods like concatenation, centroid computation, principal component analysis (Jolliffe, 1986), canonical correlation analysis (Faruqui and Dyer, 2014) etc. One such recent attempt is made by Goikoetxea et al. (2016) where they prepare vector representation from WordNet following the method proposed by Goikoetxea et al. (2015), which combines random walks over knowledge bases and neural network language model, and tries to improve the vector representation constructed from text using this. As in lexical knowledge bases, the number of lexical items involved is much less than the raw text and preparing such resources is a cumbersome task, our goal is to see whether we can use DT network instead of some knowledge bases like WordNet and achieve comparable performance on NLP tasks like word similarity and word relatedness. In order to prepare vector representation from DT network, we attempt to use various network embeddings like DeepWalk (Perozzi et al., 2014), LINE (Tang et al., 2015), struc2vec (Ribeiro et al., 2017), node2vec (Grover and Leskovec, 2016) etc. Some of those try to capture the neighbourhood or community structure in the network while others attempt to capture structural similarity between nodes, second order proximity, etc.

3 Proposed Methodology

Our aim is to analyze the effect of integrating the knowledge of Distributional Thesaurus network with the state-of-the-art word representation models to prepare a better word representation. We first prepare vector representations from Distributional Thesaurus (DT) network applying network representation learning model. Next we combine this thesaurus embedding with state-of-the-art vector representations prepared using GloVe and Word2vec model for analysis.

3.1 Distributional Thesaurus (DT) Network

Riedl and Biemann (2013) use the Google books corpus, consisting of texts from over 3.4 million digitized English books published between 1520 and 2008 and construct a distributional thesauri (DT) network using the syntactic n-gram data (Goldberg and Orwant, 2013). The authors first compute the lexicographer's mutual information (LMI) (Kilgarriff et al., 2004) for each bigram, which gives a measure of the collocational strength of a bigram. Each bigram is broken into a word and a feature, where the feature consists of the bigram relation and the related word. Then the top 1000 ranked features for each word are taken and for each word pair, intersection of their corresponding feature set is obtained. The word pairs having number of overlapping features above a threshold are retained in the network. In a nutshell, the DT network contains, for each word, a list of words that are similar with respect to their bigram distribution (Riedl and Biemann, 2013). In the network, each word is a node and there is a weighted edge between a pair of words where the weight corresponds to the number of overlapping features. A sample snapshot of the DT is shown in Figure 1.

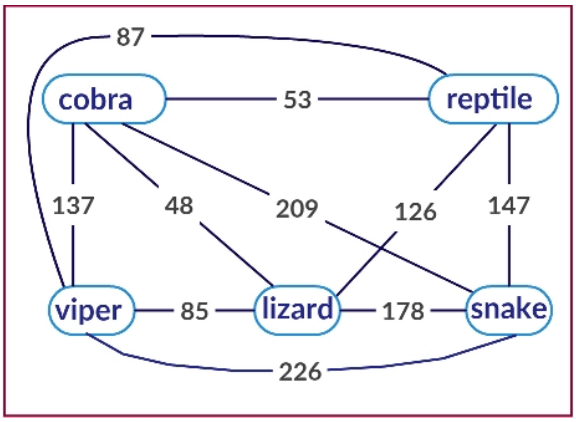

Figure 1: A sample snapshot of Distributional Thesaurus network where each node represents a word and the weight of an edge between two words is defined as the number of context features that these two words share in common.

3.2 Embedding Distributional Thesaurus

Now, from the DT network, we prepare the vector representation for each node using network representation learning models which produce vector representation for each of the node in a network. For this purpose, we use three state-of-the-art network representation learning models as discussed below.

DeepWalk: DeepWalk (Perozzi et al., 2014) learns social representations of a graph's vertices by modeling a stream of short random walks. Social representations signify latent features of the vertices that capture neighborhood similarity and community membership.

LINE: LINE (Tang et al., 2015) is a network embedding model suitable for arbitrary types of networks: undirected, directed and/or weighted. The model optimizes an objective which preserves both the local and global network structures by capturing both first-order and second-order proximity between vertices.

node2vec: node2vec (Grover and Leskovec, 2016) is a semi-supervised algorithm for scalable feature learning in networks which maximizes the likelihood of preserving network neighborhoods of nodes in a d-dimensional feature space. This algorithm can learn representations that organize nodes based on their network roles and/or communities they belong to by developing a family of biased random walks, which efficiently explore diverse neighborhoods of a given node.

Note that, by applying network embedding models on DT network we obtain 128 dimensional vectors for each word in the network. We only consider edges of the DT network having edge weight greater or equal to 50 for network embedding. Henceforth, we will use *D2V-D*, *D2V-L* and *D2V-N* to indicate vector representations obtained from DT network produced by DeepWalk, LINE and node2vec, respectively.

After obtaining vector representations, we also explore whether these can be combined with the pre-trained vector representation of Word2vec and GloVe to come up with a joint vector representation. For that purpose, we directly use very well-known GloVe 1.2 embeddings (Pennington et al., 2014) trained on 840 billion words of the common crawl dataset having vector dimension of 300. As an instance of pre-trained vector of Word2vec, we use prominent pre-trained vector representations prepared by Mikolov et al. (2013) trained on 100 billion words of Google News using skip-grams with negative sampling, having dimension of 300.

3.3 Vector Combination Methods

In order to integrate the word vectors, we apply two strategies inspired by Goikoetxea et al. (2016): concatenation (CC) and principal component analysis (PCA).

Concatenation (CC): This corresponds to the simple vector concatenation operation. Vector representations of both GloVe and Word2vec are of 300 dimensions and word embeddings learnt form DT are of 128 dimensions. The concatenated representation we use are of 428 dimensions.

Principal Component Analysis (PCA): Principal component analysis (Jolliffe, 1986) is a dimensionality reduction statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components (linear combinations of the original variables). We apply PCA to the concatenated representations (dimension of 428) reducing these to 300 dimensions. In addition to PCA, we try with truncated singular value decomposition procedure (Hansen, 1987) as well, but as per the experiment set up, it shows negligible improvement in performance compared to simple concatenation; hence we do not continue with the truncated singular value decomposition for dimensionality reduction. After obtaining the combined representations of words, we head towards evaluating the quality of the representation.

4 Experiments and Analysis

In order to evaluate the quality of the word representations, we first conduct qualitative analysis of the joint representation. Next, we follow the most acceptable way of applying on different NLP tasks like word similarity and word relatedness, synonym detection and word analogy as described next.

4.1 Qualitative Analysis:

On qualitative analysis of some of the word pairs from the evaluation dataset, we observe that the joint representation (PCA (GloVe,D2V-N)) captures the notion of similarity much better than GloVe. For example, it gives a higher cosine similarity scores to the pairs (car, cab), (sea, ocean), (cottage,cabin), (vision, perception) etc. in com-

Dataset	GloVe	W2V	D2V-D	D2V-L	D2V-N
WSSim	**0.799**	0.779	0.737	0.073	0.764
SimL-N	0.427	**0.454**	0.418	0.015	0.421
RG-65	0.791	0.777	0.804	-0.121	**0.813**
MC-30	0.799	0.819	0.859	-0.067	**0.869**
WSR	**0.637**	0.631	0.287	0.077	0.333
M771	**0.707**	0.655	0.636	0.027	0.63
M287	**0.8**	0.755	0.558	-0.027	0.591
MEN-N	**0.819**	0.764	0.619	0.004	0.612
WS-353	**0.706**	0.697	0.51	0.088	0.547

Table 1: Comparison of individual performances of different vector representation models w.r.t. word similarity and relatedness tasks. The performance metric is Spearman's rank correlation coefficient (ρ). Best result of each row in bold showing the best vector representation for each dataset.

parison to GloVe. However, in some cases, where words are not similar but are related, e.g., (airport, flight), (food, meat), (peeper, soup), (harbour, shore), the joint representation gives a lower cosine similarity score than GloVe comparatively. In the next set of evaluation experiments, we observe this utility of joint representation towards word similarity task and word relatedness task to some extent.

4.2 Word Similarity and Relatedness

In this genre of tasks, the human judgment score for each word pair is given; we report the Spearman's rank correlation coefficient (ρ) between human judgment score and the predicted score by distributional model. Note that, we take cosine similarity between vector representations of words in a word pair as the predicted score.

Datasets: We use the benchmark datasets for evaluation of word representations. Four word similarity datasets and four word relatedness datasets are used for that purpose. The descriptions of the word similarity datasets are given below.

WordSim353 Similarity (WSSim) : 203 word pairs extracted from WordSim353 dataset (Finkelstein et al., 2001) by manual classification, prepared by Agirre et al. (2009), which deals with only similarity.

SimLex999 (SimL) : 999 word pairs rated by 500 paid native English speakers, recruited via Amazon Mechanical Turk,[1] who were asked to rate the similarity. This dataset is introduced by Hill et al. (2016).

RG-65 : It consists of 65 word pairs collected by Rubenstein and Goodenough (1965). These word pairs are judged by 51 humans in a scale from 0 to 4 according to their similarity, but ignoring any other possible semantic relationships.

MC-30 : 30 words judged by 38 subjects in a scale of 0 and 4 collected by Miller and Charles (1991).

Similarly, a brief overview of word relatedness datasets is given below:

WordSim353 Relatedness (WSR) : 252 word pairs extracted from WordSim353 (Finkelstein et al., 2001) dataset by manual classification, prepared by Agirre et al. (2009) which deals with only relatedness.

MTURK771 (M771) : 771 word pairs evaluated by Amazon Mechanical Turk workers, with an average of 20 ratings for each word pair, where each judgment task consists of a batch of 50 word pairs. Ratings are collected on a 15 scale. This dataset is introduced by Halawi et al. (2012).

MTURK287 (M287) : 287 word pairs evaluated by Amazon Mechanical Turk workers, with an average of 23 ratings for each word pair. This dataset is introduced by Radinsky et al. (2011).

MEN : MEN consists of 3,000 word pairs with [0, 1]-normalized semantic relatedness ratings provided by Amazon Mechanical Turk workers. This dataset was introduced by Bruni et al. (2014).

Along with these datasets we use the full **WordSim353 (WS-353)** dataset (includes both similarity and relatedness pairs) (Finkelstein et al., 2001) which contains 353 word pairs, each associated with an average of 13 to 16 human judgments in a scale of 0 to 10. Being inspired by Baroni et al. (2014), we consider only noun pairs from **SimL** and **MEN** datasets, which will be denoted as **SimL-N** and **MEN-N** whereas other datasets only contain the noun pairs.

We start with experiments to inspect individual performance of each of the vector representations for each of the datasets. Table 1 represents individual performances of GloVe, Word2vec, *D2V-*

[1] www.mturk.com

Dataset	GloVe	CC (GloVe,D2V-D)	PCA (GloVe,D2V-D)	CC (GloVe,D2V-N)	PCA (GloVe,D2V-N)
WSSim	0.799	0.838	0.839	**0.84**	0.832
SimL-N	0.427	0.443	0.468	0.446	**0.483**
RG-65	0.791	0.816	**0.879**	0.809	0.857
MC-30	0.799	0.86	**0.89**	0.866	0.874
WSR	0.637	**0.676**	0.645	0.67	0.657
M771	0.707	0.708	0.707	0.711	**0.719**
M287	0.8	0.781	0.807	0.795	**0.82**
MEN-N	**0.819**	0.792	0.799	0.806	0.817
WS-353	0.706	**0.751**	0.74	0.75	0.75

Table 2: Comparison of performances (Spearman's ρ) of GloVe against the combined representation of word representations obtained from DT network using network embeddings (DeepWalk, node2vec) with GloVe. Two combination methods – concatenation (CC) and PCA – are used among which PCA performs better than concatenation (CC) in most of the cases. Also the results show that the combined representation leads to better performance in almost all the cases.

Dataset	W2V	CC (W2V,D2V-D)	PCA (W2V,D2V-D)	CC (W2V,D2V-N)	PCA (W2V,D2V-N)
WSSim	0.779	0.774	0.786	**0.806**	0.805
SimL-N	0.454	0.438	0.456	0.448	**0.493**
RG-65	0.777	0.855	0.864	0.867	**0.875**
MC-30	0.819	0.866	0.891	0.903	**0.909**
WSR	**0.631**	0.441	0.443	0.459	0.497
M771	0.655	0.633	0.637	0.656	**0.676**
M287	**0.755**	0.714	0.701	0.722	**0.755**
MEN-N	**0.764**	0.703	0.717	0.714	0.747
WS-353	**0.697**	0.602	0.61	0.623	0.641

Table 3: A similar experiment as Table 2 with Word2vec (W2V) instead of GloVe.

Dataset	PCA (GloVe,W2V)	PCA (GloVe,D2V-N)
WSSim	0.8	**0.832**
SimL-N	0.476	**0.483**
RG-65	0.794	**0.857**
MC-30	0.832	**0.874**
WSR	**0.68**	0.657
M771	0.717	**0.719**
M287	**0.82**	**0.82**
MEN-N	**0.829**	0.817
WS-353	0.746	**0.75**

Table 4: Comparison of performances (Spearman's ρ) between GloVe combined with Word2vec (W2V) against GloVe combined with DT embedding obtained using node2vec (D2V-N). PCA has been taken as combination method. Clearly, DT embedding outperforms Word2vec in terms of enhancing the performance of GloVe.

Dataset	PCA (GloVe, D2V-N)	PCA (GloVe, WN2V)	PCA (GloVe, WN2V, D2V-N)
WSSim	0.832	0.828	**0.853**
SimL-N	0.483	0.525	**0.531**
RG-65	0.857	0.858	**0.91**
MC-30	0.874	0.882	**0.92**
WSR	0.657	**0.699**	0.682
M771	0.719	0.762	**0.764**
M287	**0.82**	0.816	0.81
MEN-N	0.817	**0.848**	0.7993
WS-353	0.75	**0.7801**	0.7693

Table 5: Performance (ρ) reported for three combined representations: GloVe and DT embedding using node2vec (D2V-N), GloVe and WordNet embedding (WN2V), GloVe, WN2V and D2V-N. Results show that, DT embedding produces comparable performance in comparison to the WordNet embedding. Combining DT embedding along with WordNet embedding helps to boost performance further in many of the cases.

D, *D2V-L* and *D2V-N* for different datasets. In most of the cases, GloVe produces the best results although no model is a clear winner for all the datasets. Interestingly, *D2V-D* and *D2V-N* give results comparable to GloVe and Word2vec for the word similarity datasets, even surpassing GloVe and Word2vec for few of these. *D2V-L* gives very poor performance, indicating that considering second order proximity in the DT network while embedding has an adverse effect on performance in word similarity and word relatedness tasks, whereas random walk based *D2V-D* and *D2V-N* which take care of neighborhood and community, produce decent performance. Hence-

Dataset	GloVe	GloVe with retrofitting	PCA (GloVe,D2V-N)
WSSim	0.799	0.799	**0.832**
SimL-N	0.427	0.423	**0.483**
RG-65	0.791	0.791	**0.857**
MC-30	0.799	0.799	**0.874**
WSR	0.637	**0.69**	0.657
M771	0.707	0.708	**0.719**
M287	0.8	0.795	**0.82**
MEN-N	**0.819**	**0.819**	0.817
WS-353	0.706	0.703	**0.75**

Table 6: Comparison of performances (Spearman's ρ) between GloVe representation and retrofitted (by DT network) GloVe representation. Clearly, DT retrofitting is not helping much to improve the performance of GloVe.

forth, we ignore the *D2V-L* model for the rest of our experiments.

Next, we investigate whether network embeddings applied on Distributional Thesaurus network can be combined with GloVe and Word2vec to improve the performance on the pre-specified tasks. In order to do that, we combine the vector representations using two operations: concatenation (CC), and principal component analysis (PCA). Table 2 represents the performance of combining GloVe with *D2V-D* and *D2V-N* for all the datasets using these combination strategies. In general, PCA turns out to be better technique for vector combination than CC. Clearly, combining DT embeddings and GloVe boosts the performance for all the datasets except for the **MEN-N** dataset, where the combined representation produces comparable performance.

In order to ensure that this observation is consistent, we try combining DT embeddings with Word2vec. The results are presented in Table 3 and we see very similar improvements in the performance except for a few cases, indicating the fact that combining word embeddings prepared form DT network is helpful in enhancing performances. From Tables 1, 2 and 3, we see that GloVe proves to be better than word2Vec for most of the cases, *D2V-N* is the best performing network embedding, and PCA turns out to be the best combination technique. Henceforth, we consider PCA (GloVe, *D2V-N*) as our model for comparison with the baselines for the rest of the experiments.

Further, to scrutinize that the achieved result is not just the effect of combining two different word vectors, we compare PCA (GloVe, D2V-N) against combination of GloVe and Word2vec (W2V). Table 4 shows the performance comparison on different datasets and it is evident that PCA (GloVe, D2V-N) gives better results compared to PCA (GloVe, W2V) in most of the cases.

Now, as we observe that the network embedding from DT network helps to boost the performance of Word2vec and GloVe when combined with them, we further compare the performance against the case when text based embeddings are combined with embeddings from lexical resources. For that purpose, we take **one baseline** (Goikoetxea et al., 2016), where authors combined the text based representation with WordNet based representation. Here we use GloVe as the text based representation and PCA as the combination method as prescribed by the author. Note that, WordNet based representation is made publicly available by Goikoetxea et al. (2016). From the second and third columns of Table 5, we observe that even though we do not use any manually created lexical resources like WordNet our approach achieves comparable performance. Additionally we check whether we gain in terms of performance if we integrate the three embeddings together. Fourth column of Table 5 shows that we gain for some of the datasets and for other cases, it has a negative effect. Looking at the performance, we can conclude that automatically generated DT network from corpus brings in useful additional information as far as word similarity and relatedness tasks are concerned.

So far, we use concatenation and PCA as methods for combining two different representations. However, as per the literature, there are different ways of infusing knowledge from different lexical sources to improve the quality of pre-trained vector embeddings. So we compare our proposed way of combination with a completely different way of integrating information from both dimensions, known as *retrofitting*. Retrofitting is a novel way proposed by Faruqui et al. (2015) for refining vector space representations using relational information from semantic lexicons by encouraging linked words to have similar vector representations. Here instead of using semantic lexicons, we use the DT network to produce the linked words to have similar vector representation. Note that, for a target word, we consider only those words as linked words which are having edge weight greater than a certain threshold. While experimenting with various thresholds, the best results were obtained for a

threshold value of 500. Table 6 shows the performance of GloVe representations when retrofitted with information from DT network. Even though in very few cases it gives little improved performance, compared to other combinations presented in Table2, the correlation is not very good, indicating the fact that retrofitting is probably not the best way of fusing knowledge from a DT network.

Further, we extend our study to investigate the usefulness of DT embedding on other NLP tasks like synonym detection, SAT analogy task as will be discussed next.

4.3 Synonym Detection

We consider two gold standard datasets for the experiment of synonym detection. The descriptions of the used datasets are given below.

TOEFL: It contains 80 multiple-choice synonym questions (4 choices per question) introduced by Landauer and Dumais (1997), as a way of evaluating algorithms for measuring degree of similarity between words. Being consistent with the previous experiments, we consider only nouns for our experiment and prepare **TOEFL-N** which contains 23 synonym questions.

ESL: It contains 50 multiple-choice synonym questions (4 choices per question), along with a sentence for providing context for each of the question, introduced by Turney (2001). Here also we consider only nouns for our experiment and prepare **ESL-N** which contains 22 synonym questions. Note that, in our experimental setup we do not use the context per question provided in the dataset for evaluation.

While preparing both the datasets, we also keep in mind the availability of word vectors in both downloaded GloVe representation and prepared DT embedding. For evaluation of the word embeddings using **TOEFL-N** and **ESL-N**, we consider the option as the correct answer which is having highest cosine similarity with the question and report accuracy. From the results presented in Table 7, we see that DT embedding leads to boost the performance of GloVe representation.

4.4 Analogy Detection

For analogy detection we experiment with **SAT** analogy dataset. This dataset contains 374 multiple-choice analogy questions (5 choices per question) introduced by Turney and Bigham (2003) as a way of evaluating algorithms for measuring relational similarity. Considering only noun questions, we prepare **SAT-N**, which contains 159 questions.

Dataset	GloVe	D2V-N	PCA (GloVe, D2V-N)
TOEFL-N	0.826	0.739	**0.869**
ESL-N	0.636	0.591	**0.682**
SAT-N	0.465	0.509	**0.515**

Table 7: Comparison of accuracies between GloVe representation, DT embedding using node2vec and combination of both where PCA is the combination technique. Clearly DT embedding is helping to improve the performance of GloVe for synonym detection as well as analogy detection.

In order to find out the correct answer from the 5 options given for each question, we take up a score (s) metric proposed by Speer et al. (2017), where for a question 'a_1 is to b_1', we will consider 'a_2 is to b_2' as the correct answer among the options, whose score (s) is the highest. Score (s) is defined by the author as follows:

$$s = a_1.a_2 + b_1.b_2 + w_1(b_2 - a_2).(b_1 - a_1) + w_2(b_2 - b_1).(a_2 - a_1)$$

As mentioned by the authors, the appropriate values of w_1 and w_2 are optimized separately for each system using grid search, to achieve the best performance. We use accuracy as the evaluation metric. The last row of Table 7 presents the comparison of accuracies (best for each model) obtained using different embeddings portraying the same observation that combination of GloVe and DT embeddings leads to better performance compared to GloVe and DT embeddings when used separately. Note that, the optimized values of (w_1, w_2) are (0.2,0.2), (0.8,0.6), (6,0.6) for GloVe, DT embedding, combined representation of GloVe and DT embeddings, respectively, for the analogy task.

5 Conclusion

In this paper we showed that both dense count based model (GloVe) and predictive model (Word2vec) lead to improved word representation when they are combined with word representation learned using network embedding methods on Distributional Thesaurus (DT) network. We tried with various network embedding models among which *node2vec* proved to be the best in our experimental setup. We also tried with different methodologies to combine vector representations and PCA turned out to be the best among them. The combined vector representation of words yielded the better performance for most

of the similarity and relatedness datasets as compared to the performance of GloVe and Word2vec representation individually. Further we observed that we could use the information from DT as a proxy of WordNet in order to improve the state-of-the-art vector representation as we were getting comparable performances for most of the datasets. Similarly, for synonym detection task and analogy detection task, the same trend of combined vector representation continued, showing the superiority of the combined representation over state-of-the-art embeddings. All the datasets used in our experiments which are not under any copyright protection, along with the DT embeddings are made publicly available[2].

In future we plan to investigate the effectiveness of the joint representation on other NLP tasks like text classification, sentence completion challenge, evaluation of common sense stories etc. The overall aim is to prepare a better generalized representation of words which can be used across languages in different NLP tasks.

References

Clémentine Adam, Cécile Fabre, and Philippe Muller. 2013. Évaluer et améliorer une ressource distributionnelle. *Traitement Automatique des Langues* 54(1):71–97.

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 19–27.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *ACL (1)*. pages 238–247.

Chris Biemann and Martin Riedl. 2013. Text: Now in 2d! a framework for lexical expansion with contextual similarity. *Journal of Language Modelling* 1(1):55–95.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. AcM, pages 1247–1250.

[2] https://tinyurl.com/yamrutrm

Fabian Both, Steffen Thoma, and Achim Rettinger. 2017. Cross-modal knowledge transfer: Improving the word embedding of apple by looking at oranges. In *K-CAP2017, The 9th International Conference on Knowledge Capture*. International Conference on Knowledge Capture, ACM.

Elia Bruni, Nam-Khanh Tran, and Marco Baroni. 2014. Multimodal distributional semantics. *J. Artif. Intell. Res.(JAIR)* 49(2014):1–47.

Alexander Budanitsky and Graeme Hirst. 2006. Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics* 32(1):13–47.

James R Curran and Marc Moens. 2002. Improvements in automatic thesaurus extraction. In *Proceedings of the ACL-02 workshop on Unsupervised lexical acquisition-Volume 9*. Association for Computational Linguistics, pages 59–66.

Manaal Faruqui, Jesse Dodge, Sujay K. Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of NAACL*.

Manaal Faruqui and Chris Dyer. 2014. Improving vector space word representations using multilingual correlation. Association for Computational Linguistics.

Olivier Ferret. 2017. Turning distributional thesauri into word vectors for synonym extraction and expansion. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. volume 1, pages 273–283.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*. ACM, pages 406–414.

John R Firth. 1957. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis* .

Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Tomas Mikolov, et al. 2013. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*. pages 2121–2129.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. Ppdb: The paraphrase database. In *HLT-NAACL*. pages 758–764.

Josu Goikoetxea, Eneko Agirre, and Aitor Soroa. 2016. Single or multiple? combining word representations independently learned from text and wordnet. In *AAAI*. pages 2608–2614.

Josu Goikoetxea, Aitor Soroa, Eneko Agirre, and Basque Country Donostia. 2015. Random walks and neural network language models on knowledge bases. In *HLT-NAACL*. pages 1434–1439.

Yoav Goldberg and Jon Orwant. 2013. A dataset of syntactic-ngrams over time from a very large corpus of english books. In *Second Joint Conference on Lexical and Computational Semantics (* SEM)*. volume 1, pages 241–247.

Gregory Grefenstette. 2012. *Explorations in automatic thesaurus discovery*, volume 278. Springer Science & Business Media.

Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pages 855–864.

Guy Halawi, Gideon Dror, Evgeniy Gabrilovich, and Yehuda Koren. 2012. Large-scale learning of word relatedness with constraints. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pages 1406–1414.

Per Christian Hansen. 1987. The truncatedsvd as a method for regularization. *BIT Numerical Mathematics* 27(4):534–553.

Felix Hill, Roi Reichart, and Anna Korhonen. 2016. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics* .

Ian T Jolliffe. 1986. Principal component analysis and factor analysis. In *Principal component analysis*, Springer, pages 115–128.

Douwe Kiela and Léon Bottou. 2014. Learning image embeddings using convolutional neural networks for improved multi-modal semantics. In *EMNLP*. pages 36–45.

Adam Kilgarriff, Pavel Rychly, Pavel Smrz, and David Tugwell. 2004. Itri-04-08 the sketch engine. *Information Technology* 105:116.

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(2):211.

Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*. pages 2177–2185.

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.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*. Association for Computational Linguistics, pages 768–774.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* .

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM* 38(11):39–41.

George A Miller and Walter G Charles. 1991. Contextual correlates of semantic similarity. *Language and cognitive processes* 6(1):1–28.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. pages 1532–1543.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, KDD '14, pages 701–710. https://doi.org/10.1145/2623330.2623732.

Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul Markovitch. 2011. A word at a time: computing word relatedness using temporal semantic analysis. In *Proceedings of the 20th international conference on World wide web*. ACM, pages 337–346.

Pushpendre Rastogi, Benjamin Van Durme, and Raman Arora. 2015. Multiview lsa: Representation learning via generalized cca. In *HLT-NAACL*. pages 556–566.

Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. 2017. struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, pages 385–394.

Martin Riedl and Chris Biemann. 2013. Scaling to large3 data: An efficient and effective method to compute distributional thesauri. In *EMNLP*. pages 884–890.

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

Robert Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*. pages 4444–4451.

Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, pages 1067–1077.

Steffen Thoma, Achim Rettinger, and Fabian Both. 2017. Knowledge fusion via embeddings from text, knowledge graphs, and images. *arXiv preprint arXiv:1704.06084* .

Fei Tian, Bin Gao, En-Hong Chen, and Tie-Yan Liu. 2016. Learning better word embedding by asymmetric low-rank projection of knowledge graph. *Journal of Computer Science and Technology* 31(3):624–634. https://doi.org/10.1007/s11390-016-1651-5.

Peter Turney. 2001. Mining the web for synonyms: Pmi-ir versus lsa on toefl. *Machine Learning: ECML 2001* pages 491–502.

Peter D Turney and Jeffrey Bigham. 2003. Combining independent modules to solve multiple-choice synonym and analogy. In *Problems. Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP-03*. Citeseer.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph and text jointly embedding. In *EMNLP*. volume 14, pages 1591–1601.