Regularized and Retrofitted models for Learning Sentence Representation with Context

Tanay Kumar Saha
Indiana University Purdue University Indianapolis
750 W. Michigan St
Indianapolis, IN 46202
tksaha@iupui.edu

Shafiq Joty
Nanyang Technological University
50 Nanyang Ave
Singapore 639798
srjoty@ntu.edu.sg

Naeemul Hassan
University of Mississippi
Oxford, Mississippi 38677
nhassan@olemiss.edu

Mohammad Al Hasan
Indiana University Purdue University Indianapolis
750 W. Michigan St
Indianapolis, IN 46202
alhasan@iupui.com

ABSTRACT

Vector representation of sentences is important for many text processing tasks that involve classifying, clustering, or ranking sentences. For solving these tasks, bag-of-word based representation has been used for a long time. In recent years, distributed representation of sentences learned by neural models from unlabeled data has been shown to outperform traditional bag-of-words representations. However, most existing methods belonging to the neural models consider only the content of a sentence, and disregard its relations with other sentences in the context. In this paper, we first characterize two types of contexts depending on their scope and utility. We then propose two approaches to incorporate contextual information into content-based models. We evaluate our sentence representation models in a setup, where context is available to infer sentence vectors. Experimental results demonstrate that our proposed models outshine existing models on three fundamental tasks, such as, classifying, clustering, and ranking sentences.

CCS CONCEPTS

•Computing methodologies → Learning latent representations; •Information systems → Clustering and classification; Summarization;

KEYWORDS

Sen2Vec; distributed representation of sentences; feature learning; discourse; retrofitting; classification; ranking; clustering

1 INTRODUCTION AND MOTIVATION

Many sentence-level text processing tasks rely on representing the sentences using fixed-length vectors. For example, *classifying* sentences into topics using a statistical classifier like Maximum Entropy requires the sentences to be represented by vectors. Similarly, for the task of *ranking* sentences based on their importance in the text using a ranking model like LexRank [5] or SVMRank [10], one needs to first represent the sentences with fixed-length vectors. The most common approach uses a bag-of-words or a bag-of-ngrams representation, where each dimension of the vector is computed by some form of term frequency statistics (e.g., tf^*idf).

Recently, *distributed representations*, in the form of dense real-valued vectors, learned by neural network models from unlabeled data, has been shown to outperform traditional bag-of-words representation [14]. Distributed representations encode the semantics of linguistic units and yield better generalization [3, 19]. However, most existing methods to devise distributed representation for sentences consider only the content of a sentence, and disregard relations between sentences in a text by and large [8, 14]. But, sentences rarely stand on their own in a well-formed text. On a finer level, sentences are connected with each other by certain logical relations (e.g., *elaboration*, *contrast*) to express the meaning as a whole [9]. On a coarser level, sentences in a text address a common topic, often covering multiple subtopics; i.e., sentences are also topically related [31]. Our main hypothesis in this paper is that distributed representation methods for sentences should not only consider the content of the sentence but also the contextual information in the text.

Recent studies [6, 35, 37] on learning distributed representations for *words* have shown that semantic relations between words (e.g., synonymy, hypernymy, hyponymy) encoded in semantic lexicons like WordNet [21] or Framenet [2] can improve the quality of word vectors that are trained solely on unlabeled data [6, 35, 37]. Our work in this paper is reminiscent of this line of research with a couple of crucial differences. Firstly, we are interested in representation of sentences as opposed to words, for the former such resources are not readily available. Secondly, our main goal is to incorporate extra-sentential context in some form of inter-sentence relations as opposed to semantic relations between words. These differences posit a number of new research challenges: (*i*) how can we obtain extra-sentential context that can capture semantic relations between sentences? (*ii*) how can we effectively exploit the inter-sentence relations in our representation learning model? and

CIKM'17, November 6–10, 2017, Singapore.
 ISBN 978-1-4503-4918-5/17/11…$15.00
DOI: https://doi.org/10.1145/3132847.3133011

finally, (*iii*) how can we evaluate the quality of the vectors learned by our model?

To tackle the first issue, we explore two different methods to obtain extra-sentential context. In our first method, we consider the adjoining sentences of a sentence in a text as the context. We call this *discourse context* since it captures the actual order of the sentences. In our second method, we build a similarity network of sentences, and consider adjacent nodes (i.e., one-hop neighbors) of a sentence as its context. We call this *similarity context* since it is based on a direct measure of similarity. Our choice of network to encode context is due to the fact that networks provide flexible ways to represent relations between any pair of sentences [5, 17].

We address the second challenge by proposing two different approaches to exploit the context information. In our first approach, we first learn sentence vectors using an existing content-based model, Sen2Vec [14]. Then, we refine these vectors to encourage the new estimated vectors to be similar to the vectors of its neighbors and similar to their prior Sen2Vec representations. The refinement is performed by using an efficient iterative algorithm [6, 32]. We call this model *retrofitted* model since it retrofits the initially learned Sen2Vec vectors using contextual information. In our second approach, we alter the objective function of Sen2Vec with a regularizer or prior that encourages neighboring sentences to have similar vector representations. We call this *regularized* model. In this approach, the vectors are learned from scratch by jointly modeling the content of the sentences and the relation between sentences.

Several recent methods also exploit contextual information to learn sentence vectors, e.g., FastSent [8] and Skip-Thought [12]. These methods learn sentence representations by predicting *content* (words or word sequences) of adjoining sentences. By learning representations that can predict contents of adjacent sentences, these methods may learn semantic and syntactic properties that are more specific to the neighbors rather than the sentence under consideration. Furthermore, these methods either make simple BOW (bag of words) assumption or disregard context when extracting a sentence vector. By contrast, our models learn sentence representations directly, and they treat adjacent sentences as atomic linguistic units.

Different approaches to evaluate sentence representation methods have been proposed in the past including sentence-level prediction tasks (e.g., sentiment classification, paraphrase identification) and sentence-pair similarity computation task [8, 14]. These approaches evaluate sentences independently out of context. Instead, in this paper, we propose an evaluation setup, where extra-sentential context is available to infer sentence vectors. We evaluate our models on three different types of tasks: *classification, clustering* and *ranking*. In particular, we consider the tasks of classifying and clustering sentences into topics, and of ranking sentences in a document to create an extractive summary of the document (i.e., by selecting the top-ranked sentences). There are standard datasets with document-level topic annotations (e.g., *Reuters-21578*, *20 Newsgroups*). However, to our knowledge, no dataset exists with topic annotations at the sentence level. We generate sentence-level topic annotations from the document-level ones by selecting subsets of sentences that can be considered as representatives of the document and label them with the same document-level topic label. We use the standard DUC 2001 and 2002 datasets to evaluate our models on the summarization task, where we compare the system-generated summaries with the human-authored summaries.

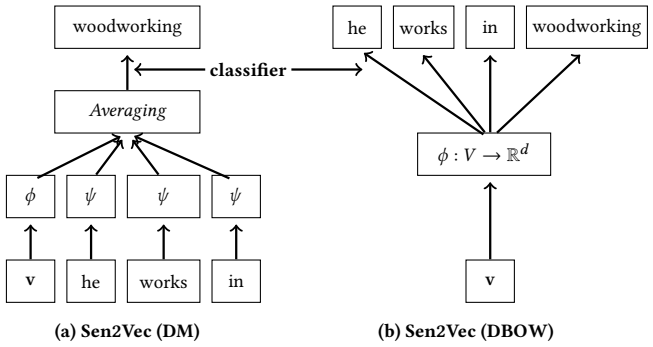


Figure 1: Distributed memory (DM) and Distributed bag of words (DBOW) versions of Sen2Vec.

Our evaluation on these tasks across multiple datasets shows impressive results for our model, which outperforms the best existing models by up to 6.29 F_1-score in classification, 12.78 V-score in clustering, 2.90 ROUGE-1 score in summarization. We found that the discourse context perform better on topic classification and clustering tasks, and similarity context performs better on summarization. We have implemented all our proposed models in a flexible software stack, which enables effective evaluation of existing or future sentence representation learning models. We make our code[1] publicly available.

The rest of the paper is organized as follows. In Section 2, we present a content-only model followed by two extensions of this model, which incorporate contextual information. In Sections 3, 4 and 5, we discuss experimental settings and results. Section 6 gives an account on related work, and finally, we conclude with a discussion of future work in Section 7.

2 METHODOLOGY

Let $\phi : V \rightarrow \mathbb{R}^d$ be the mapping function from sentences to their distributed representations, i.e., real-valued vectors of d dimensions. Equivalently, ϕ can be thought of as a look-up matrix of size $|V| \times d$, where $|V|$ is the total number of sentences. Our goal is to learn ϕ by exploiting information from two different sources: (*i*) the content of the sentence, $\mathbf{v} = (v_1, v_2 \cdots v_m)$; and (*ii*) the context of the sentence, $N(\mathbf{v})$. In the following subsections, we first describe an existing model that considers only the content of a sentence (Subsection 2.1). We then formalize types of extra-sentential context (Subsection 2.2). Finally, we present our models that extend the content-based model to incorporate contextual information (Subsections 2.3 – 2.4).

2.1 Content-based Model: Sen2Vec

Le and Mikolov [14] proposed two log-linear models for learning vector representation of sentences: (*a*) a distributed memory (DM) model, and (*b*) a distributed bag of words (DBOW) model. As shown in Figure 1, both models are trained solely based on the content of the sentences. In the DM model, every sentence in

[1] https://github.com/tksaha/con-s2v/tree/jointlearning

V is represented by a d dimensional vector in a shared lookup matrix $\phi \in \mathbb{R}^{|V|\times d}$. Similarly, every word in the vocabulary $\mathcal{D}$ is represented by a d dimensional vector in another shared lookup matrix $\psi \in \mathbb{R}^{|\mathcal{D}|\times d}$. Given an input sentence $\mathbf{v} = (v_1, v_2 \cdots v_m)$, the corresponding sentence vector from ϕ and the corresponding word vectors from ψ are averaged to predict the next word in a context. More formally, the DM model minimizes the following loss (negative log likelihood):

$$\begin{aligned}\mathcal{L}_c(\mathbf{v}) &= \sum_{t=k}^{m-k} -\log P(v_t|\mathbf{v}; v_{t-k+1}, \cdots, v_{t-1}) \\ &= \sum_{t=k}^{m-k} -\log \frac{\exp(\omega(v_t)^T \mathbf{z})}{\sum_{v_i\in\mathcal{D}} \exp(\omega(v_i)^T \mathbf{z})} \end{aligned} \quad (1)$$

where $\mathbf{z}$ is the average of $\phi(\mathbf{v}), \psi(v_{t-k+1}), \cdots, \psi(v_{t-1})$ *input* vectors, and $\omega(v_t)$ is the *output* vector representation of word v_t. The sentence vector $\phi(\mathbf{v})$ is shared across all (sliding window) contexts extracted from the same sentence, thus acts as a distributed memory. Instead of predicting the next word in the context, the DBOW model predicts the words in the context independently given the sentence id as input. More formally, DBOW minimizes the following loss:

$$\begin{aligned}\mathcal{L}_c(\mathbf{v}) &= \sum_{t=k}^{m-k} \sum_{j=t-k+1}^{t} -\log P(v_j|\mathbf{v}) \\ &= \sum_{t=k}^{m-k} \sum_{j=t-k+1}^{t} -\log \frac{\exp(\omega(v_j)^T \phi(\mathbf{v}))}{\sum_{v_i\in\mathcal{D}} \exp(\omega(v_i)^T \phi(\mathbf{v}))} \end{aligned} \quad (2)$$

Training of the models is typically performed using gradient-based online methods, such as stochastic gradient descend (SGD). Unfortunately, this could be impractically slow on large corpora due to summation over all vocabulary items $\mathcal{D}$ in the denominator (Equations 1 and 2), which needs to be performed for every training instance $(\mathbf{v}, v_j)$. To address this, Mikolov et. al [19] use *negative sampling*, which samples negative examples to approximate the summation term. For instance, for each training instance $(\mathbf{v}, v_j)$ in Equation 2, we add S negative examples $\{(\mathbf{v}, v_j^s)\}_{s=1}^S$ by sampling v_j^s from a known noise distribution μ (e.g., *unigram*, *uniform*). The log probability is then formulated as such to discriminate a *positive* instance v_j from a *negative* one v_j^s.

$$\log P(v_j|\mathbf{v}) = \log \sigma\left(\omega(v_j)^T \phi(\mathbf{v})\right) + \log \sum_{s=1}^{S} \mathbb{E}_{v_j^s\sim\mu}\, \sigma\left(-\omega(v_j^s)^T \phi(\mathbf{v})\right) \quad (3)$$

where σ is the sigmoid function defined as $\sigma(x) = 1/(1+e^{-x})$. The loss in Equation 3 can be optimized efficiently as S is a small number (5 – 10) compared to the vocabulary size $|\mathcal{D}|$ (26K – 139K).

Both DM and DBOW models attempt to capture the overall semantics of a sentence by looking at its content words. However, sentences in a well-formed text are rarely independent, rather the meaning of one sentence depends on the meaning of other sentences in its context. For instance, consider the sentences in Figure 2(a), which are taken from the *science.space* category of the *Newsgroups* dataset. Here, the paragraph is talking about *shuttle's reusability* for the missions in space. If we consider the sentences independently, it is very hard to understand the topic. Sentence, **u** is talking about shuttle, **v** is raising concern about its reusability and sentence **y** is elaborating on **v** to convey the concern more straightforwardly. When the sentences are considered together, it becomes easier to interpret. This suggests that representation learning models should also consider extra-sentential context to learn better representations for sentences.

2.2 Context Types

We distinguish between two types of context: discourse context and similarity context, as we elaborate on them below.

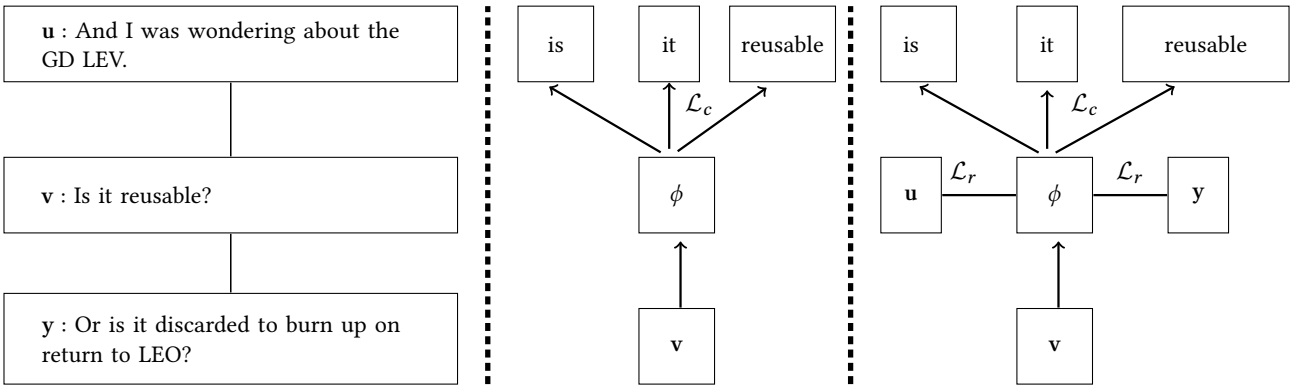
Discourse Context: Sentences in a text segment (e.g., paragraph) are semantically related by certain coherence relations (e.g., *elaboration*, *contrast*), and they address a common topic [31]. This indicates that adjacent sentences of a particular sentence is essential to better understand the meaning of the sentence. The discourse context of a sentence is comprised by its previous and the following sentences in a text.

Similarity Context: While the sequential order of the sentences carries important information, sentences that are far apart in the temporal order can also be related. In an empirical evaluation of data structures for representing discourse coherence, [34] advocates for a graph representation of discourse allowing non-adjacent connections. Moreover, graph-based methods for topic segmentation [17] and summarization [5] rely on complete graphs of sentences, where edge weights represent cosine similarity between sentences. Therefore, we consider a context type that is based on a direct measure of similarity, and considers relations between all possible sentences in a document and possibly across multiple documents.

Our similarity context allows any other sentence in the corpus to be in the context of a sentence depending on how similar they are. To measure the similarity, we first represent the sentences with vectors learned by Sen2Vec [14], then we measure the cosine distance between the vectors. We restrict the context size of a sentence for computational efficiency, while still ensuring that it is informative enough. We achieve this by imposing two kinds of constraints. First, we set thresholds for intra- and across-document connections: sentences in a document are connected only if their similarity value is above a pre-specified threshold δ, and sentences across documents are connected only if their similarity value is above another pre-specified threshold γ. Second, we allow up to 20 most similar neighbors. We call the resulting network *similarity network*. Equation 4 formalizes the similarity network construction strategy explained above.

$$(\mathbf{u}, \mathbf{v}) = \begin{cases} 1,\ if\ \sigma(\mathbf{u},\mathbf{v}) \le \delta \mid \mathbf{u}\in D_\ell, \mathbf{v}\in D_m, \ell = m, \mathbf{v}\in top_{20} \\ 1,\ if\ \sigma(\mathbf{u},\mathbf{v}) \le \gamma \mid \mathbf{u}\in D_\ell, \mathbf{v}\in D_m, \ell \ne m, \mathbf{v}\in top_{20} \\ 0,\ otherwise \end{cases} \quad (4)$$

where D_l and D_m refer to l-th and m-th documents in the corpus, respectively. In the following two subsections, we present two different methods to incorporate context (discourse or similarity) for learning vector representation of sentences.



(a) A Collection of Sentences **(b) Sen2Vec (DBOW)** **(c) Reg-dis**

Figure 2: (c) presents an instance of our regularized model for learning representation of sentence v in comparison to (b) Sen2Vec (DBOW) model within a context of two other sentences: u and y in (a). Directed and undirected edges indicate prediction loss and regularization loss, respectively. (Collected from:newsgroup/20news-bydate-train/sci.space/61019. The central topic is "science.space".)

Algorithm 1: Jacobi method for retrofitting.

Input :
- Graph $G = (V, E)$
- Prior vectors ϕ'
- Probabilities $\alpha_{\mathbf{v}}$ and $\beta_{\mathbf{v},\mathbf{u}}$

Output : Retrofitted vectors ϕ

$\phi \leftarrow \phi'$ `// initialization`

repeat
  for *all* $\mathbf{v} \in V$ **do**
    $\phi(\mathbf{v}) \leftarrow \frac{\alpha_{\mathbf{v}}\phi'(\mathbf{v}) + \sum_{\mathbf{u}} \beta_{\mathbf{v},\mathbf{u}}\phi(\mathbf{u})}{\alpha_{\mathbf{v}} + \sum_{\mathbf{u}} \beta_{\mathbf{v},\mathbf{u}}}$
  end
until *convergence*;

2.3 Retrofitted Models: Ret-dis, Ret-sim

We explore the general idea of *retrofitting* [6] to incorporate information from both the content and context of a node (sentence) in a joint learning framework. Let $\phi'(\mathbf{v})$ denote the vector representation for sentence $\mathbf{v}$ that has already been learned by our content-based model (Sen2Vec) in Section 2.1. Our aim is to retrofit this vector using either discourse context or similarity context such that the revised vector $\phi(\mathbf{v})$: (*i*) is similar to the prior vector $\phi'(\mathbf{v})$, and (*ii*) is also similar to the vectors of its adjoining sentences (discourse context) or its adjacent nodes (similarity context). To this end, we define the following objective function to minimize:

$$J(\phi) = \sum_{\mathbf{v} \in V} \alpha_v ||\phi(\mathbf{v}) - \phi'(\mathbf{v})||^2 + \sum_{(\mathbf{v},\mathbf{u}) \in E} \beta_{u,v} ||\phi(\mathbf{u}) - \phi(\mathbf{v})||^2 \quad (5)$$

where α values control the strength to which the algorithm should match the prior vectors, and β values control the degree of smoothness based on the graph similarity. The quadratic cost in Equation 5 is convex in ϕ, and has a closed form solution [32]. The closed form expression requires an inversion operation, which could be expensive for big graphs. A more efficient way is to use the Jacobi method, an online algorithm to solve the Equation iteratively. The Jacobi method leads to following update rule:

$$\phi(\mathbf{v}) \leftarrow \frac{\alpha_{\mathbf{v}}\phi'(\mathbf{v}) + \sum_{\mathbf{u}} \beta_{\mathbf{v},\mathbf{u}}\phi(\mathbf{u})}{\alpha_{\mathbf{v}} + \sum_{\mathbf{u}} \beta_{\mathbf{v},\mathbf{u}}} \quad (6)$$

In our case, we set $\alpha_{\mathbf{v}} = 1$, and $\beta_{\mathbf{v},\mathbf{u}} = \frac{1}{degree(\mathbf{v})}$, i.e. we give higher weights to vectors learned from Sen2Vec than to its contextual counterpart. Similar settings have been used in [6]. In Algorithm 1, we formally describe the training procedure of our retrofitted model. We use the DBOW model to learn the prior vectors ϕ'. We name the model that consider discourse context as Ret-dis and the model considering similarity context as Ret-sim.

2.4 Regularized Models: Reg-dis, Reg-sim

Rather than retrofitting the vectors learned from a content-based model using context as a post-processing step, we can incorporate neighborhood information directly into the objective function of the content-based model as a regularizer, and learn the sentence vectors in a single step. We define the following objective to minimize:

$$J(\phi) = \sum_{\mathbf{v} \in V} \Big[\mathcal{L}_c(\mathbf{v}) + \beta \mathcal{L}_r(\mathbf{v}, N(\mathbf{v})) \Big] \quad (7)$$

$$= \sum_{\mathbf{v} \in V} \Big[\mathcal{L}_c(\mathbf{v}) + \beta \sum_{(\mathbf{v},\mathbf{u}) \in E} ||\phi(\mathbf{u}) - \phi(\mathbf{v})||^2 \Big] \quad (8)$$

where the first component $\mathcal{L}_c(\mathbf{v})$ refers to the loss of the content-based model described in Section 2.1. The second component $\mathcal{L}_r(\mathbf{v}, N(\mathbf{v}))$ is a Laplacian regularizer with β being the regularization strength. The regularizer brings the vector representation of a sentence closer to its context. Depending on the context type, this leads to different objectives. The model that uses discourse context (call this Reg-dis) trains the vectors to be closer to the adjacent sentences in a text. Similarly, the model with similarity context (call this Reg-sim) trains the vectors to be closer to its neighbors in

Algorithm 2: Training REG with SGD

Input : set of sentences V, graph $G = (V, E)$, window size k
Output: learned sentence vectors ϕ

1. Initialize model parameters: ϕ, ψ and ω's;
2. Compute noise distribution: μ
3. **repeat**
   for *each sentence* $\mathbf{v} \in V$ **do**
      for *each content word* $v \in \mathbf{v}$ **do**
         `// for word vectors`
         for *each word* v_i *around* v *for window* k **do**
            (*a*) Generate a positive pair (v_i, v) and S negative pairs $\{(v_i, v^s)\}_{s=1}^{S}$ using μ;
            (*b*) Take a gradient step for $\mathcal{L}_c(v_i, v)$
         end
         `// for sentence vectors`
         (*c*) Generate a positive pair $(\mathbf{v}, v)$ and S negative pairs $\{(\mathbf{v}, v^s)\}_{s=1}^{S}$ using μ;
         (*d*) Take a gradient step for $\mathcal{L}_c(\mathbf{v}, v)$;
         (*e*) Take a gradient step for $\mathcal{L}_r(\mathbf{v}, N(\mathbf{v}))$;
      end
   end
until *convergence*;

the similarity network. As in the retrofitted models, we use DBOW as our content-based model.

Since the regularized models learn the vectors from scratch in one shot by considering information from both sources, the two components can be better adjusted to produce better quality vectors. Figure 2 (c) shows one instance of our model with discourse context (**u** and **y** are the adjoining sentences of sentence **v**). Algorithm 2 formally describes the training procedure for the regularized models. First, we initialize the model parameters: ϕ, ψ and ω, and compute the unigram distribution over words as our noise distribution μ. In each epoch of SGD (Line 3), we iterate over the sentences, and take one gradient step to learn word embeddings (Step b) and take two gradient steps (Steps d and e) to learn sentence embeddings: one for the content prediction loss, and the other for the regularization loss.

3 EVALUATION METHOD: TASKS, DATASETS AND METRICS

We evaluate our representation learning models on three different tasks that involve *classification*, *clustering*, and *ranking* sentences. These are the three fundamental information system tasks, and good performance over these tasks will indicate the robustness of our models in a wide range of downstream applications.

For classification (or clustering), we measure how effective the learned vectors are when they are used for classifying (or clustering) sentences based on their *topics*. Text categorization is now a standard task for evaluating cross-lingual *word* embeddings [7]. For ranking, we evaluate how effective the vectors are when they are used to rank sentences for generating an *extractive summary* [22] of a document.

Table 1: Basic Statistics of the DUC Datasets

Dataset	# Doc.	# Sen. (Avg)	# Sum. (Avg)
DUC 2001	486	40	2.17
DUC 2002	471	28	2.04

As our representation learning models exploit inter-sentence relations,[2] which can possibly be constrained by document boundaries (e.g., *similarity context*), therefore, for topic classification and clustering, we require datasets containing documents with sentence-level annotations. However, to the best of our knowledge, no dataset exists with topic annotations at the sentence level. We generate sentence-level topic annotations from the document-level ones by selecting subsets of sentences that can be considered as representatives of the document using an extractive summarization tool, and label the selected sentences with the same document-level topic. In both of our tasks, extractive summarization is a key component, therefore in the following, we first describe the summarization task.

3.1 Extractive Summarization (Ranking) Task

The Extractive Summarizer: Extractive summarization is often considered as a ranking problem with the goal to select the most important sentences to form a compressed version of the source document. Unsupervised methods are the predominant paradigm for determining sentence importance [22]. We use popular graph-based algorithm LexRank [5] for this purpose. To get the summary sentences of a document, we first build a weighted graph, where nodes represent the sentences of a document and edge weights represent cosine similarity between learned vector space representations (using any vector space representation models of our choice) of the two corresponding sentences. To make the graph sparse, we avoid edges with weight less than 0.10. We then run the PageRank algorithm [24] on the graph to determine the rank of each sentence in a document, and thereby *extract* the key sentences as summary of that document. The dumping factor in PageRank was set to 0.85.

Datasets: We use the benchmark datasets from DUC 2001 and 2002, where the task[3] is to generate a 100-words summary for each document in the datasets. Table 1 shows some basic statistics about the datasets. DUC-2001 and DUC-2002 has 486 and 471 documents respectively. The average number of sentences per document is 40 and 28, respectively. For each document, 2-3 short reference (human authored) summaries are available, which we use as gold summaries in our evaluation. The human authored summaries are of approximately 100 words. On average, the datasets have 2.17 and 2.04 human authored summaries per document, respectively. The sentence representations are learned independently a priori from the same source documents.

Metrics: We use the widely used automatic evaluation metric ROUGE [15] to evaluate the system-generated summaries. ROUGE is a recall oriented metric that computes *n*-gram recall between a

[2] For this reason, we did not evaluate our models on tasks previously used to evaluate sentence representation models.
[3] http://www-nlpir.nist.gov/projects/duc/guidelines

Table 2: Statistics about Reuters and Newsgroups.

Dataset	#Doc.	Total #sen.	Annot. #sen	Train #sen.	Test #sen.	#Class
Reuters	9,001	42,192	13,305	7,738	3,618	8
Newsgroups	7,781	95,809	22,374	10,594	9,075	8

candidate summary and a set of reference (human authored) summaries. Among the variants, ROUGE-1 (i.e., $n = 1$) has been shown to correlate well with human judgments for short summaries [15]. Therefore, we only report ROUGE-1 in this paper. The configuration for ROUGE in our case is: *-c* 99 *-2 -1 -r* 1000 *-w* 1.2 *-n* 4 *-m -s -a -l 100*. Depending on the task at hand, ROUGE collects the first 100 words from the summary after removing the stop words to compare with the corresponding reference summaries.

3.2 Topic Classification and Clustering Tasks

Classification and Clustering Tools: We train a maximum entropy (MaxEnt) classifier using the vectors learned from the models with no additional fine-tuning for evaluation. Following [13], we restrict ourselves to linear classifier. The two main reasons are: (*i*) it makes reproducing results of experiments straight-forward, and (*ii*) it allows us to better analyze the quality of the learned vector representation. For clustering, we use k-means++ [1] algorithm for producing the clusters given the vector representation from the models. One can use non-linear classifiers (e.g., neural networks) or spectral clustering algorithms [23, 33] to achieve additional performance gain, but it is not the goal of our paper.

Datasets: We use *20-Newsgroups* and *Reuters-21578* datasets for the classification and clustering tasks. These datasets are publicly available and widely used for text categorization tasks.

***20 Newsgroups*:** This dataset is a collection of approximately 20, 000 news documents[4]. The documents are organized into 20 different topics. Some of these topics are closely related (e.g., *talk.politics.guns* and *talk.politics.mideast*), while others are diverse in nature (e.g., *misc.forsale* and *soc.religion.christian*). We selected 8 diverse topics in our experiments from the 20 topics. The selected topics are: *talk.politics.mideast*, *comp.graphics*, *soc.religion.christian*, *rec.autos*, *sci.space*, *talk.politics.guns*, *rec.sport.baseball*, and *sci.med*.

***Reuters-21578*:** Reuters Newswire[5] has 21578 documents covering 672 topics. We use "ModApte" train-test split and selected documents only from the most 8 frequent topics. The selected topics are: *acq, crude, earn, grain, interest, money-fx, ship,* and *trade*.

Generating Sentence-level Topic Annotations: As discussed earlier, for our evaluation on topic classification and clustering tasks, we have to create topic annotations at the sentence-level from the document-level topic labels. One option is to assume that all the sentences from a document have the same topic label as the document. However, this naive assumption propagates a lot of noises. Although sentences in a document collectively address a common topic, not all sentences are directly linked to that topic, rather some of them play supporting roles. To minimize this noise, we use the extractive (unsupervised) summarizer described in Section 3.1 to select the top $P\%$ (in our case, $P = 20$) sentences as representatives of the document and label them with the same topic label as the document. We used Sen2Vec [14] representation to compute cosine similarity between two sentences in LexRank. Table 2 shows statistics of the resulting datasets. Note that the sentence vectors are learned independently from an entire dataset (Total #sen.), and the annotated part (Annot. #sen.) is used for topic classification and clustering evaluation.

Metrics: We use accuracy (Acc), Macro-averaged F1 measure (F1), and Cohen's Kappa (κ) as evaluation metrics for comparing the performance of various vector representation methods on topic classification task. For measuring topic clustering performance [27], we use V-measure (V), and adjusted mutual information (AMI) score. V-measure is the harmonic mean of the *homogeneity* and *completeness* score. The idea of homogeneity is that the topic distribution within each cluster should be skewed to a single topic. Completeness score determines whether all members of a given topic are assigned to the same cluster. On the other hand, AMI measures the agreement of two assignments, in our case the clustering and the topic distribution. It is normalized against chance. All these measures are bounded by $[0, 1]$. Higher score means a better clustering.

4 EXPERIMENTAL SETTINGS

In this section, we briefly discuss the models that we compare with and the settings (hyperparameters, training) for our models.

4.1 Models Compared

We compare our models against a non-distributed baseline and a number of existing distributed representation models.

Non-Distributed Baseline: We implement a **TF-IDF** model as our non-distributed baseline. The model encodes representation of a sentence as the count of a set of word-features weighted by tf-idf. We use all the words in the corpus as features.

Sen2Vec: We described Sen2Vec in details as a content-only model in Section 2.1. We use Mikolov's implementation[6] of Sen2Vec as it gave better results than gensim's [7] version when validated on the sentiment treebank [29]. Following the recommendation by [14], we concatenate the vectors learned by DM and DBOW models. The concatenated vectors gave improvements over individual ones on our tasks. The vector dimensions in DM and DBOW were fixed to 300, thus the concatenation yields vectors of 600 dimensions. For this model, we only tune the window size (k) hyper-parameter.

W2V-avg: Sen2Vec model learns word representation along with the sentence representation. To encode a sentence using W2V-avg, we perform an averaging operation on the vector representation (learned from Sen2Vec) of all the words in a particular sentence. For this model, we consider window size (k) as a tuning parameter.

C-PHRASE: C-PHRASE [26] learns vector representation of words. It extended the CBOW model [20] to consider the hierarchical nature of syntactic phrasing. As the implementation of this model is not publicly available, we use pretrained word vectors from author's webpage.[8] We first perform simple addition of word sequences of a sentence for obtaining vector representation of a sentence, and

[4] http://qwone.com/ jason/20Newsgroups/
[5] http://kdd.ics.uci.edu/databases/reuters21578/
[6] https://code.google.com/archive/p/word2vec/
[7] https://radimrehurek.com/gensim/
[8] http://clic.cimec.unitn.it/composes/cphrase-vectors.html

Table 3: Similarity Network Statistics

Dataset	# Nodes	# Edges	Avg. # Edges
20 Newsgroups	95809	1370149	14.03
Reuters-21578	42192	471163	11.17
DUC 2001	19549	321423	20.15
DUC 2002	13129	216492	16.49

Table 4: Optimal values of the hyper-parameters for different models on different tasks.

Dataset	Task	Sen2Vec	FastSent *(win. size)*	W2V-avg	Reg-sim *(win. size, reg. str.)*	Reg-dis
Reuters	clas.	8	10	10	(8, 1.0)	(8, 1.0)
	clus.	12	8	12	(12, 0.3)	(12, 1.0)
Newsgroups	clas.	10	8	10	(10, 1.0)	(10, 1.0)
	clus.	12	12	12	(12, 1.0)	(12, 1.0)
DUC 2001	rank.	10	12	12	(10, 0.8)	(10, 0.5)
DUC 2002	rank.	8	8	10	(8, 0.8)	(8, 0.3)

then normalize the vector. Normalized vectors performed better on our tasks than the ones obtained through simple addition. The latent dimension of the pretrained word vectors is 300.

FastSent: FastSent [8] is an additive model that learns representation of words in a sentence by predicting words of adjacent sentences. We use the autoencode version of the model, which also predicts the words of the current sentence. In FastSent, a sentence vector is obtained by adding the word vectors. We run the model on our corpus to learn sentence representations of 600 dimensions, and tune the window size (k) hyperparameter on the dev. set.

Skip-Thought: Skip-Thought [13] uses an encoder-decoder approach to reconstruct adjacent sentences of an input sentence. Training Skip-Thought is computationally expensive [8], and it requires a lot of data to learn an effective model. We use the pretrained combine-skip model[9], which was trained on the book corpus [38] along with vocabulary expansion. Skip-thought vectors are of 4800 dimensions.

4.2 Hyper-Parameter Tuning and Training Details

All of our models except the retrofitted ones (i.e., Ret-sim, Ret-dis) are trained with stochastic gradient descent (SGD), where the gradient is obtained via backpropagation. We used subsampling of frequent words in the classification layer as described in [19], which together with negative sampling give significant speed-ups in training. The number of noise samples (S) in negative sampling was 5. In all our models, the embeddings vectors (ϕ, ψ) were of 600 dimensions, which were initialized with random numbers sampled from a small uniform distribution, $\mathcal{U}(-0.5/d, 0.5/d)$. The weight vectors ω's were initialized with zero. Increasing the dimension may increase performance, however, it also increases the complexity of the model. So, we keep it 600, which is a reasonable size [8]. For Ret-sim, and Ret-dis, the number of iteration was set to 20 following [6]. For the similarity context, the intra- and across-document thresholds δ and γ were set to 0.5 and 0.8, respectively. Table 3 shows the basic statistics of the resultant similarity network for all of our datasets.

For each dataset, we randomly selected 20% documents from the whole set to form a held-out validation set on which we tune the hyperparameters of the models. To find the best parameter values, we optimize F1 for classification, AMI for clustering and ROUGE-1 for summarization on the validation set. Window size (k) parameter for our model and the baselines were tuned over $\{8, 10, 12\}$ and the regularization strength parameter was tuned over $\{0.3, 0.6, 0.8, 1.0\}$. Table 4 shows the optimal values of each hyper-parameter for the four datasets. We evaluate our models on the test set with these optimal values, run each test experiment five times and take the average to avoid any random behavior appearing in the results. We observed the standard deviation to be quite low.

5 RESULTS AND DISCUSSION

We present our results on topic classification and clustering in Table 5, and results on ranking (summarization) in Table 6. The results in each table are shown in four groups. Sen2Vec belongs to the first group. The second group contains other existing models described in Section 4.1. The third group contains our *retrofitted* models with discourse context (Ret-dis) and similarity context (Ret-sim). Finally, the fourth group contains our *regularized* models, again considering discourse (Reg-dis) and similarity (Reg-sim) contexts. We report absolute value of the performance metrics for Sen2Vec, and for other models, we present their scores relative to Sen2Vec. In the following, we highlight the key points of our results.

Skip-Thought and FastSent perform poorly on our tasks: Unexpectedly, FastSent and Skip-Thought perform quite poorly on our tasks. Skip-Thought, in particular, has the worst performance on topic classification and clustering tasks. The model gives small improvement over Sen2Vec on ranking task in one of the datasets (DUC'01). These results contradict the claim made by [13] that skip-thought vectors are generic. To investigate if the poor results are due to shift of domains (*book* vs. *news*), we also trained Skip-Thought on our training corpora with vector size 600 and vocabulary size $30K$. The performance was even worse. We hypothesize, this is due to our training set size, which may not be enough for the heavy model. Another reason could be that Skip-Thought does not perform any inference to extract the vector using a context – although the model was trained to generate neighboring sentences, context was ignored when the encoder was used to extract the sentence vector. Also, by learning representations to predict contents of adjacent sentences, the learned vectors might capture linguistic properties that are more specific to the neighbors than the current sentence. Similar justification holds for FastSent, which performed quite poorly in five out of six settings (Tasks + Datasets combinations). Furthermore, FastSent does not learn sentence representation directly, rather it adds word vectors to get sentence representations.

Existing distributed methods show promising results: Apart from Skip-Thought and FastSent, other existing distributed models show promising results. As Table 5 shows, Sen2Vec outperforms TF-IDF representation by a good margin on both classification and

[9]https://github.com/ryankiros/skip-thoughts

Table 5: Performance of our models on topic classification and clustering tasks in comparison to Sen2Vec.

	Topic Classification Results						Topic Clustering Results			
	Reuters			*Newsgroups*			*Reuters*		*Newsgroups*	
	F_1	Acc	κ	F_1	Acc	κ	V	AMI	V	AMI
Sen2Vec	83.25	83.91	79.37	79.38	79.47	76.16	42.74	40.00	35.30	34.74
TF-IDF	(−) 3.51	(−) 2.68	(−) 3.85	(−) 9.95	(−) 9.72	(−) 11.55	(−) 21.34	(−) 20.14	(−) 29.20	(−) 30.60
W2V-avg	(+) 2.06	(+) 1.91	(+) 2.51	(−) 0.42	(−) 0.44	(−) 0.50	(−) 11.96	(−) 10.18	(−) 17.90	(−) 18.50
C-PHRASE	(−) 2.33	(−) 2.01	(−) 2.78	(−) 2.49	(−) 2.38	(−) 2.86	(−) 11.94	(−) 10.80	(−) 1.70	(−) 1.44
FastSent	(−) 0.37	(−) 0.29	(−) 0.41	(−) 12.23	(−) 12.17	(−) 14.21	(−) 15.54	(−) 13.06	(−) 34.40	(−) 34.16
Skip-Thought	(−) 19.13	(−) 15.61	(−) 21.8	(−) 13.79	(−) 13.47	(−) 15.76	(−) 29.94	(−) 28.00	(−) 27.50	(−) 27.04
RET-sim	(+) 0.92	(+) 1.28	(+) 1.65	(+) 2.00	(+) 1.97	(+) 2.27	(+) 3.72	(+) 3.34	(+) 5.22	(+) 5.70
RET-dis	(+) 1.66	(+) 1.79	(+) 2.30	(+) 5.00	(+) 4.91	(+) 5.71	(+) 4.56	(+) 4.12	(+) 6.28	(+) 6.76
REG-sim	(+) **2.53**	(+) **2.53**	(+) **3.28**	(+) 3.31	(+) 3.29	(+) 3.81	(+) 4.76	(+) 4.40	(+) **12.78**	(+) 12.18
REG-dis	(+) 2.52	(+) 2.43	(+) 3.17	(+) **5.41**	(+) **5.34**	(+) **6.20**	(+) **7.40**	(+) **6.82**	(+) 12.54	(+) **12.44**

Table 6: ROUGE-1 scores of the models on DUC datasets in comparison to Sen2Vec.

	DUC'01	DUC'02
Sen2Vec	43.88	54.01
TF-IDF	(+) **4.83**	(+) 1.51
W2V-avg	(−) 0.62	(+) 1.44
C-PHRASE	(+) 2.52	(+) 1.68
FastSent	(−) 4.15	(−) 7.53
Skip-Thought	(+) 0.88	(−) 2.65
RET-sim	(−) 0.62	(+) 0.42
RET-dis	(+) 0.45	(−) 0.37
REG-sim	(+) 2.90	(+) **2.02**
REG-dis	(−) 1.92	(−) 8.77

clustering tasks – up to 11.6 points on classification, and up to 30.6 points on clustering. W2V-avg shows 2 points improvement over Sen2Vec in topic classification on Reuters. The performance of C-PHRASE and W2V-avg is close to Sen2Vec for classification, however, the models lag substantially behind on clustering. Overall, Sen2Vec appears to be the strongest baseline for these two tasks.

In the ranking task (Table 6), Sen2Vec gets ROUGE-1 scores of 43.88 and 54.01 on DUC'01 and DUC'02 datasets, respectively. C-PHRASE outshines other distributed models on this task, and provides 2.52 and 1.68 points improvements over Sen2Vec. W2V-avg shows mixed results in summarization; it performs better than Sen2Vec on one dataset and worse on the other. Surprisingly, TF-IDF becomes the best performer on DUC'01, and gives improvements of 4.15 points over Sen2Vec. Overall, the results indicate that TF-IDF is a strong baseline for the summarization task.

Regularized and Retrofitted models outperform Sen2Vec: The retrofitting and regularized models improve over Sen2Vec on both classification and clustering tasks, showing gains of up to 6.2 points on classification and up to 12.8 points on clustering. We observe similar patterns in ranking given that the model considers the right context (ignoring the mixed results for retrofitted models). The improvements in most cases are significant. This demonstrates that contextual information is beneficial for these tasks.

Regularized models are the best performer: Our regularized models (REG-sim, REG-dis) performs best in five out of six settings (Dataset + Task combination). From the results presented in Table 5, we observe that regularized models are the top-performer in topic classification and clustering tasks. For topic classification on Newsgroups, our model gives around 6 points improvement over Sen2Vec and 8 points over C-PHRASE in all the metrics (F_1, *Acc* and κ). The improvements are even larger for clustering – about 13 points over Sen2Vec and 15 points over C-PHRASE. Similarly, on Reuters dataset, REG-dis gives around 3 and 7 points improvements over Sen2Vec in topic classification and clustering tasks, respectively.

Regularized models also perform well on summarization task in Table 6 – best in DUC'02 and second best in DUC'01. Given that the existing models fail to beat the TF-IDF baseline on this task, our results are rather encouraging.

Regularization is better than retrofitting given the right context: From the third and fourth groups of results in Table 5, it is clear that REG-dis and REG-sim are better models than their retrofitted counterparts. REG-sim also outperforms RET-sim in ranking (Table 6) by 2 to 3 points. The good performance comes from the fact that regularized models consider contextual information during training rather than in the post-processing step. Thus, the model can better adjust contributions from different components (prediction vs. regularization) accordingly.

Discourse context is good for topic classification and clustering: Discourse context perform better than similarity context in most cases on classification and clustering tasks. From Table 5, we notice that, RET-dis outperform RET-sim by up to 3 points

in classification and by about 1 point in clustering. Reg-dis and Reg-sim perform similarly on Reuters dataset for classification and on Newsgroup dataset for clustering. However, Reg-dis outperform Reg-sim by a wide margin on Newsgroup dataset for classification and on Reuters dataset for clustering. The primary reason is that sentences appearing together in a discourse tend to address the same (sub)topic [31]. Discourse context is cheaper to obtain as it is readily available (consider only adjoining sentences). For obtaining similarity context, we need to obtain the similarity network as described in Section 2.2.

Similarity context is good for summarization: Similarity context is more suitable than discourse context for summarization – Reg-sim is the best performer in DUC'02 dataset and the second best in DUC'01 dataset. Similarity context is based on a direct measure of similarity, and consider relations beyond adjacency. From a context of topically similar sentences, our model learns representations that capture linguistic aspects related to information centrality.

Other comments: We also experimented with *Sequential Denoising Autoencoder* (SDAE) and *Sequential Autoencoder* (SAE) models proposed in [8]. However, they performed poorly on our tasks (thus not shown in the table). For example, SAE gave accuracies of around 40% on reuters and 18% on newsgroups. This is similar to what [8] observed. They propose to use pretrained word embeddings to improve the results. We did not achieve significant gains by using pretrained embeddings on our tasks.

6 RELATED WORK

Recently, learning distributed representation of words, phrases, and sentences has gained a lot of attention due to its applicability and superior performance over bag-of-words (BOW) features in a wide range of text processing tasks [6, 14, 25, 35–37]. These models can be categorized into two groups: (*i*) task-agnostic or unsupervised models, and (*ii*) task-specific or supervised models. Task-agnostic models learn general purpose representation from naturally occurring unlabeled training data, and can capture interesting linguistic properties [8, 11, 19]. On the other hand, task-specific models are trained to solve a particular task, e.g., sentiment analysis [30], machine translation [4], and parsing [28]. Our focus in this paper is on learning distributed representation of sentences from unlabeled data.

The Word2vec model [18] to learn distributed representation of words is very popular for text processing tasks. The model also scales well in practice due to its simple architecture. Sen2Vec [14] extended Word2vec [18] to learn the representation for sentences and documents. The model maps each sentence to an unique id and learns the representation for the sentence using the contexts of words in the sentence – either by predicting the whole context independently (DBOW), or by predicting a word in the context (DM) given the rest. In our work, we extend the DBOW model to incorporate inter-sentence relations in the form of a discourse context or a similarity context. We do this using a graph-smoothing regularizer in the original objective function, or by retrofitting the initial vectors with different types of context.

In [6, 35, 37], retrofitting and regularization methods have been explored to incorporate lexical semantic knowledge into word representation models. Our overall idea of using external information is reminiscent of these models with two key differences: (*i*) the semantic network (WordNet, FrameNet) is given in their case, whereas we construct the network using similarities between sentences (nodes); (*ii*) we also explore discourse context that incorporate knowledge from adjacent sentences.

Adjacent sentences have been used previously for modeling task-agnostic representation of sentences. For example, Hill et al. [8] proposed FastSent, which learns word representation of a sentence by predicting words of its adjacent sentences. It derives a sentence vector by summing up the word vectors. The auto-encode version of FastSent also predicts the words of the current sentence. FastSent is fundamentally different from our models as we consider nearby sentences as atomic units, and we encode the sentence vector directly.

Hill et al. [8] also proposed two other models, *Sequential Denoising Autoencoder* (SDAE) and *Sequential Autoencoder* (SAE). SDAE employs an encoder-decoder framework, similar to neural machine translation (NMT) [4], to denoise an original sentence (target) from its corrupted version (source). SAE uses the same NMT framework to reconstruct (decode) the same source sentence. Both SAE and SDAE compose sentence vectors sequentially, but they disregard context of the sentence.

Another context-sensitive model is Skip-Thought [12], which uses the NMT framework to predict adjacent sentences (target) given a sentence (source). Since the encoder and the decoder use recurrent layers to compose vectors sequentially, SDAE and Skip-Thought are very slow to train. Furthermore, by learning representations to predict content of neighboring sentences, these methods (FastSent and Skip-Thought) may learn linguistic properties that are more specific to the neighbors rather than the sentence under consideration.

In contrast, we encode a sentence directly by treating it as an atomic unit, and we predict the words to model its content. Similarly, our model incorporates contextual information by treating neighboring sentences as atomic units. This makes our model quite efficient to train and effective for many tasks as we have shown.

7 CONCLUSION AND FUTURE WORK

In this paper, we have proposed a set of novel models for learning vector representation of sentences that consider not only content of a sentence but also context of a sentence in the text. We have explored two different ways to incorporate contextual information: (*i*) by retrofitting the initial vectors learned from a content-based model using context, and (*ii*) by regularizing the content-based model with a graph smoothing factor. We have also introduced two types of context: (*i*) discourse context, and (*ii*) similarity context.

While existing evaluation methods ignore contexts, we created an evaluation setup that allows one to infer sentence vectors using contextual information. We evaluated our models on tasks involving classifying and clustering sentences into topics, and ranking sentences for extractive single-document summarization. Our results across multiple datasets show impressive gains over existing distributed models in all evaluation tasks. The discourse context

was found to be beneficial for topic classification and clustering, whereas the similarity context was beneficial for summarization.

In this study, we restrict the evaluation of our models on *topic classification and clustering* using automatically annotated dataset. We would like to explore further how our models perform compared to the existing compositional models [12, 30], where documents with sentence-level sentiment annotation exists. Existing datasets – IMDB [16] or Sentiment Treebank [30], are not suitable for our purpose because in IMDB, there is no sentence-label annotation, and in sentiment treebank, there is no contextual information. We plan to create a dataset through manual annotation in the future.

ACKNOWLEDGMENTS

This research is partially supported by Mohammad Hasan's NSF CAREER Award (IIS-1149851).

REFERENCES

[1] David Arthur and Sergei Vassilvitskii. 2007. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 1027–1035.

[2] Collin F Baker, Charles J Fillmore, and John B Lowe. 1998. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1*. 86–90.

[3] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A Neural Probabilistic Language Model. *J. Mach. Learn. Res.* 3 (March 2003). http://dl.acm.org/citation.cfm?id=944919.944966

[4] Kyunghyun Cho, Bart van Merriënboer, Çalar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1724–1734. http://www.aclweb.org/anthology/D14-1179

[5] Günes Erkan and Dragomir R. Radev. 2004. LexRank: Graph-based Lexical Centrality As Salience in Text Summarization. *J. Artif. Int. Res.* 22, 1 (Dec. 2004), 457–479.

[6] Manaal Faruqui, Jesse Dodge, Sujay K. Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting Word Vectors to Semantic Lexicons. In *Proc. of NAACL*.

[7] Karl Moritz Hermann and Phil Blunsom. 2014. Multilingual Models for Compositional Distributed Semantics. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Baltimore, Maryland, 58–68. http://www.aclweb.org/anthology/P14-1006

[8] Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning Distributed Representations of Sentences from Unlabelled Data. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California, 1367–1377.

[9] Jerry R Hobbs. 1979. Coherence and coreference. *Cognitive science* 3, 1 (1979), 67–90.

[10] Thorsten Joachims. 2006. Training Linear SVMs in Linear Time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '06)*. 217–226.

[11] Tom Kenter, Alexey Borisov, and Maarten de Rijke. 2016. Siamese CBOW: Optimizing Word Embeddings for Sentence Representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

[12] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. Skip-thought Vectors. In *Proceedings of the 28th International Conference on Neural Information Processing Systems (NIPS'15)*. MIT Press, Montreal, Canada, 3294–3302.

[13] Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems*. 3294–3302.

[14] Quoc V Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents.. In *ICML*, Vol. 14. 1188–1196.

[15] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, Vol. 8. Barcelona, Spain.

[16] Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 142–150.

[17] Igor Malioutov and Regina Barzilay. 2006. Minimum cut model for spoken lecture segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics (ACL-44)*. Association for Computational Linguistics, Sydney, Australia, 25–32.

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

[19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS'13)*. 3111–3119.

[20] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[21] George A Miller. 1995. WordNet: a lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[22] Ani Nenkova, Kathleen McKeown, and others. 2011. Automatic summarization. *Foundations and Trends® in Information Retrieval* 5, 2–3 (2011), 103–233.

[23] Andrew Y Ng, Michael I Jordan, Yair Weiss, and others. 2001. On spectral clustering: Analysis and an algorithm. In *NIPS*, Vol. 14. 849–856.

[24] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank citation ranking: bringing order to the web. (1999).

[25] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global Vectors for Word Representation.. In *EMNLP*, Vol. 14. 1532–1543.

[26] Nghia The Pham, Germán Kruszewski, Angeliki Lazaridou, and Marco Baroni. 2015. Jointly optimizing word representations for lexical and sentential tasks with the C-PHRASE model. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*. 971–981.

[27] Andrew Rosenberg and Julia Hirschberg. 2007. V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure.. In *EMNLP-CoNLL*, Vol. 7. 410–420.

[28] Richard Socher, John Bauer, Christopher D. Manning, and Ng Andrew Y. 2013. Parsing with Compositional Vector Grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Sofia, Bulgaria, 455–465. http://www.aclweb.org/anthology/P13-1045

[29] Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng, and Christopher D. Manning. 2011. Parsing Natural Scenes and Natural Language with Recursive Neural Networks.. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*. 129–136.

[30] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Stroudsburg, PA, 1631–1642.

[31] Manfred Stede. 2011. *Discourse processing*. Morgan & Claypool Publishers.

[32] Partha Pratim Talukdar and Koby Crammer. 2009. New Regularized Algorithms for Transductive Learning. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part II (ECML PKDD '09)*. Springer-Verlag, 442–457.

[33] Stijn Van Dongen. 2000. A cluster algorithm for graphs. *Report-Information systems* 10 (2000), 1–40.

[34] F. Wolf and E. Gibson. 2005. Representing Discourse Coherence: A Corpus-Based Study. *Computational Linguistics* 31 (June 2005), 249–288. Issue 2.

[35] 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 Conference on Information and Knowledge Management*. ACM, 1219–1228.

[36] Liner Yang, Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. 2017. Improving Word Representations with Document Labels. *IEEE/ACM Trans. Audio, Speech & Language Processing* 25, 4 (2017), 863–870.

[37] 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 (Volume 2: Short Papers)*. Baltimore, Maryland, 545–550.

[38] Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books. In *arXiv preprint arXiv:1506.06724*.