Extending Multi-Sense Word Embedding to Phrases and Sentences for Unsupervised Semantic Applications

Haw-Shiuan Chang, Amol Agrawal, Andrew McCallum CICS, University of Massachusetts Amherst
{hschang,amolagrawal,mccallum}@cs.umass.edu

Abstract

Most unsupervised NLP models represent each word with a single point or single region in semantic space, while the existing multi-sense word embeddings cannot represent longer word sequences like phrases or sentences. We propose a novel embedding method for a text sequence (a phrase or a sentence) where each sequence is represented by a distinct set of multi-mode codebook embeddings to capture different semantic facets of its meaning. The codebook embeddings can be viewed as the cluster centers which summarize the distribution of possibly co-occuring words in a pre-trained word embedding space. We introduce an end-to-end trainable neural model that directly predicts the set of cluster centers from the input text sequence during test time. Our experiments show that the per-sentence codebook embeddings significantly improve the performances in unsupervised sentence similarity and extractive summarization benchmarks. In phrase similarity experiments, we discover that the multi-facet embeddings provide an interpretable semantic representation but do not outperform the single-facet baseline.

1 Introduction

Collecting manually labeled data is an expensive and tedious process for new or low-resource NLP applications. Many of these applications require the text similarity measurement based on the text representation learned from the raw text without any supervision. Examples of the representation include word embedding like Word2Vec (Mikolov et al. 2013) or GloVe (Pennington, Socher, and Manning 2014), sentence embeddings like skip-thoughts (Kiros et al. 2015), contextualized word embedding like ELMo (Peters et al. 2018) and BERT (Devlin et al. 2019) without fine-tuning.

The existing work often represents a word sequence (e.g., a sentence or a phrase) as a single embedding. However, when squeezing all the information into a single embedding (e.g., by averaging the word embeddings or using CLS embedding in BERT), the representation might lose some important information of different facets in the sequence.

Inspired by the multi-sense word embeddings (Lau et al. 2012; Neelakantan et al. 2014; Athiwaratkun and Wilson 2017; Singh et al. 2020), we propose a multi-facet representation that characterizes a phrase or a sentence as a fixed number of embeddings, where each embedding is a clustering center of the words co-occurring with the input word sequence.

In this work, a facet refers to a mode of the co-occurring word distribution, which might be multimodal. For example, the multi-facet representation of *real property* is illustrated in Figure 1. Real property can be observed in legal documents where it usually means real estate, while real property can also mean a true characteristic in philosophic discussions. The previous unsupervised multi-sense embeddings discover those senses by clustering the observed neighboring words (e.g., *acquired*, *save*, and *tax*) and an important facet, a mode with high probability, could be represented by several close cluster centers. Notice that the approaches need to solve a distinct local clustering problem for each phrase in contrast with the topic modeling like LDA (Blei, Ng, and Jordan 2003), which clusters all the words in the corpus into a global set of topics.

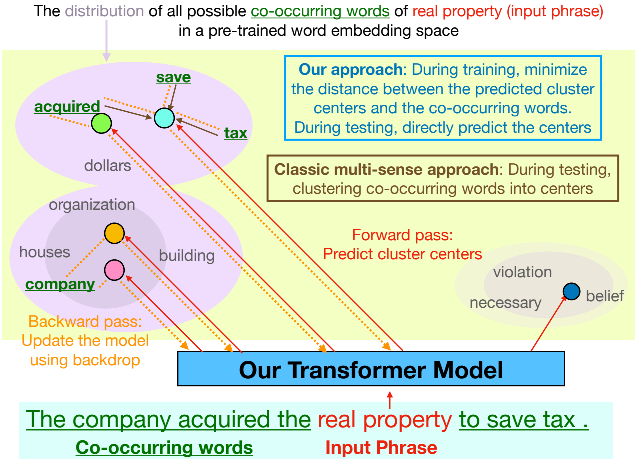
The distribution of all possible co-occurring words of real property (input phrase)
in a pre-trained word embedding space
save
acquired
tax
dollars
Our approach: During training, minimize the distance between the predicted cluster centers and the co-occurring words. During testing, directly predict the centers
Classic multi-sense approach: During testing, clustering co-occurring words into centers
organization
houses
building
company
Forward pass: Predict cluster centers
violation
necessary
belief
Backward pass: Update the model using backdrop
Our Transformer Model
The company acquired the real property to save tax .
Co-occurring words
Input Phrase

Figure 1: The input phrase *real property* is represented by $K = 5$ cluster centers. The previous work discovers the multiple senses by clustering the embedding of observed co-occurring words. Instead, our compositional model learns to predict the embeddings of cluster centers from the sequence of words in the input phrase so as to reconstruct the (unseen) co-occurring distribution well.

In addition to a phrase, we can also cluster the nearby words of a sentence which appears frequently in the corpus. The cluster centers usually correspond to important aspects rather than senses (see an example in Figure 2) because a sentence usually has multiple aspects but only one sense. However, extending the clustering-based multi-sense word embeddings to long sequences such as sentences is difficult in practice due to two efficiency challenges. First, there are usually many more unique phrases and sentences in a corpus than there are words, while the number of parameters for clustering-based approaches is $O(|V| \times |K| \times |E|)$, where $|V|$ is the number of unique sequences, $|K|$ is the number of clusters, and $|E|$ is the embedding dimensions. Estimating and storing such a large number of parameters takes time and space. More importantly, much more unique sequences imply much fewer co-occurring words to be clustered for each sequence, especially for sentences. An effective model needs to overcome this sample efficiency challenge (i.e., sparseness in the co-occurring statistics), but clustering approaches often have too many parameters to learn the compositional meaning of each sequence without overfitting.

Nevertheless, the sentences (or phrases) sharing multiple words often lead to similar cluster centers, so we should be able to solve these local clustering problems using much fewer parameters to circumvent the challenges. To achieve the goal, we develop a novel Transformer-based neural encoder and decoder. As shown in Figure 1, instead of clustering co-occurring words beside an input sequence at test time as in previous approaches, we learn a mapping between the input sequence (i.e., phrases or sentences) and the corresponding cluster centers during training so that we can directly predict those cluster centers using a single forward pass of the neural network for an arbitrary unseen input sequence during testing.

To train the neural model that predicts the clustering centers, we match the sequence of predicted cluster centers and the observed set of co-occurring word embeddings using a non-negative and sparse permutation matrix. After the permutation matrix is estimated for each input sequence, the gradients are back-propagated to cluster centers (i.e., codebook embeddings) and to the weights of our neural model, which allows us to train the whole model end-to-end.

In the experiments, we evaluate whether the proposed multi-facet embeddings could improve the similarity measurement between two sentences, between a sentence and a document (i.e., extractive summarization), and between phrases. The results demonstrate multi-facet embeddings significantly outperforms the classic single embedding baseline when the input sequence is a sentence.

We also demonstrate several advantages of the proposed multi-facet embeddings over the (contextualized) embeddings of all the words in a sequence. First, we discover that our model tends to use more embeddings to represent an important facet or important words. This tendency provides an unsupervised estimation of word importance, which improves various similarity measurements between a sentence pair. Second, our model outputs a fixed number of facets by compressing long sentences and extending short sentences. In unsupervised extractive summarization, this capability prevents the scoring function from biasing toward longer or shorter sentences. Finally, in the phrase similarity experiments, our methods capture the compositional meaning (e.g., a *hot dog* is a food) of a word sequence well and the quality of our similarity estimation is not sensitive to the choice of *K*, the number of our codebook embeddings.

1.1 Main Contributions

1. As shown in Figure 1, we propose a novel framework that predicts the cluster centers of co-occurring word embeddings to overcomes the sparsity challenges in our self-supervised training signals. This allows us to extend the idea of clustering-based multi-sense embeddings to phrases or sentences.
2. We propose a deep architecture that can effectively encode a sequence and decode a set of embeddings. We also propose non-negative sparse coding (NNSC) loss to train our neural encoder and decoder end-to-end.
3. We demonstrate how the multi-facet embeddings could be used in unsupervised ways to improve the similarity between sentences/phrases, infer word importance in a sentence, extract important sentences in a document. In Appendix B.1, we show that our model could provide asymmetric similarity measurement for hypernym detection.
4. We conduct comprehensive experiments in the main paper and appendix to show that multi-facet embedding is consistently better than classic single-facet embedding for modeling the co-occurring word distribution of sentences, while multi-facet phrase embeddings do not yield a clear advantage against the single embedding baseline, which supports the finding in Dubossarsky, Grossman, and Weinshall (2018).

2 Method

In this section, we first formalize our training setup and next describe our objective function and neural architecture. Our approach is visually summarized in Figure 2.

2.1 Self-supervision Signal

We express tth sequence of words in the corpus as $I_t = w_{x_t} ... w_{y_t}$<eos>, where x_t and y_t are the start and end position of the input sequence, respectively, and <eos> is the end of sequence symbol.

We assume neighboring words beside each input phrase or sentence are related to some facets of the sequence, so given I_t as input, our training signal is to reconstruct a set of co-occurring words, $N_t = \{w_{x_t-d_1^t}, ...w_{x_t-1}, w_{y_t+1}, ...w_{y_t+d_2^t}\}$.[1] In our experiments, we train our multi-facet sentence embeddings by setting N_t as the set of all words in the previous and the next sentence, and train multi-facet phrase embeddings by setting a fixed window size $d_1^t = d_2^t = 5$.

Since there are not many co-occurring words for a long sequence (none are observed for unseen testing sequences), the goal of our model is to predict the cluster centers of the

[1] The self-supervised signal is a generalization of the loss for prediction-based word embedding like Word2Vec (Mikolov et al. 2013). They are the same when the input sequence length $|I_t|$ is 1.

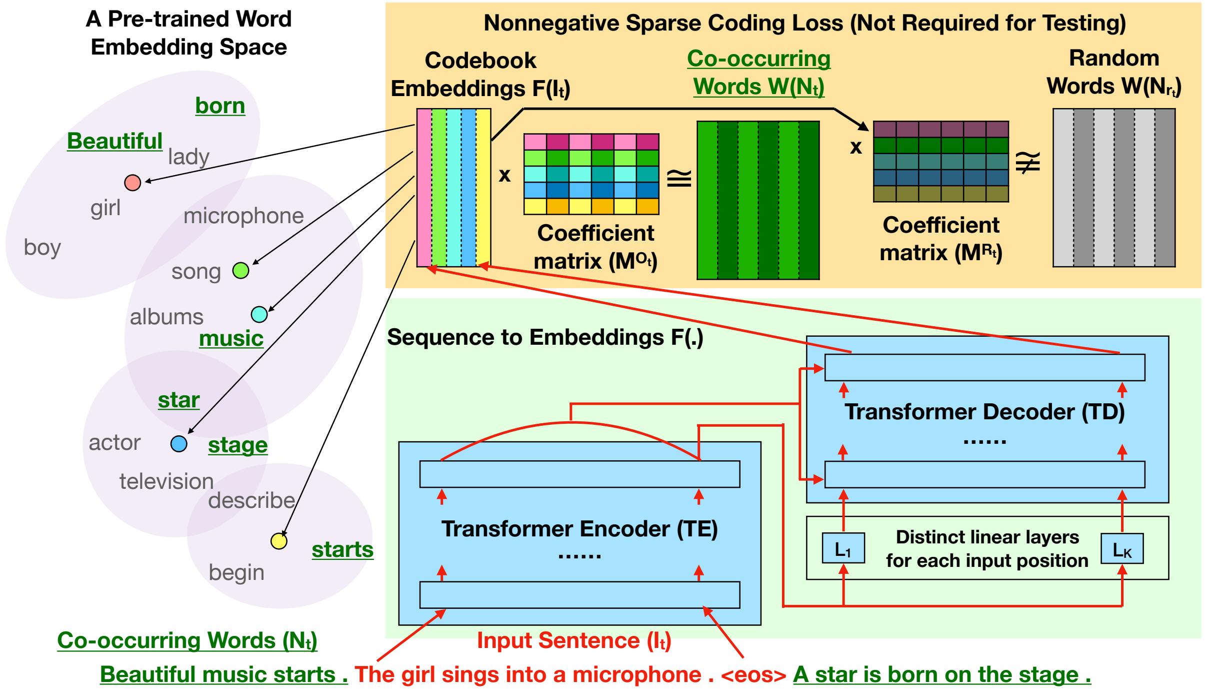
A Pre-trained Word Embedding Space
born
Beautiful
lady
girl
microphone
boy
song
albums
music
star
actor
stage
television
describe
starts
begin
Nonnegative Sparse Coding Loss (Not Required for Testing)
Codebook Embeddings F(I_t)
Co-occurring Words W(N_t)
Random Words W(N_r)
x
≅
x
≇
Coefficient matrix (M^{O_t})
Coefficient matrix (M^{R_t})
Sequence to Embeddings F(.)
Transformer Decoder (TD)
......
Transformer Encoder (TE)
......
L1
Distinct linear layers for each input position
LK
Co-occurring Words (N_t)
Input Sentence (I_t)
Beautiful music starts . The girl sings into a microphone . <eos> A star is born on the stage .

Figure 2: Our model for sentence representation. We represent each sentence as multiple codebook embeddings (i.e., cluster centers) predicted by our sequence to embeddings model. Our loss encourages the model to generate codebook embeddings whose linear combination can well reconstruct the embeddings of co-occurring words (e.g., *music*), while not able to reconstruct the negatively sampled words (i.e., the co-occurring words from other sentences).

words that could "possibly" occur beside the text sequence rather than the cluster centers of the actual occurring words in N_t (e.g., the hidden co-occurring distribution instead of green and underlined words in Figure 2). The cluster centers of an unseen testing sequence are predictable because the model could learn from similar sequences and their co-occurring words in the training corpus.

To focus on the semantics rather than syntax, we view the co-occurring words as a set rather than a sequence as in skip-thoughts (Kiros et al. 2015). Notice that our model considers the word order information in the input sequence I_t, but ignores the order of the co-occurring words N_t.

2.2 Non-negative Sparse Coding Loss

In a pre-trained word embedding space, we predict the cluster centers of the co-occurring word embeddings. The embeddings of co-occurring words N_t are arranged into a matrix $\boldsymbol{W}(\boldsymbol{N_t}) = [\underline{\boldsymbol{w}}_j^t]_{j=1...|N_t|}$ with size $|E| \times |N_t|$, where $|E|$ is the dimension of pre-trained word embedding, and each of its columns $\underline{\boldsymbol{w}}_j^t$ is a normalized word embedding whose 2-norm is 1. The normalization makes the cosine distance between two words become half of their squared Euclidean distance.

Similarly, we denote the predicted cluster centers $\underline{\boldsymbol{c}}_k^t$ of the input sequence I_t as a $|E| \times K$ matrix $\boldsymbol{F}(\boldsymbol{I_t}) = [\underline{\boldsymbol{c}}_k^t]_{k=1...K}$, where $\boldsymbol{F}$ is our neural network model and K is the number of clusters. We fix the number of clusters K to simplify the design of our prediction model and the unsupervised scoring functions used in the downstream tasks. When the number of modes in the (multimodal) co-occurring distribution is smaller than K, the model can output multiple cluster centers to represent a mode (e.g., the *music* facet in Figure 2 is represented by two close cluster centers). As a result, the performances in our downstream applications are not sensitive to the setting of K when K is larger than the number of facets in most input word sequences.

The reconstruction loss of k-means clustering in the word embedding space can be written as $||\boldsymbol{F}(\boldsymbol{I_t})\boldsymbol{M} - \boldsymbol{W}(\boldsymbol{N_t})||^2 = \sum_j ||(\sum_k \boldsymbol{M}_{k,j}\underline{\boldsymbol{c}}_k^t) - \underline{\boldsymbol{w}}_j^t||^2$, where $\boldsymbol{M}_{k,j} = 1$ if the jth word belongs to the k cluster and 0 otherwise. That is, $\boldsymbol{M}$ is a permutation matrix which matches the cluster centers and co-occurring words and allow the cluster centers to be predicted in an arbitrary order.

Non-negative sparse coding (NNSC) (Hoyer 2002) relaxes the constraints by allowing the coefficient $\boldsymbol{M}_{k,j}$ to be a positive value but encouraging it to be 0. We adopt NNSC in this work because we observe that the neural network trained by NNSC loss generates more diverse topics than k-means loss does. We hypothesize that it is because the loss is smoother and easier to be optimized for a neural network. Using NNSC, we define our reconstruction error as

$$\begin{aligned}
&Er(\boldsymbol{F}(\boldsymbol{I_t}), \boldsymbol{W}(\boldsymbol{N_t})) = ||\boldsymbol{F}(\boldsymbol{I_t})\boldsymbol{M}^{\boldsymbol{O_t}} - \boldsymbol{W}(\boldsymbol{N_t})||^2 \\
&s.t., \boldsymbol{M}^{\boldsymbol{O_t}} = \arg\min_{\boldsymbol{M}} ||\boldsymbol{F}(\boldsymbol{I_t})\boldsymbol{M} - \boldsymbol{W}(\boldsymbol{N_t})||^2 + \lambda||\boldsymbol{M}||_1, \\
&\forall k, j, \ 0 \leq \boldsymbol{M}_{k,j} \leq 1, \qquad (1)
\end{aligned}$$

where λ is a hyper-parameter controlling the sparsity of $\boldsymbol{M}$. We force the coefficient value $\boldsymbol{M}_{k,j} \leq 1$ to avoid the neural network learning to predict centers with small magnitudes which makes the optimal values of $\boldsymbol{M}_{k,j}$ large and unstable.

We adopt an alternating optimization strategy similar to the EM algorithm for k-means. At each iteration, our E-step estimates the permutation coefficient $\boldsymbol{M^{O_t}}$ after fixing our neural model, while our M-step treats $\boldsymbol{M^{O_t}}$ as constants to back-propagate the gradients of NNSC loss to our neural network. A pseudo-code of our training procedure could be found in Algorithm 1 in the appendix. Estimating the permutation between the prediction and ground truth words is often computationally expensive (Qin et al. 2019). Nevertheless, optimizing the proposed loss is efficient because for each training sequence I_t, $\boldsymbol{M^{O_t}}$ can be efficiently estimated using convex optimization (our implementation uses RMSprop (Tieleman and Hinton 2012)). Besides, we minimize the L2 distance, $||\boldsymbol{F(I_t)}\boldsymbol{M^{O_t}} - \boldsymbol{W(N_t)}||^2$, in a pre-trained embedding space as in Kumar and Tsvetkov (2019); Li et al. (2019) rather than computing softmax.

To prevent the neural network from predicting the same global topics regardless of the input, our loss function for tth sequence is defined as

$$L_t(\boldsymbol{F}) = Er(\boldsymbol{F(I_t)}, \boldsymbol{W(N_t)}) - Er(\boldsymbol{F(I_t)}, \boldsymbol{W(N_{r_t})}), \quad (2)$$

where N_{r_t} is a set of co-occurring words of a randomly sampled sequence I_{r_t}. In our experiment, we use SGD to solve $\widehat{\boldsymbol{F}} = \arg\min_{\boldsymbol{F}} \sum_t L_t(\boldsymbol{F})$. Our method could be viewed as a generalization of Word2Vec (Mikolov et al. 2013) that can encode the compositional meaning of the words and decode multiple embeddings.

2.3 Sequence to Embeddings

Our neural network architecture is similar to Transformer-based sequence to sequence (seq2seq) model (Vaswani et al. 2017). We use the same encoder $TE(I_t)$, which transforms the input sequence into a contextualized embeddings

$$[\underline{\boldsymbol{e}}_{x_t} ... \underline{\boldsymbol{e}}_{y_t} \underline{\boldsymbol{e}}_{\text{<eos>}}] = TE(w_{x_t} ... w_{y_t} \text{<eos>}), \quad (3)$$

where the goal of the encoder is to map the similar sentences, which are likely to have similar co-occurring word distribution, to similar contextualized embeddings.

Different from the typical seq2seq model (Sutskever, Vinyals, and Le 2014; Vaswani et al. 2017), our decoder does not need to make discrete decisions because our outputs are a sequence of embeddings instead of words. This allows us to predict all the codebook embeddings in a single forward pass as in Lee et al. (2019) without requiring an expensive softmax layer or auto-regressive decoding.[2]

To make different codebook embeddings capture different facets, we pass the embeddings of <eos>, $\underline{\boldsymbol{e}}_{\text{<eos>}}$, to different linear layers L_k before becoming the input of the decoder TD. The decoder allows the input embeddings to attend each other to model the dependency among the facets and attend the contextualized word embeddings from the encoder, $\underline{\boldsymbol{e}}_{x_t} ... \underline{\boldsymbol{e}}_{y_t} \underline{\boldsymbol{e}}_{\text{<eos>}}$, to copy the embeddings of some keywords in the word sequence as our facet embeddings more easily. Specifically, the codebook embeddings

$$\boldsymbol{F(I_t)} = TD(L_1(\underline{\boldsymbol{e}}_{\text{<eos>}}) ... L_K(\underline{\boldsymbol{e}}_{\text{<eos>}}), \underline{\boldsymbol{e}}_{x_t} ... \underline{\boldsymbol{e}}_{y_t} \underline{\boldsymbol{e}}_{\text{<eos>}}). \quad (4)$$

We find that removing the attention on the $\underline{\boldsymbol{e}}_{x_t} ... \underline{\boldsymbol{e}}_{y_t} \underline{\boldsymbol{e}}_{\text{<eos>}}$ significantly deteriorates our validation loss for sentence representation because there are often too many facets to be compressed into a single embedding. On the other hand, the encoder-decoder attention does not significantly change the performance of phrase representation, so we remove the connection (i.e., encoder and decoder have the same architecture) in models for phrase representation. Notice that the framework is flexible. For example, we can encode the genre of the document containing the sentence if desired.

3 Experiments

Quantitatively evaluating the quality of our predicted cluster centers is difficult because the existing label data and metrics are built for global clustering. The previous multi-sense word embedding studies often show that multiple embeddings could improve the single word embedding in the unsupervised word similarity task to demonstrate its effectiveness. Thus, our goal of experiments is to discover when and how the multi-facet embeddings can improve the similarity measurement in various unsupervised semantic tasks upon the widely-used general-purpose representations, such as single embedding and (contextualized) word embeddings.

3.1 Experiment Setup

Our models only require the raw corpus and sentence/phrase boundaries, so we will only compare them with other unsupervised alternatives that do not require any manual labels or multi-lingual resources such as PPDB (Pavlick et al. 2015). To simplify the comparison, we also omit the comparison with the methods using character-level information such as fastText (Bojanowski et al. 2017) or bigram information such as Sent2Vec (Pagliardini, Gupta, and Jaggi 2018a).

It is hard to make a fair comparison with BERT (Devlin et al. 2019). Its masked language modeling loss is designed for downstream supervised tasks and preserves more syntax information which might be distractive in unsupervised semantic applications. Furthermore, BERT uses word piece tokenization while other models use word tokenization. Nevertheless, we still present the performances of the BERT Base model as a reference even though it is trained using more parameters, larger embedding size, larger corpus, and more computational resources compared with our models. Since we focus on unsupervised setting, we directly use the final hidden states of the BERT models without supervised fine-tuning in most of the comparisons. One exception is that we also report the performance of sentence-BERT (Reimers and Gurevych 2019) in a low-resource setting.

Our model is trained on English Wikipedia 2016 while the stop words are removed from the set of co-occurring words. In the phrase experiments, we only consider noun phrases, and their boundaries are extracted by applying simple regular expression rules to POS tags before training. We use

[2] The decoder can also be viewed as another Transformer encoder which attends the output of the first encoder and models the dependency between predicted cluster centers.

Input Phrase: civil order <eos>	
Output Embedding (K = 1):	
e1	government 0.817 civil 0.762 citizens 0.748
Output Embeddings (K = 3):	
e1	initiatives 0.736 organizations 0.725 efforts 0.725
e2	army 0.815 troops 0.804 soldiers 0.786
e3	court 0.758 federal 0.757 judicial 0.736

Input Sentence: SMS messages are used in some countries as reminders of hospital appointments . <eos>	
Output Embedding (K = 1):	
e1	information 0.702, use 0.701, specific 0.700
Output Embeddings (K = 3):	
e1	can 0.769, possible 0.767, specific 0.767
e2	hospital 0.857, medical 0.780, hospitals 0.739
e3	SMS 0.791, Mobile 0.635, Messaging 0.631
Output Embeddings (K = 10):	
e1	can 0.854, should 0.834, either 0.821
e2	hospital 0.886, medical 0.771, hospitals 0.745
e3	services 0.768, service 0.749, web 0.722
e4	SMS 0.891, sms 0.745, messaging 0.686
e5	messages 0.891, message 0.801, emails 0.679
e6	systems 0.728, technologies 0.725, integrated 0.723
e7	appointments 0.791, appointment 0.735, duties 0.613
e8	confirmation 0.590, request 0.568, receipt 0.563
e9	countries 0.855, nations 0.737, Europe 0.732
e10	Implementation 0.613, Application 0.610, Programs 0.603

Table 1: Examples of the codebook embeddings predicted by our models with different K. The embedding in each row is visualized by the three words whose GloVe embeddings have the highest cosine similarities (also presented) with the codebook embedding.

the cased version (840B) of GloVe embedding (Pennington, Socher, and Manning 2014) as the pre-trained word embedding space for our sentence representation and use the uncased version (42B) for phrase representation.[3] To control the effect of embedding size, we set the hidden state size in our transformers as the GloVe embedding size (300).

Limited by computational resources, we train all the models using one GPU (e.g., NVIDIA 1080 Ti) within a week. Because of the relatively small model size, we find that our models underfit the data after a week (i.e., the training loss is very close to the validation loss).

3.2 Qualitative Evaluation

The cluster centers predicted by our model are visualized in Table 1 (as using *girl* and *lady* to visualize the red cluster center in Figure 2). Some randomly chosen examples are also visualized in Appendix D.

The centers summarize the input sequence well and more codebook embeddings capture more fine-grained semantic facets of a phrase or a sentence. Furthermore, the embeddings capture the compositional meaning of words. For example, each word in the phrase *civil order* does not mean *initiatives*, *army*, or *court*, which are facets of the whole phrase. When the input is a sentence, we can see that the output embeddings are sometimes close to the embeddings of words in the input sentence, which explains why attending the contextualized word embeddings in our decoder could improve the quality of the output embeddings.

3.3 Unsupervised Sentence Similarity

We propose two ways to evaluate the multi-facet embeddings using sentence similarity tasks.

First way: Since similar sentences should have similar word distribution in nearby sentences and thus similar codebook embeddings, the codebook embeddings of a query sentence $\widehat{\boldsymbol{F}}_{\boldsymbol{u}}(\boldsymbol{S_q^1})$ should be able to well reconstruct the codebook embeddings of its similar sentence $\widehat{\boldsymbol{F}}_{\boldsymbol{u}}(\boldsymbol{S_q^2})$. We compute the reconstruction error of both directions and add them as a symmetric distance **SC**:

$$\begin{aligned} SC(S_q^1, S_q^2) = {} & Er(\widehat{\boldsymbol{F}}_{\boldsymbol{u}}(\boldsymbol{S_q^1}), \widehat{\boldsymbol{F}}_{\boldsymbol{u}}(\boldsymbol{S_q^2})) \\ & + Er(\widehat{\boldsymbol{F}}_{\boldsymbol{u}}(\boldsymbol{S_q^2}), \widehat{\boldsymbol{F}}_{\boldsymbol{u}}(\boldsymbol{S_q^1})), \end{aligned} \qquad (5)$$

where $\widehat{\boldsymbol{F}}_{\boldsymbol{u}}(\boldsymbol{S_q}) = [\frac{\underline{\boldsymbol{c}}_k^q}{||\underline{\boldsymbol{c}}_k^q||}]_{k=1...K}$ is a matrix of normalized codebook embeddings and Er function is defined in equation 1. We use the negative distance to represent similarity.

Second way: One of the main challenges in unsupervised sentence similarity tasks is that we do not know which words are more important in each sentence. Intuitively, if one word in a query sentence is more important, the chance of observing related/similar words in the nearby sentences should be higher. Thus, we should pay more attention to the words in a sentence that have higher cosine similarity with its multi-facet embeddings, a summary of the co-occurring word distribution. Specifically, our importance/attention weighting for all the words in the query sentence S_q is defined by

$$\underline{\boldsymbol{a}}_{\boldsymbol{q}} = \max(0, \boldsymbol{W}(\boldsymbol{S_q})^T \widehat{\boldsymbol{F}}_{\boldsymbol{u}}(\boldsymbol{S_q}))\ \underline{\boldsymbol{1}}, \qquad (6)$$

where $\underline{\boldsymbol{1}}$ is an all-one vector. We show that the attention vector (denoted as **Our a** in Table 2) could be combined with various scoring functions and boost their performances. As a baseline, we also report the performance of the attention weights derived from the k-means loss rather than NNSC loss and call it **Our a (k-means)**.

Setup: STS benchmark (Cer et al. 2017) is a widely used sentence similarity task. We compare the correlations between the predicted semantic similarity and the manually labeled similarity. We report Pearson correlation coefficient, which is strongly correlated with Spearman correlation in all our experiments. Intuitively, when two sentences are less similar to each other, humans tend to judge the similarity based on how similar their facets are. Thus, we also compare the performances on the lower half of the datasets where their ground truth similarities are less than the median similarity in the dataset, and we call this benchmark STSB Low.

A simple but effective way to measure sentence similarity is to compute the cosine similarity between the average (contextualized) word embedding (Milajevs et al. 2014). The scoring function is labeled as **Avg**. Besides, we test the sentence embedding from BERT and from skip-thought (Kiros et al. 2015) (denoted as **CLS** and **Skip-thought Cosine**, respectively).

[3] nlp.stanford.edu/projects/glove/

Sentences		A man is lifting weights in a garage .		A man is lifting weights .
Output Embeddings	e1	can 0.872, even 0.851, should 0.850	e1	can 0.865, either 0.843, should 0.841
	e2	front 0.762, bottom 0.742, down 0.714	e2	front 0.758, bottom 0.758, sides 0.691
	e3	lifting 0.866, lift 0.663, Lifting 0.621	e3	lifting 0.847, lift 0.635, Lifting 0.610
	e4	garage 0.876, garages 0.715, basement 0.707	e4	lifting 0.837, lift 0.652, weights 0.629
	e5	decreasing 0.677, decreases 0.655, negligible 0.649	e5	decreasing 0.709, decreases 0.685, increases 0.682
	e6	weights 0.883, Weights 0.678, weight 0.665	e6	weights 0.864, weight 0.700, Weights 0.646
	e7	cylindrical 0.700, plurality 0.675, axial 0.674	e7	annular 0.738, cylindrical 0.725, circumferential 0.701
	e8	configurations 0.620, incorporating 0.610, utilizing 0.605	e8	methods 0.612, configurations 0.610, graphical 0.598
	e9	man 0.872, woman 0.682, men 0.672	e9	sweating 0.498, cardiovascular 0.494, dehydration 0.485
	e10	man 0.825, men 0.671, woman 0.653	e10	man 0.888, woman 0.690, men 0.676

Figure 3: Comparison of our attention weights and the output embeddings between two similar sentences from STSB. A darker red indicates a larger attention value in equation 6 and the output embeddings are visualized using the same way in Table 1.

Method		Dev		Test	
Score	Model	All	Low	All	Low
Cosine	Skip-thought	43.2	28.1	30.4	21.2
CLS	BERT	9.6	-0.4	4.1	0.2
Avg		62.3	42.1	51.2	39.1
SC	Our c K1	55.7	43.7	47.6	45.4
	Our c K10	63.0	51.8	52.6	47.8
WMD	GloVe	58.8	35.3	40.9	25.4
	Our a K1	63.1	43.3	47.5	34.8
	Our a K10	66.7	47.4	52.6	39.8
Prob_WMD	GloVe	75.1	59.6	63.1	52.5
	Our a K1	74.4	60.8	62.9	54.4
	Our a K10	**76.2**	**62.6**	**66.1**	58.1
Avg	GloVe	51.7	32.8	36.6	30.9
	Our a K1	54.5	40.2	44.1	40.6
	Our a K10	61.7	47.1	50.0	46.5
Prob_avg	GloVe	70.7	56.6	59.2	54.8
	Our a K1	68.5	56.0	58.1	55.2
	Our a K10	72.0	60.5	61.4	**59.3**
SIF†	GloVe	75.1	65.7	63.2	58.1
	Our a K1	72.5	64.0	61.7	58.5
	Our a K10	**75.2**	**67.6**	**64.6**	**62.2**
	Our a (k-means) K10	71.5	62.3	61.5	57.2
sentence-BERT (100 pairs)*		71.2	55.5	64.5	58.2

Table 2: Pearson correlation (%) in the development and test sets in the STS benchmark. The performances of all sentence pairs are indicated as All. Low means the performances on the half of sentence pairs with lower similarity (i.e., STSB Low). Our c means our codebook embeddings and Our a means our attention vectors. * indicates a supervised method. † indicates the methods which use training distribution to approximate testing distribution. The best score with and without † are highlighted.

In order to deemphasize the syntax parts of the sentences, Arora, Liang, and Ma (2017) propose to weight the word w in each sentence according to $\frac{\alpha}{\alpha+p(w)}$, where α is a constant and $p(w)$ is the probability of seeing the word w in the corpus. Following its recommendation, we set α to be 10^{-4} in this paper. After the weighting, we remove the first principal component of all the sentence embeddings in the training data as suggested by Arora, Liang, and Ma (2017) and denote the method as **SIF**. The post-processing requires an estimation of testing embedding distribution, which is not desired in some applications, so we also report the performance before removing the principal component, which is called **Prob_avg**.

We also test word mover's distance (**WMD**) (Kusner et al. 2015), which explicitly matches every word in a pair of sentences. As we do in **Prob_avg**, we apply $\frac{\alpha}{\alpha+p(w)}$ to **WMD** to down-weight the importance of functional words, and call this scoring function as **Prob_WMD**. When using **Our a**, we multiple our attention vector with the weights of every word (e.g., $\frac{\alpha}{\alpha+p(w)}$ for **Prob_avg** and **Prob_WMD**).

To motivate the unsupervised setting, we present the performance of sentence-BERT (Reimers and Gurevych 2019) that are trained by 100 sentence pairs. We randomly sample the sentence pairs from a data source that is not included in STSB (e.g., headlines in STS 2014), and report the testing performance averaged across all the sources from STS 2012 to 2016. More details are included in Appendix B.2.

Results: In Figure 3, we first visualize our attention weights in equation 6 and our output codebook embeddings for a pair of similar sentences from STSB to intuitively explain why modeling co-occurring distribution could improve the similarity measurement.

Many similar sentences might use different word choices or using extra words to describe details, but their possible nearby words are often similar. For example, appending *in the garage* to *A man is lifting weights* does not significantly change the facets of the sentences and thus the word *garage* receives relatively a lower attention weight. This makes its similarity measurement from our methods, **Our c** and **Our a**, closer to the human judgment than other baselines.

In Table 2, **Our c SC**, which matches between two sets of facets, outperforms **WMD**, which matches between two sets of words in the sentence, and also outperforms **BERT Avg**, especially in STSB Low. The significantly worse performances from **Skip-thought Cosine** justify our choice of ignoring the order in the co-occurring words.

All the scores in **Our * K10** are significantly better than **Our * K1**, which demonstrates the co-occurring word distribution is hard to be modeled well using a single embedding. Multiplying the proposed attention weighting consistently boosts the performance in all the scoring functions especially in STSB Low and without relying on the generalization assumption of the training distribution. Finally, using k-means loss, **Our a (k-means) K10**, significantly degrades the performance compared to **Our a K10**, which justify the proposed NNSC loss. In Appendix B.2, our methods are compared with more baselines using more datasets to test the effectiveness of multi-facet embeddings and our design

Setting	Method	R-1	R-2	Len
Unsup, No Sent Order	Random	28.1	8.0	68.7
	Textgraph (tfidf)†	33.2	11.8	-
	Textgraph (BERT)†	30.8	9.6	-
	W Emb (GloVe)	26.6	8.8	37.0
	Sent Emb (GloVe)	32.6	10.7	78.2
	W Emb (BERT)	31.3	11.2	45.0
	Sent Emb (BERT)	32.3	10.6	91.2
	Our c (K=3)	32.2	10.1	75.4
	Our c (K=10)	34.0	11.6	81.3
	Our c (K=100)	**35.0**	**12.8**	92.9
Unsup	Lead-3	40.3	17.6	87.0
	PACSUM (BERT)†	**40.7**	**17.8**	-
Sup	RL*	**41.7**	**19.5**	-

Table 3: The ROUGE F1 scores of different methods on CNN/Daily Mail dataset. The results with † are taken from Zheng and Lapata (2019). The results with * are taken from Celikyilmaz et al. (2018).

choices more comprehensively.

3.4 Unsupervised Extractive Summarization

The classic representation of a sentence uses either a single embedding or the (contextualized) embeddings of all the words in the sentence. In this section, we would like to show that both options are not ideal for extracting a set of sentences as a document summary.

Table 1 indicates that our multiple codebook embeddings of a sentence capture its different facets well, so we represent a document summary S as the union of the multi-facet embeddings of the sentences in the summary $R(S) = \cup_{t=1}^{T}\{\widehat{F}_u(S_t)\}$, where $\{\widehat{F}_u(S_t)\}$ is the set of column vectors in the matrix $\widehat{\boldsymbol{F}}_{\boldsymbol{u}}(\boldsymbol{S_t})$ of sentence S_t.

A good summary should cover multiple facets that well represent all topics/concepts in the document (Kobayashi, Noguchi, and Yatsuka 2015). The objective can be quantified as discovering a summary S whose multiple embeddings $R(S)$ best reconstruct the distribution of normalized word embedding $\underline{\boldsymbol{w}}$ in the document D (Kobayashi, Noguchi, and Yatsuka 2015). That is,

$$\arg\max_{S} \sum_{\underline{\boldsymbol{w}}\in D} \frac{\alpha}{\alpha + p(w)} \max_{\underline{\boldsymbol{s}}\in R(S)} \underline{\boldsymbol{w}}^{T} \underline{\boldsymbol{s}}, \tag{7}$$

where $\frac{\alpha}{\alpha+p(w)}$ is the weights of words we used in the sentence similarity experiments (Arora, Liang, and Ma 2017). We greedily select sentences to optimize equation 7 as in Kobayashi, Noguchi, and Yatsuka (2015).

Setup: We compare our multi-facet embeddings with other alternative ways of modeling the facets of sentences. A simple way is to compute the average word embedding as a single-facet sentence embedding.[4] This baseline is labeled as **Sent Emb**. Another way is to use the (contextualized) embedding of all the words in the sentences as different facets of the sentences. Since longer sentences have more words, we normalize the gain of the reconstruction similarity by the sentence length. The method is denoted as **W Emb**. We also test the baselines of selecting random sentences (**Rnd**) and first 3 sentences (**Lead-3**) in the document.

The results on the testing set of CNN/Daily Mail (Hermann et al. 2015; See, Liu, and Manning 2017) are compared using F1 of ROUGE (Lin and Hovy 2003) in Table 3. R-1, R-2, and Len mean ROUGE-1, ROUGE-2, and average summary length, respectively. All methods choose 3 sentences by following the setting in Zheng and Lapata (2019). *Unsup, No Sent Order* means the methods do not use the sentence order information in CNN/Daily Mail.

In CNN/Daily Mail, the unsupervised methods which access sentence order information such as **Lead-3** have performances similar to supervised methods such as RL (Celikyilmaz et al. 2018). To evaluate the quality of unsupervised sentence embeddings, we focus on comparing the unsupervised methods which do not assume the first few sentences form a good summary.

Results: In Table 3, predicting 100 clusters yields the best results. Notice that our method greatly alleviates the computational and sample efficiency challenges, which allows us to set cluster numbers K to be a relatively large number.

The results highlight the limitation of classic representations. The single sentence embedding cannot capture its multiple facets. On the other hand, if a sentence is represented by the embeddings of its words, it is difficult to eliminate the bias of selecting longer or shorter sentences and a facet might be composed by multiple words (e.g., the input sentence in Table 1 describes a service, but there is not a single word in the sentence that means service).

3.5 Unsupervised Phrase Similarity

Recently, Dubossarsky, Grossman, and Weinshall (2018) discovered that the multiple embeddings of each word may not improve the performance in word similarity benchmarks even if they capture more senses or facets of polysemies. Since our method can improve the sentence similarity estimation, we want to see whether multi-facet embeddings could also help the phrase similarity estimation.

In addition to **SC** in equation 5, we also compute the average of the contextualized word embeddings from our transformer encoder as the phrase embedding. We find that the cosine similarity between the two phrase embeddings is a good similarity estimation, and the method is labeled as **Ours Emb**.

Setup: We evaluate our phrase similarity using SemEval 2013 task 5(a) English (Korkontzelos et al. 2013) and Turney 2012 (Turney 2012). The task of SemEval 2013 is to distinguish similar phrase pairs from dissimilar phrase pairs. In Turney (5), given each query bigram, each model predicts the most similar unigram among 5 candidates, and Turney (10) adds 5 more negative phrase pairs by pairing the reverse of the query bigram with the 5 unigrams.

Results: The performances are presented in Table 4. **Ours (K=1)** is usually slightly better than **Ours (K=10)**, and the result supports the finding of Dubossarsky, Grossman, and Weinshall (2018). We hypothesize that unlike sentences, most of the phrases have only one facet/sense, and thus can

[4] Although equation 7 weights each word in the document, we find that the weighting $\frac{\alpha}{\alpha+p(w)}$ does not improve the sentence representation when averaging the word embeddings.

Method		SemEval 2013		Turney (5)	Turney (10)
Model	Score	AUC	F1	Accuracy	Accuracy
BERT	CLS	54.7	66.7	29.2	15.5
	Avg	66.5	67.1	43.4	24.3
GloVe	Avg	79.5	73.7	25.9	12.9
FCT LM†	Emb	-	67.2	42.6	27.6
Ours (K=10)	SC	80.3	72.8	45.6	28.8
	Emb	85.6	77.1	49.4	31.8
Ours (K=1)	SC	81.1	72.7	45.3	28.4
	Emb	**87.8**	**78.6**	**50.3**	**32.5**

Table 4: Performance of phrase similarity tasks. Every model is trained on a lowercased corpus. In SemEval 2013, AUC (%) is the area under the precision-recall curve of classifying similar phrase pairs. In Turney, we report the accuracy (%) of predicting the correct similar phrase pair among 5 or 10 candidate pairs. The results with † are taken from Yu and Dredze (2015).

be modeled by a single embedding well. In Appendix B.1, the hypernym detection results also support this hypothesis.

Even though being slightly worse, the performances of **Ours (K=10)** remain strong compared with baselines. This implies that the similarity performances are not sensitive to the number of clusters as long as sufficiently large K is used because the model is able to output multiple nearly duplicated codebook embeddings to represent one facet (e.g., using two centers to represent the facet related to *company* in Figure 1). The flexibility alleviates the issues of selecting K in practice. Finally, the strong performances in Turney (10) verify that our encoder respects the word order when composing the input sequence.

4 Related Work

Topic modeling (Blei, Ng, and Jordan 2003) has been extensively studied and widely applied due to its interpretability and flexibility of incorporating different forms of input features (Mimno and McCallum 2008). Cao et al. (2015); Srivastava and Sutton (2017) demonstrate that neural networks could be applied to discover semantically coherent topics. Instead of optimizing a global topic model, our goal is to efficiently discover different sets of topics/clusters on the words beside each (unseen) phrase or sentence.

Sparse coding on word embedding space is used to model the multiple facets of a word (Faruqui et al. 2015; Arora et al. 2018), and parameterizing word embeddings using neural networks is used to test hypothesis (Han et al. 2018) and save storage space (Shu and Nakayama 2018). Besides, to capture asymmetric relations such as hypernyms, words are represented as single or multiple regions in Gaussian embeddings (Vilnis and McCallum 2015; Athiwaratkun and Wilson 2017) rather than a single point. However, the challenges of extending these methods to longer sequences are not addressed in these studies.

One of our main challenges is to design a loss for learning to predict cluster centers while modeling the dependency among the clusters. This requires a matching step between two sets and computing the distance loss after the matching (Eiter and Mannila 1997). One popular loss is called Chamfer distance, which is widely adopted in the autoencoder models for point clouds (Yang et al. 2018a; Liu et al. 2019), while more sophisticated matching loss options are also proposed (Stewart, Andriluka, and Ng 2016; Balles and Fischbacher 2019). The goal of the previous studies focuses on measuring symmetric distances between the ground truth set and predicted set (usually with an equal size), while our loss tries to reconstruct the ground truth set using much fewer codebook embeddings.

Other ways to achieve the permutation invariant loss for neural networks include sequential decision making (Welleck et al. 2018), mixture of experts (Yang et al. 2018b; Wang, Cho, and Wen 2019), beam search (Qin et al. 2019), predicting the permutation using a CNN (Rezatofighi et al. 2018), Transformers (Stern et al. 2019; Gu, Liu, and Cho 2019; Carion et al. 2020) or reinforcement learning (Welleck et al. 2019). In contrast, our goal is to efficiently predict a set of cluster centers that can well reconstruct the set of observed instances rather than directly predicting the observed instances.

5 Conclusions

In this work, we propose a framework for learning the co-occurring distribution of the words surrounding a sentence or a phrase. Even though there are usually only a few words that co-occur with each sentence, we demonstrate that the proposed models can learn to predict interpretable cluster centers conditioned on an (unseen) sentence.

In the sentence similarity tasks, the results indicate that the similarity between two sets of multi-facet embeddings well correlates with human judgments, and we can use the multi-facet embeddings to estimate the word importance and improve various widely-used similarity measurements in a pre-trained word embedding space such as GloVe. In a single-document extractive summarization task, we demonstrate multi-facet embeddings significantly outperform classic unsupervised sentence embedding or individual word embeddings. Finally, the results of phrase similarity tasks suggest that a single embedding might be sufficient to represent the co-occurring word distribution of a phrase.

Acknowledgements

We thank Ao Liu and Mohit Iyyer for many helpful discussions and Nishant Yadav for suggesting several related work. We also thank the anonymous reviewers for their constructive feedback.

This work was supported in part by the Center for Data Science and the Center for Intelligent Information Retrieval, in part by the Chan Zuckerberg Initiative under the project Scientific Knowledge Base Construction, in part using high performance computing equipment obtained under a grant from the Collaborative R&D Fund managed by the Massachusetts Technology Collaborative, in part by the National Science Foundation (NSF) grant numbers DMR-1534431 and IIS-1514053.

Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

Ethics Statement

We propose a novel framework, neural architecture, and loss to learn multi-facet embedding from the co-occurring statistics in NLP. In this study, we exploit the co-occurring relation between a sentence and its nearby words to improve the sentence representation. In our follow-up studies, we discover that the multi-facet embeddings could also be used to learn other types of co-occurring statistics. For example, we can use the co-occurring relation between a scientific paper and its citing paper to improve paper recommendation methods in Bansal, Belanger, and McCallum (2016). Paul, Chang, and McCallum (2021) use the co-occurring relation between a sentence pattern and its entity pair to improve relation extraction in Verga et al. (2016). Chang et al. (2021) use the co-occurring relation between a context paragraph and its subsequent words to control the topics of language generation. In the future, the approach might also be used to improve the efficiency of document similarity estimation (Luan et al. 2020).

On the other hand, we improve the sentence similarity and summarization tasks in this work using the assumption that important words are more likely to appear in the nearby sentences. The assumption might be violated in some domains and our method might degrade the performances in such domains if the practitioner applies our methods without considering the validity of the assumption.

References

Agirre, E.; Banea, C.; Cardie, C.; Cer, D.; Diab, M.; Gonzalez-Agirre, A.; Guo, W.; Lopez-Gazpio, I.; Maritxalar, M.; Mihalcea, R.; Rigau, G.; Uria, L.; and Wiebe, J. 2015. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *SemEval*.

Agirre, E.; Banea, C.; Cardie, C.; Cer, D.; Diab, M.; Gonzalez-Agirre, A.; Guo, W.; Mihalcea, R.; Rigau, G.; and Wiebe, J. 2014. Semeval-2014 task 10: Multilingual semantic textual similarity. In *SemEval*.

Agirre, E.; Banea, C.; Cer, D.; Diab, M.; Gonzalez-Agirre, A.; Mihalcea, R.; Rigau, G.; and Wiebe, J. 2016. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *SemEval*.

Agirre, E.; Cer, D.; Diab, M.; Gonzalez-Agirre, A.; and Guo, W. 2013. * SEM 2013 shared task: Semantic textual similarity. In ** SEM*.

Agirre, E.; Diab, M.; Cer, D.; and Gonzalez-Agirre, A. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In *SemEval*.

Arora, S.; Li, Y.; Liang, Y.; Ma, T.; and Risteski, A. 2018. Linear algebraic structure of word senses, with applications to polysemy. *Transactions of the Association of Computational Linguistics* 6: 483–495.

Arora, S.; Liang, Y.; and Ma, T. 2017. A Simple but Tough-to-beat Baseline for Sentence Embeddings. In *ICLR*.

Asaadi, S.; Mohammad, S. M.; and Kiritchenko, S. 2019. Big BiRD: A Large, Fine-Grained, Bigram Relatedness Dataset for Examining Semantic Composition. In *NAACL-HLT*.

Athiwaratkun, B.; and Wilson, A. 2017. Multimodal Word Distributions. In *ACL*.

Balles, L.; and Fischbacher, T. 2019. Holographic and other Point Set Distances for Machine Learning. URL https://openreview.net/forum?id=rJlpUiAcYX.

Bansal, T.; Belanger, D.; and McCallum, A. 2016. Ask the GRU: Multi-task Learning for Deep Text Recommendations. In *RecSys*.

Bentley, J. L. 1975. Multidimensional binary search trees used for associative searching. *Communications of the ACM* 18(9): 509–517.

Bird, S.; Klein, E.; and Loper, E. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3(Jan): 993–1022.

Bojanowski, P.; Grave, E.; Joulin, A.; and Mikolov, T. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics* 5: 135–146.

Cao, Z.; Li, S.; Liu, Y.; Li, W.; and Ji, H. 2015. A novel neural topic model and its supervised extension. In *AAAI*.

Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-End Object Detection with Transformers. *arXiv preprint arXiv:2005.12872* .

Celikyilmaz, A.; Bosselut, A.; He, X.; and Choi, Y. 2018. Deep Communicating Agents for Abstractive Summarization. In *NAACL-HLT*.

Cer, D.; Diab, M.; Agirre, E.; Lopez-Gazpio, I.; and Specia, L. 2017. SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation. In *SemEval-2017*.

Chang, H.-S.; Yuan, J.; Iyyer, M.; and McCallum, A. 2021. Changing the Mind of Transformers for Topically-Controllable Language Generation. In *EACL*.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*.

Dubossarsky, H.; Grossman, E.; and Weinshall, D. 2018. Coming to your senses: on controls and evaluation sets in polysemy research. In *EMNLP*.

Eiter, T.; and Mannila, H. 1997. Distance measures for point sets and their computation. *Acta Informatica* 34(2): 109–133.

Faruqui, M.; Tsvetkov, Y.; Yogatama, D.; Dyer, C.; and Smith, N. A. 2015. Sparse Overcomplete Word Vector Representations. In *ACL*.

Gu, J.; Liu, Q.; and Cho, K. 2019. Insertion-based decoding with automatically inferred generation order. *Transactions of the Association for Computational Linguistics* 7: 661–676.

Han, R.; Gill, M.; Spirling, A.; and Cho, K. 2018. Conditional Word Embedding and Hypothesis Testing via Bayes-by-Backprop. In *EMNLP*.

Hermann, K. M.; Kocisky, T.; Grefenstette, E.; Espeholt, L.; Kay, W.; Suleyman, M.; and Blunsom, P. 2015. Teaching machines to read and comprehend. In *NeurIPS*.

Hoyer, P. O. 2002. Non-negative Sparse Coding. In *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing*.

Huang, L.; Ji, H.; et al. 2017. Learning Phrase Embeddings from Paraphrases with GRUs. In *Proceedings of the First Workshop on Curation and Applications of Parallel and Comparable Corpora*.

Kiros, R.; Zhu, Y.; Salakhutdinov, R. R.; Zemel, R.; Urtasun, R.; Torralba, A.; and Fidler, S. 2015. Skip-thought vectors. In *NeurIPS*.

Kobayashi, H.; Noguchi, M.; and Yatsuka, T. 2015. Summarization based on embedding distributions. In *EMNLP*.

Korkontzelos, I.; Zesch, T.; Zanzotto, F. M.; and Biemann, C. 2013. Semeval-2013 task 5: Evaluating phrasal semantics. In *SemEval 2013*.

Kumar, S.; and Tsvetkov, Y. 2019. Von Mises-Fisher Loss for Training Sequence to Sequence Models with Continuous Outputs. In *ICLR*.

Kusner, M.; Sun, Y.; Kolkin, N.; and Weinberger, K. 2015. From word embeddings to document distances. In *ICML*.

Lau, J. H.; Cook, P.; McCarthy, D.; Newman, D.; and Baldwin, T. 2012. Word sense induction for novel sense detection. In *EACL*.

Lee, J.; Lee, Y.; Kim, J.; Kosiorek, A. R.; Choi, S.; and Teh, Y. W. 2019. Set transformer: A framework for attention-based permutation-invariant neural networks. In *ICML*.

Li, L. H.; Chen, P. H.; Hsieh, C.-J.; and Chang, K.-W. 2019. Efficient Contextual Representation Learning With Continuous Outputs. *Transactions of the Association for Computational Linguistics* 7: 611–624.

Lin, C.-Y.; and Hovy, E. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *NAACL-HLT*.

Liu, X.; Han, Z.; Wen, X.; Liu, Y.-S.; and Zwicker, M. 2019. L2g auto-encoder: Understanding point clouds by local-to-global reconstruction with hierarchical self-attention. In *Proceedings of the 27th ACM International Conference on Multimedia*.

Luan, Y.; Eisenstein, J.; Toutanova, K.; and Collins, M. 2020. Sparse, Dense, and Attentional Representations for Text Retrieval. *arXiv preprint arXiv:2005.00181* .

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *NeurIPS*.

Milajevs, D.; Kartsaklis, D.; Sadrzadeh, M.; and Purver, M. 2014. Evaluating Neural Word Representations in Tensor-Based Compositional Settings. In *EMNLP*.

Mimno, D. M.; and McCallum, A. 2008. Topic Models Conditioned on Arbitrary Features with Dirichlet-multinomial Regression. In *UAI*.

Neelakantan, A.; Shankar, J.; Passos, A.; and McCallum, A. 2014. Efficient Non-parametric Estimation of Multiple Embeddings per Word in Vector Space. In *EMNLP*.

Newman-Griffis, D.; Lai, A. M.; and Fosler-Lussier, E. 2018. Jointly Embedding Entities and Text with Distant Supervision. In *Proceedings of the 3rd Workshop on Representation Learning for NLP (Repl4NLP)*.

Pagliardini, M.; Gupta, P.; and Jaggi, M. 2018a. Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. In *NAACL-HLT*.

Pagliardini, M.; Gupta, P.; and Jaggi, M. 2018b. Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. In *NAACL*.

Paul, R.; Chang, H.-S.; and McCallum, A. 2021. Multi-facet Universal Schema. In *EACL*.

Pavlick, E.; Rastogi, P.; Ganitkevitch, J.; Van Durme, B.; and Callison-Burch, C. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *ACL*.

Pennington, J.; Socher, R.; and Manning, C. 2014. GloVe: Global vectors for word representation. In *EMNLP*.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep contextualized word representations. In *NAACL-HLT*.

Qin, K.; Li, C.; Pavlu, V.; and Aslam, J. A. 2019. Adapting RNN Sequence Prediction Model to Multi-label Set Prediction. In *NAACL-HLT*.

Reimers, N.; and Gurevych, I. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *EMNLP-IJCNLP*.

Rezatofighi, S. H.; Kaskman, R.; Motlagh, F. T.; Shi, Q.; Cremers, D.; Leal-Taixé, L.; and Reid, I. 2018. Deep perm-set net: learn to predict sets with unknown permutation and cardinality using deep neural networks. *arXiv preprint arXiv:1805.00613* .

See, A.; Liu, P. J.; and Manning, C. D. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *ACL*.

Shu, R.; and Nakayama, H. 2018. Compressing Word Embeddings via Deep Compositional Code Learning. In *ICLR*.

Shwartz, V.; Goldberg, Y.; and Dagan, I. 2016. Improving Hypernymy Detection with an Integrated Path-based and Distributional Method. In *ACL*.

Singh, S. P.; Hug, A.; Dieuleveut, A.; and Jaggi, M. 2020. Context mover's distance & barycenters: Optimal transport of contexts for building representations. In *International Conference on Artificial Intelligence and Statistics*.

Srivastava, A.; and Sutton, C. A. 2017. Autoencoding Variational Inference For Topic Models. In *ICLR*.

Stern, M.; Chan, W.; Kiros, J.; and Uszkoreit, J. 2019. Insertion Transformer: Flexible Sequence Generation via Insertion Operations. In *ICML*.

Stewart, R.; Andriluka, M.; and Ng, A. Y. 2016. End-to-end people detection in crowded scenes. In *CVPR*.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *NeurIPS*.

Tieleman, T.; and Hinton, G. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning* 4(2): 26–31.

Turney, P. D. 2012. Domain and function: A dual-space model of semantic relations and compositions. *Journal of Artificial Intelligence Research* .

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *NeurIPS*.

Verga, P.; Belanger, D.; Strubell, E.; Roth, B.; and McCallum, A. 2016. Multilingual Relation Extraction using Compositional Universal Schema. In *NAACL-HLT*.

Vilnis, L.; and McCallum, A. 2015. Word Representations via Gaussian Embedding. In *ICLR*.

Wang, T.; Cho, K.; and Wen, M. 2019. Attention-based mixture density recurrent networks for history-based recommendation. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*.

Welleck, S.; Brantley, K.; Daumé III, H.; and Cho, K. 2019. Non-Monotonic Sequential Text Generation. In *ICML*.

Welleck, S.; Yao, Z.; Gai, Y.; Mao, J.; Zhang, Z.; and Cho, K. 2018. Loss Functions for Multiset Prediction. In *NeurIPS*.

Yang, Y.; Feng, C.; Shen, Y.; and Tian, D. 2018a. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *CVPR*.

Yang, Z.; Dai, Z.; Salakhutdinov, R.; and Cohen, W. W. 2018b. Breaking the softmax bottleneck: A high-rank RNN language model. In *ICLR*.

Yu, M.; and Dredze, M. 2015. Learning composition models for phrase embeddings. *Transactions of the Association for Computational Linguistics* 3: 227–242.

Zheng, H.; and Lapata, M. 2019. Sentence Centrality Revisited for Unsupervised Summarization. In *ACL*.

A Structure of Appendix

We conduct more comprehensive experiments and analyses in Section B. The details of our method and experiments (e.g., training algorithm, preprocessing, and hyperparameter settings) are presented in Section C, and we visualize more codebook embeddings and the derived attention weights of the sentences in Section D.

B More Experiments

In the main paper, we show that multi-facet embeddings can improve the estimation of symmetric relations like similarity. To know whether they are also useful in asymmetric relations like entailment, we test our method on a hypernym detection dataset in Section B.1.

Due to the page limits, we cannot present all of our results in the main paper, so we put more comprehensive analyses for sentence similarity tasks in Section B.2, for extractive summarization in Section B.3, and for phrase similarity tasks in Section B.4. We also present the results of BERT Large model in Section B.5 as a reference. Section B.6 and B.7 provide some motivating examples for a sentence similarity task and for the extractive summarization, respectively.

B.1 Unsupervised Hypernymy Detection

We apply our model to HypeNet (Shwartz, Goldberg, and Dagan 2016), an unsupervised hypernymy detection dataset, based on the assumption that the co-occurring words of a phrase are often less related to some of its hyponyms. For instance, *animal* is a hypernym of *brown dog*. *flies* is a co-occurring word of *animal* which is less related to *brown dog*.

Accordingly, the predicted codebook embeddings of a hyponym S_q^{hypo} (e.g., *brown dog*), which cluster the embeddings of co-occurring words (e.g., *eats*), often reconstruct the embeddings of its hypernym S_q^{hyper} (e.g., *animal*) better than the other way around (e.g., the embedding of *flies* cannot reconstruct the embeddings of *brown dog* well). That is, $Er(\widehat{\boldsymbol{F}}_{\boldsymbol{u}}(\boldsymbol{S_q^{hypo}}), \boldsymbol{W}(\boldsymbol{S_q^{hyper}}))$ is smaller than $Er(\widehat{\boldsymbol{F}}_{\boldsymbol{u}}(\boldsymbol{S_q^{hyper}}), \boldsymbol{W}(\boldsymbol{S_q^{hypo}})))$.

Based on the assumption, our asymmetric scoring function is defined as

$$\begin{aligned}\text{Diff}(S_q^{hyper}, S_q^{hypo}) = {} & Er(\widehat{\boldsymbol{F}}_{\boldsymbol{u}}(\boldsymbol{S_q^{hyper}}), \boldsymbol{W}(\boldsymbol{S_q^{hypo}})) \\ & - Er(\widehat{\boldsymbol{F}}_{\boldsymbol{u}}(\boldsymbol{S_q^{hypo}}), \boldsymbol{W}(\boldsymbol{S_q^{hyper}})).\end{aligned} \tag{8}$$

where Er function is defined in equation 1.

The AUC of detecting hypernym among other relations and accuracy of detecting the hypernym direction are compared in Table 5. Our methods outperform baselines, which only provide symmetric similarity measurement, and **Ours (K=1)** performs similarly compared with **Ours (K=10)**.

B.2 More Analysis on Sentence Similarity

We design more experiments and present the results in Table 6 and Table 7 in order to answer the following research questions.

1. Is ignoring the order of co-occurring words effective in emphasizing the semantic side of the sentences?

Method		Dev		Test	
Model	Score	AUC	Acc	AUC	Acc
BERT	CLS	20.6	50	21.3	50
	Avg	25.6	50	25.6	50
GloVe	Avg	17.4	50	17.7	50
Our c K10	Diff	**29.4**	78.9	**29.6**	79.1
Our c K1	Diff	29.3	**82.7**	**29.6**	**81.0**

Table 5: Hypernym detection performances in the development and test set of HypeNet. AUC (%) refers to the area under precision and recall curve, which measures the quality of retrieving hypernym phrases. Acc (%) means the accuracy of predicting specificity given a pair of hypernym phrases.

Method		Dev		Test	
Score	Model	All	Low	All	Low
Cosine	Skip-thought	43.2	28.1	30.4	21.2
Avg	ELMo	65.6	47.4	54.2	44.1
Prob_avg	ELMo	70.3	54.6	60.4	54.2
	Our a (GloVe) K1	69.3	54.1	60.8	55.8
	Our a (GloVe) K10	70.5	55.9	61.1	56.6
Avg	BERT	62.3	42.1	51.2	39.1
Prob_avg		72.1	57.0	57.8	55.1
Avg	Sent2Vec	71.9	51.2	63.6	46.0
	Our a (GloVe) K10	76.1	62.9	**71.5**	**62.7**
	Our a (GloVe) K1	72.0	56.1	66.8	55.7
SC	NNSC clustering K10	38.6	37.8	25.4	38.9
	Our c (w2v) K10	54.7	38.8	43.9	36.0
	Our c (k-means) K10	37.8	25.9	29.5	19.7
	Our c (LSTM) K10	58.9	49.2	49.8	46.4
	Our c (GloVe) K10	63.0	51.8	52.6	47.8
Prob_WMD	w2v	72.9	56.6	62.1	54.0
	Our a (w2v) K10	73.6	60.1	63.5	57.8
Prob_avg	w2v	68.3	53.7	54.3	50.9
	Our a (w2v) K10	68.3	56.8	55.1	53.1
SIF†	w2v	70.5	56.9	59.4	54.7
	Our a (w2v) K10	71.6	60.9	61.3	57.6
Prob_WMD	GloVe	75.1	59.6	63.1	52.5
	Our a (k-means) K10	72.5	57.9	60.3	49.9
	Our a (LSTM) K10	**76.3**	**63.2**	65.8	57.4
	Our a (GloVe) K10	76.2	62.6	66.1	58.1
Prob_avg	GloVe	70.7	56.6	59.2	54.8
	Our a (k-means) K10	66.6	53.4	55.8	51.8
	Our a (LSTM) K10	71.7	60.1	61.3	58.3
	Our a (GloVe) K10	72.0	60.5	61.4	59.3
SIF†	GloVe	75.1	65.7	63.2	58.1
	Our a (k-means) K10	71.5	62.3	61.5	57.2
	Our a (LSTM) K10	74.6	66.9	64.3	60.9
	Our a (GloVe) K10	**75.2**	**67.6**	**64.6**	**62.2**

Table 6: The Pearson correlation (%) in STS benchmarks. w2v means Word2Vec. Our * (k-means) means using the k-means loss rather than the NNSC loss. Our * (LSTM) means replacing the transformers in our encoder with bi-LSTM and replacing our transformer decoder with LSTM. Other abbreviations and symbols share the same meaning in Table 2.

To answer this question, we replace our transformer encoder with bi-LSTM and our transformer decoder with LSTM. Then, this architecture becomes very similar to skip-thought (Kiros et al. 2015) except that skip-thoughts decodes a sequence instead of a set, and we ignore the word order in the nearby sentences. As we can see in Table 6, **Our c (LSTM) K10 SC** performs much better than **Skip-thought Cosine**, which compute the cosine similarity between their sentence embeddings. This result further justifies our approach of ignoring the order of co-occurring words in our NNSC loss.

2. Is our word importance estimation generally useful for composing (contextualized) word embedding models?

We cannot apply our attention weights (i.e., **Our a**) to BERT because BERT uses word piece tokenization. Instead, we use the top layer of ELMo (Peters et al. 2018) as the contextualized word embedding, apply $\frac{\alpha}{\alpha+p(w)}$ weighting multiplied with our attention weights in equation 6 . The results in Table 6 show that the performance of **ELMo Prob_avg** could also be boosted by our attention weighting even though our model is trained on GloVe semantic space. The importance weights from multiple embeddings can also help boost the performance of a version of Sent2Vec (Pagliardini, Gupta, and Jaggi 2018b) that uses only unigram information.

3. Could our model be trained on word embedding space other than GloVe?

First, we train Word2Vec (Mikolov et al. 2013) (denoted as w2v) on the Wikipedia 2016 corpus. We then train our multi-facet embeddings to fit the Word2Vec embedding of co-occurring words in the Wikipedia 2016 corpus. The results in Table 6 show that **Our a (w2v) K10** improves the performance using different scoring functions as we did in GloVe space.

4. How well could clustering-based multi-facet embeddings perform on long text sequences such as sentences?

Lots of the testing sentences in the STS benchmark are not observed in our training corpus. To test clustering-based multi-facet embeddings, we first average word embedding in every sentence into sentence embedding, and for each testing query sentence, we perform approximated nearest neighbor search using KDTree (Bentley 1975) to retrieve 1000 most similar sentences. Then, we remove the stop words in the 1000 sentences and perform NNSC clustering on the rest of the words. Finally, we compute **SC** distance between two sets of cluster centers derived from testing sentence pairs and denote the baseline as **NNSC clustering K10 SC** in Table 6.

The testing time of this baseline is much slower than the proposed method due to the need for the nearest neighbor search, and its performance is also much worse. This result justifies our approach of predicting clustering centers directly to generate multi-facet embeddings.

5. How much better is NNSC loss compared with k-means loss?

In the method section, we mention that we adopt NNSC rather than k-means in our loss because k-means loss cannot generate diverse cluster centers in all of the neural architectures (including transformers and bi-LSTMs) we tried. We hypothesize that the k-means loss does not stably encourage predicted clusters to play different roles for reconstructing the embeddings of observed co-occurring words. We present the much worse results of the model using k-means loss in Table 6 to justify our usage of NNSC in our loss.

Dataset			Prob_avg			Prob_WMD		WMD	Avg			
Category	Data	Year	Our a K10	GloVe	Our a K1	Our a K10	GloVe	GloVe	w2v	BERT	ELMo	ST
forum	deft-forum	2014	**40.3**	33.2	35.6	**41.4**	33.3	28.5	33.9	25.6	**35.9**	22.5
	answers-forums	2015	**53.9**	49.6	43.4	**63.7**	61.4	48.7	51.2	55.1	**56.0**	36.8
	answer-answer	2016	**39.5**	34.1	33.1	**48.9**	45.0	46.4	31.0	50.4	**58.2**	32.3
	question-question	2016	**63.5**	61.4	60.1	**66.9**	62.6	28.7	**53.7**	44.9	38.4	42.3
	belief	2015	**58.6**	57.2	53.5	**67.8**	66.9	62.6	63.8	62.8	**71.1**	38.5
news	surprise.SMTnews	2012	**56.7**	50.9	54.4	**55.9**	49.0	48.4	48.7	**56.8**	55.1	44.2
	headlines	2013	**57.9**	55.9	53.3	**67.5**	66.2	59.6	54.8	55.1	**59.7**	44.6
	headlines	2014	**53.0**	51.2	49.1	**61.8**	59.9	53.3	50.4	53.2	**54.2**	41.6
	headlines	2015	**54.7**	51.8	49.8	**65.4**	63.6	59.1	51.8	53.3	**59.2**	47.6
	headlines	2016	**53.9**	52.0	49.8	**65.6**	64.6	62.6	50.3	**58.0**	57.3	46.6
definition	surprise.OnWN	2012	**66.6**	64.0	63.6	**69.8**	68.2	65.3	63.3	60.3	**68.8**	33.5
	OnWN	2013	**69.8**	68.6	63.6	**65.0**	61.8	35.6	**63.8**	59.7	44.8	21.1
	OnWN	2014	**75.9**	74.8	72.7	**73.8**	71.9	52.5	**74.5**	71.0	61.1	31.1
	FNWN	2013	**39.4**	38.6	38.9	45.7	**46.0**	40.1	28.0	36.5	**38.1**	11.0
captions	MSRvid	2012	**81.5**	81.2	80.5	**80.4**	78.3	46.3	**76.0**	52.3	63.0	54.3
	images	2014	**76.9**	74.6	75.5	**78.1**	75.1	57.3	**72.1**	55.4	64.9	62.8
	images	2015	76.1	**76.5**	74.5	**82.2**	81.8	67.3	72.8	66.3	**73.3**	29.5
education	answers-students	2015	53.3	**54.7**	52.5	66.4	68.5	**70.4**	**64.0**	62.8	60.4	41.5
	plagiarism	2016	72.1	**74.4**	72.2	78.5	**79.1**	71.2	74.0	76.6	**78.4**	53.6
out of domain	deft-news	2014	62.4	**65.6**	59.0	63.9	**65.2**	55.5	58.9	**73.3**	72.6	43.8
	tweet-news	2014	64.0	**66.2**	60.1	71.1	**72.7**	70.5	69.7	66.5	**72.0**	53.1
similar	MSRpar	2012	34.6	**43.4**	36.2	48.0	**53.2**	49.0	37.2	**40.5**	34.0	24.5
	SMTeuroparl	2012	52.3	**54.4**	42.8	**53.6**	53.3	51.3	**51.8**	46.0	46.8	28.4
	postediting	2016	65.4	**66.8**	63.5	78.9	**79.9**	78.0	74.3	79.1	**80.2**	57.4
STS	All	2012	68.4	**68.5**	68.0	**66.8**	64.9	40.0	**60.5**	32.5	44.1	6.6
	All	2013	**64.1**	62.0	58.9	**66.5**	64.2	47.3	**58.5**	57.4	54.1	36.3
	All	2014	**58.6**	56.1	52.7	**61.4**	58.8	39.2	**56.1**	53.6	52.4	25.0
	All	2015	**59.3**	56.8	54.2	**71.1**	70.1	58.1	**61.6**	58.1	57.4	25.8
	All	2016	**56.9**	53.7	52.4	**66.7**	62.8	49.6	52.9	**60.4**	57.9	38.8
	All	All	**61.4**	59.5	56.5	**66.8**	64.8	48.4	**59.1**	54.1	55.4	26.1
	Low	2012	67.1	**67.9**	**67.9**	**60.8**	59.2	17.5	**57.5**	18.8	33.0	-2.0
	Low	2013	**49.4**	45.1	44.3	**39.4**	32.7	10.5	**33.1**	31.5	27.4	22.9
	Low	2014	**50.2**	45.1	43.5	**50.5**	45.5	22.3	43.5	**46.6**	40.4	29.4
	Low	2015	**48.8**	45.3	43.8	**54.2**	50.9	33.4	**49.8**	38.2	41.3	18.7
	Low	2016	**51.0**	45.1	44.5	**53.8**	46.3	21.7	**39.4**	36.4	32.4	21.6
	Low	All	**51.2**	47.7	45.9	**52.3**	48.4	25.8	**45.9**	39.8	39.5	21.7

Table 7: Comparing Pearson correlation (%) of different unsupervised methods from STS 2012 to STS 2016. We highlight the best performance in each of the three blocks.

6. Could our method improve the similarity estimation of all kinds of datasets?

In Table 7, we compare the performance before and after applying our attention weights in the English part of STS 2012 (Agirre et al. 2012), 2013 (Agirre et al. 2013), 2014 (Agirre et al. 2014), 2015 (Agirre et al. 2015), and 2016 (Agirre et al. 2016). We categorize each of the dataset in different years based on either its source (*forum*, *news*, *definition*, *caption*, and *education*) or its characteristic (*out of domain* or *similar*).

Out of domain means the testing sentences are very different from our training corpus, Wikipedia 2016. *deft-news* from STS 2014 is included in this category because all the sentences in the dataset are lowercased. *Similar* means there are lots of pairs in the datasets whose two sentences have almost the identical meaning.

From the Table 7, we can see that **GloVe Prob_avg** and **GloVe Prob_WMD** perform well compare with other baselines, and the attention weights from our multi-facet embedding stably boost **GloVe Prob_avg** and **GloVe Prob_WMD** except in the categories *education*, *out of domain*, and *similar*. Thus, we recommend adopting our method when the source of training and testing sentences are not too different from each other, and the task is not to identify duplicated sentences.

7. Are supervised methods such as sentence-BERT sensitive to the training data?

Table 8 compares the performance of sentence-BERT (Reimers and Gurevych 2019) trained on different data sources. We observe that the performance of sentence-BERT could be degraded when the distribution of training data is very different from that of testing data. For example, Sentence-BERT also does not perform well when the training sentence pairs tend to be similar with each other (e.g., in *postediting* and *SMTeuroparl*) or come from a writing style that is different from the style of testing sentence pairs (e.g., *tweet-news* and *answers-students*).

Furthermore, a supervised model trained by a limited amount of labels could perform worse than the unsupervised alternatives. For example, on STSB Dev, the weighted average of word embedding (Prob_avg) outputted by the BERT base model outperforms the sentence-BERT trained by 100 labels on average. Sentence-BERT model trained by *SMTeuroparl* is even worse than just averaging all the contextualized word embeddings in BERT on STSB Test.

Training Data		Dev		Test	
Data Source	Year	All	Low	All	Low
SMTeuroparl	2012	63.5	41.7	49.6	43.3
surprise.SMTnews	2012	67.1	49.2	58.4	56.5
postediting	2016	70.3	53.9	60.5	56.3
tweet-news	2014	68.8	52.4	61.4	57.1
answers-students	2015	67.5	53.1	62.5	56.4
headlines	2014	69.2	52.8	62.6	53.8
plagiarism	2016	72.1	59.5	65.4	62.4
belief	2015	71.9	53.5	65.7	59.1
FNWN	2013	71.2	54.7	67.1	61.9
headlines	2015	73.8	58.9	67.9	53.1
question-question	2016	75.0	63.2	69.3	65.5
OnWN	2013	**75.9**	63.4	70.3	64.0
OnWN	2014	74.9	**63.5**	70.6	**65.6**
surprise.OnWN	2012	75.5	57.5	**71.5**	60.6
Average		71.2	55.5	64.5	58.2

Table 8: The Pearson correlation (%) of sentence-BERT on STS benchmark. The sentence-BERT is initialized by the BERT base model and trained by 100 samples in each data source. All results are the average of three runs. The order of rows is determined by their performance on the test set of STSB.

B.3 Summarization Comparison Given the Same Summary Length

In Section 3.4, we compare our methods with other baselines when all the methods choose the same number of sentences. We suspect that the bad performances for **W Emb (*)** methods (i.e., representing each sentence using the embedding of words in the sentence) might come from the tendency of selecting shorter sentences. To verify the hypothesis, we plot the R-1 performance of different unsupervised summarization methods that do not use the sentence order information versus the sentence length in Figure 4.

In the figure, we first observe that **Ours (K=100)** significantly outperforms **W Emb (GloVe)** and **Sent Emb (GloVe)** when summaries have similar length. In addition, we find that **W Emb (*)** usually outperforms **Sent Emb (*)** when comparing the summaries with a similar length. Notice that this comparison might not be fair because **W Emb (*)** are allowed to select more sentences given the same length of summary and it might be easier to cover more topics in the document using more sentences. In practice, preventing choosing many short sentences might be preferable in an extractive summarization if fluency is an important factor.

Nevertheless, suppose our goal is simply to maximize the ROUGE F1 score given a fixed length of the summary without accessing the ground truth summary and sentence order information. In that case, the figure indicates that **Ours (K=100)** significantly outperform **W Emb (GloVe)** and is the best choice when the summary length is less than around 50 words and **W Emb (BERT)** becomes the best method for a longer summary. The BERT in this figure is the BERT base model. The mixed results suggest that combining our method with BERT might be a promising direction to get the best performance in this task (e.g., use contextualized word embedding from BERT as our pre-trained word embedding space).

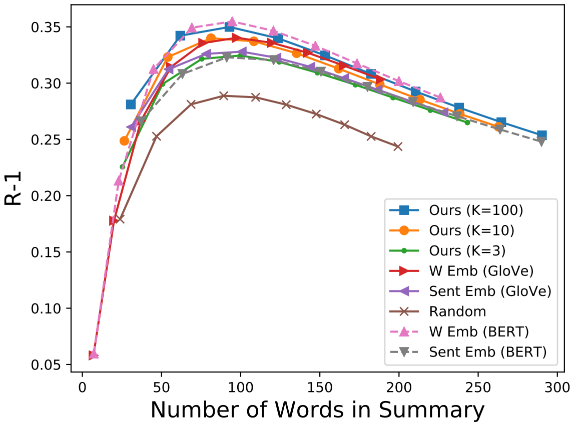

Figure 4: Comparing the F1 of ROUGE-1 score on unsupervised methods that do not access the sentence order information in CNN/Daily Mail.

B.4 Experiments on More Phrase Similarity Datasets

We conduct the phrase similarity experiments on two recently proposed datasets, BiRD (Asaadi, Mohammad, and Kiritchenko 2019), and WikiSRS (Newman-Griffis, Lai, and Fosler-Lussier 2018), which contain ground truth phrase similarities derived from human annotations. BiRD and WikiSRS-Rel measure the relatedness of phrases and WikiSRS-Sim measures the similarity of phrases. The phrases are proper nouns in WikiSRS and are mostly common nouns in BiRD. Since the main goal of WikiSRS is to test the entity representation, we also test the different models trained on the corpus without lowercasing all the words.

The results are presented in Table 9. The multi-facet embedding performs similarly compared with single-facet embedding and is better than other baselines. This result confirms our findings in the main paper that the phrase similarity performance is not sensitive to the number of clusters K.

B.5 Comparison with BERT Large

In Table 12, we compare the size and running time of different models for sentence representation. As mentioned in Section 3.1, our model has fewer parameters than the BERT base model and uses much fewer computational resources for training, so we only present the BERT Base performance in the experiment sections. Nevertheless, we still wonder how well BERT large can perform in these unsupervised semantic tasks, so we compare our method with BERT Large in Table 13, Table 14, Table 15, Table 16. As we can see, BERT large is usually better than BERT base in the similarity tasks but performs worse in the hypernym detection task. The BERT's performance gains in similarity tasks might imply that training a larger version of our model might be a promising future direction.

		Lowercase			Uppercase	
Method		BiRD	WikiSRS-Sim	WikiSRS-Rel	WikiSRS-Sim	WikiSRS-Rel
Model	Score	Pearson	Spearman			
ELMo	Avg	-	-	-	54.4	49.2
BERT	Avg	44.4	43.3	40.7	40.3	37.8
GloVe	Avg	47.9[5]	47.7	49.1	62.7	62.7
Ours (K=10)	Emb	57.3	**67.4**	**66.9**	68.6	**69.6**
Ours (K=1)	Emb	**60.6**	67.1	65.8	**69.4**	69.4

Table 9: Performances of phrase similarity tasks. In BiRD and WikiSRS, the correlation coefficient (%) between the predicted similarity and the ground truth similarity is presented.

Similarity			Summarization
STSB All		STS2012-6 All	CNN DM
Dev	Test	Test	Dev
1,500	1,379	12,544	11,490

Table 10: Dataset sizes for sentence representations.

Similarity					Hypernym	
SemEval 2013	Turney2012	BiRD	WikiSRS		HypeNet	
Test	Test	Test	Sim	Rel	Val	Test
7,814	1,500	3,345	688	688	3,534	17,670

Table 11: Dataset sizes for phrase representations.

Method	Hidden size	#Parameters	Testing Time
K=1	300	6.7M	9 ms
K=10	300	13.7M	18 ms
BERT Base	768	86.0M	18 ms
BERT Large	1024	303.9M	65 ms

Table 12: Comparison of model sizes. The number of parameters does not include the word embedding layer. We show the test time required for a batch with 50 sentences using one 1080Ti GPU.

B.6 Motivating Examples in Sentence Similarity

In order to further understand when and why our methods perform well, we present some sentences pairs from the MSRvid dataset in STS 2012 in Table 17 and 18 on which our methods perform well.

In Table 17, the first two sentence pairs have relatively high similarities but a lower ratio of overlapping words, so the baseline based on average word embedding (i.e., **Avg**) underestimates the similarities. Softly removing the stop words (i.e., **Prob_avg**) alleviates the problem, but the inverse frequency of words do not completely align with the importance of words in the sentences.

We visualize our predicted word importance and codebook embeddings in Table 18. Combining the estimated word importance with the inverse word frequency (i.e., **Prob_avg + Our a**) improves the performance. Finally, computing the similarity between the codebook embeddings (i.e., **Our c**) leads to the best results. The reason of the improvement might be that the unimportant words in the sentence often do not significantly affect the co-occuring word distribution. Take the second sentence pair as an example, mentioning *"with the big eyes"* does not change the sentence's meaning and facets too much.

On the contrary, the last sentence pair in Table 17 has a low similarity but relatively higher word overlapping. Our model could infer that *riding a horse* is very different from *riding an elephant* because their co-occurring word distributions are different. The appearance of *riding a horse* implies that we are more likely to observe a race topic in nearby sentences, but *riding an elephant* increases the chance of seeing a movie topic instead.

B.7 Motivating Examples in Extractive Summarization

In Table 19, we show the top three sentences that different methods choose to summarize a story about a photographer, Erik Johansson, and his artwork.

In this document, **Lead-3** does not cover its main points because this article starts with a preamble. Our method selects the first sentence as a good summary because it highlights the main character of the story, Erik Johansson, and his art style. The selected sentences contain the aspects that cover several topics in the whole document.

Average word embedding baselines, **Sent_Emb (GloVe)** and **Sent_Emb (BERT)**, select the sentences that focus on describing how his artwork is created. Nevertheless, the sentences are hard to understand without the context in the article. We hypothesize that the methods tend to avoid selecting the sentences with diverse aspects because after averaging the word embeddings, the resulting single embedding is not close to the embedding of words in the documents.

Finally, **W_Emb (GloVe)** and **W_Emb (BERT)** tend to select shorter sentences because we normalize the objective function by the sentence lengths. It is hard to remove the bias of selecting shorter or longer sentences because each sentence is represented by a different number of embeddings.

C Experimental Details

C.1 Training

The training algorithm of non-negative sparse coding (NNSC) loss can be seen in Algorithm 1. Given the computational resource constraints, we keep our model simple

[5] The number is different from the one reported in Asaadi, Mohammad, and Kiritchenko (2019) because we use the uncased version (42B), the embedding space our model is trained on, and they use the cased version (840B).

Method		Dev		Test	
Model	Score	All	Low	All	Low
BERT Base	Prob_avg	72.1	57.0	57.8	55.1
BERT Large	Prob_avg	74.3	61.0	65.0	**60.0**
Our a (GloVe) K10	Prob_avg	72.0	60.5	61.4	59.3
	Prob_WMD	**76.2**	**62.6**	**66.1**	58.1

Table 13: Compare BERT Large with Ours in Table 2.

Method		R-1	R-2	Len
BERT Base	W Emb	31.2	11.2	44.9
	Sent Emb	32.3	10.6	91.2
BERT Large	W Emb	31.1	11.0	46.8
	Sent Emb	32.7	10.9	86.5
Our c (K=100)	Bases	**35.0**	**12.8**	92.9

Table 14: Compare BERT Large with Ours in Table 3.

Method		Lowercased							Uppercased	
		SemEval		Turney (5)	Turney (10)	BiRD	WikiSRS-Sim	WikiSRS-Rel	WikiSRS-Sim	WikiSRS-Rel
Model	Score	AUC	F1	Acc	Acc	Pearson	Spearman			
BERT Base	Avg	66.5	67.1	43.4	24.3	44.4	43.3	40.7	40.3	37.8
BERT Large	Avg	72.4	66.7	**51.3**	32.1	47.5	49.6	48.1	28.6	34.0
Ours (K=1)	Emb	**87.8**	**78.6**	50.3	**32.5**	**60.6**	**67.1**	**65.8**	**69.4**	**69.4**

Table 15: Compare BERT Large with Ours in Table 4.

Method	Dev		Test	
	AUC	Acc	AUC	Acc
BERT Base (Avg)	25.6	50	25.6	50
BERT Large (Avg)	20.2	50	20.1	50
Ours (K=1)	**29.3**	**82.7**	**29.6**	**81.0**

Table 16: Compare BERT Large with Ours in Table 5.

enough to have the training loss nearly converged after 1 or 2 epoch(s). Since training takes a long time, we do not fine-tune the hyper-parameters in our models. We use a much smaller model than BERT but the architecture details in our transformer and most of its hyper-parameters are the same as those used in BERT.

The sparsity penalty weights on coefficient matrix λ in equation 1 is set to be 0.4. The maximal sentence size is set to be 50, and we ignore the sentences longer than that. The maximal number of co-occurring words is set to be 30 (after removing the stop words), and we sub-sample the words if there are more words in the previous and next sentence. All words occurring less than 100 times in the training corpus are mapped to <unk>.

The number of dimensions in transformers is set to be 300. For sentence representation, dropout on attention is 0.1. Its number of transformer layers on the decoder side is 5 for $K = 10$, and the number of transformer layers on the decoder side is set to be 1 for $K = 1$ because we do not need to model the dependency of output codebook embeddings. For phrase representation, the number of transformer layers on the decoder side is 2, and the dropout on attention is 0.5.

All the architecture and hyperparameters (except the number of codebook embeddings) in our models are determined by the validation loss of the self-supervised co-occurring word reconstruction task in equation 2 . The number of codebook embeddings K is chosen by the performance of training data in each task, but we observe that the performances are usually not sensitive to the numbers as long as K is large enough as shown in our phrase experiments. Furthermore, we suspect that the slight performance drops of models with too large K might just be caused by the fact that larger K needs longer training time and 1 week of training is insufficient to make the model converge.

We use RegexpParser in NLTK (Bird, Klein, and Loper 2009) to detect the phrase boundary. We use the grammar *NP: <JJ.*>*<VBG>*<NN.*>+*. The sentence boundaries are detected using the rule-based pipeline in spaCy[6] and POS tags are also detected using spaCy.

The lowercased list we use for removing stop words includes @-@, =, <*eos*>, <*unk*>, *disambiguation, etc, etc.,* –, @*card*@, ∼, -, _, @, ˆ, &, *, <, >, (,), \, |, {, },], [, :, ;, ', '', /, ?, !, „ ., 't, 'd, 'll, 's, 'm, 've, a, about, above, after, again, against, all, am, an, and, any, are, aren, as, at, be, because, been, before, being, below, between, both, but, by, can, cannot, could, couldn, did, didn, do, does, doesn, doing, don, down, during, each, few, for, from, further, had, hadn, has, hasn, have, haven, having, he, her, here, here, hers, herself, him, himself, his, how, how, i, if, in, into, is, isn, it, it, its, itself, let, me, more, most, mustn, my, myself, no, nor, not, of, off, on, once, only, or, other, ought, our, ours, ourselves, out, over, own, same, she, should, shouldn, so, some, such, than, that, the, their, theirs, them, themselves, then, there, these, they, this, those, through, to, too, under, until, up, very, was, wasn, we, were, weren, what, when, where, which, while, who, whom, why, with, won, would, wouldn, you, your, yours, yourself, yourselves*.

C.2 Testing

The dataset sizes for sentence representation and phrase representation are summarized in Table 10 and Table 11, respectively. In our phrase experiments, we report the test sets of SemEval 2013 and Turney. For Turney dataset, we follow the evaluation setup of Yu and Dredze (2015); Huang, Ji et al. (2017), which ignores two unigram candidates being contained in the target phrase, because the original setup (Turney 2012) is too difficult for unsupervised methods to get a meaningful score (e.g., the accuracy of **GloVe Avg** is 0 in the original setting).

For skip-thoughts, the hidden embedding size is set to be 600. To make the comparison fair, we retrain the skip-

[6] spacy.io/

Sentence 1	Sentence 2	Score	Score Rank Among 1500 Pairs				
		GT	GT	Our c	Prob_avg + Our a	Prob_avg	Avg
A turtle walks over the ground .	A large turtle crawls in the grass .	3.75	326	400	638	717	761
The animal with the big eyes is eating .	A slow loris is eating .	2.60	690	611	1001	1223	1370
A man is riding on a horse .	A woman is riding an elephant .	1.53	1021	869	722	549	540

Table 17: Motivating examples for a sentence similarity task. The sentences are image captions from MSRvid dataset in STS 2012. GT means ground truth. All our methods here set $K = 10$.

	Sentence 1	Sentence 2
Sentences	A turtle walks over the ground .	A large turtle crawls in the grass .
Output Embeddings	can 0.876 you 0.846 even 0.845 turtle 0.914 turtles 0.822 dolphin 0.755 hillside 0.737 hills 0.711 mountains 0.704 species 0.937 habitat 0.771 habitats 0.759 animals 0.689 pigs 0.675 animal 0.658 white 0.842 blue 0.839 red 0.833 insectivorous 0.650 immatures 0.627 insectivores 0.618 ground 0.693 soil 0.676 surface 0.626 ground 0.861 grass 0.598 Ground 0.576 waking 0.551 strolls 0.551 wander 0.546	can 0.857 even 0.838 often 0.832 turtle 0.917 turtles 0.818 dolphin 0.755 beneath 0.697 hillside 0.645 down 0.639 species 0.949 habitat 0.778 habitats 0.760 females 0.655 males 0.630 male 0.622 white 0.863 blue 0.862 red 0.856 immatures 0.616 foliose 0.609 tussocks 0.607 seawater 0.644 salinity 0.597 soils 0.593 grass 0.870 grasses 0.739 weeds 0.713 length 0.732 width 0.639 diameter 0.627
Sentences	The animal with the big eyes is eating .	A slow loris is eating .
Output Embeddings	even 0.887 sure 0.877 want 0.868 animals 0.896 animal 0.896 rabbits 0.678 food 0.825 foods 0.802 eating 0.798 fingers 0.695 legs 0.692 shoulders 0.691 species 0.919 habitats 0.738 habitat 0.731 blue 0.834 red 0.809 white 0.805 ingestion 0.608 inflammation 0.591 concentrations 0.588 male 0.652 female 0.636 disease 0.618 profound 0.668 perceived 0.647 profoundly 0.634 beady 0.626 beaks 0.623 mandibles 0.602	often 0.880 usually 0.860 sometimes 0.838 loris 0.864 lorises 0.670 langur 0.596 foods 0.763 food 0.709 nutritional 0.690 eating 0.799 food 0.798 eat 0.794 species 0.949 habitat 0.787 habitats 0.769 blue 0.844 white 0.839 red 0.827 gently 0.649 wet 0.642 beneath 0.641 male 0.685 female 0.659 females 0.658 decreasing 0.710 decreases 0.699 decrease 0.697 C. 0.624 L. 0.620 A. 0.593
Sentences	A man is riding on a horse .	A woman is riding an elephant .
Output Embeddings	sure 0.883 even 0.883 want 0.867 slid 0.686 legs 0.681 shoulders 0.670 moisture 0.518 drying 0.517 coated 0.516 horse 0.917 horses 0.878 stallion 0.728 mortals 0.668 fearful 0.664 beings 0.646 man 0.864 woman 0.764 boy 0.700 man 0.858 woman 0.652 boy 0.637 discovers 0.647 informs 0.638 learns 0.634 race 0.812 races 0.800 championship 0.697 Horse 0.763 Riding 0.696 Horses 0.656	sure 0.880 even 0.876 know 0.866 underneath 0.701 shoulders 0.671 legs 0.668 elephant 0.912 elephants 0.842 hippo 0.759 elephant 0.885 elephants 0.817 animals 0.728 fearful 0.677 disdain 0.637 anguish 0.632 woman 0.822 women 0.787 female 0.718 girl 0.784 woman 0.781 lady 0.721 discovers 0.662 learns 0.659 realizes 0.648 movie 0.622 film 0.615 films 0.590 riding 0.873 bike 0.740 biking 0.731

Table 18: The predicted word importance and codebook embeddings on sentences from Table 17. The way of visualization is the same as that in Section D.

```
Algorithm 1: Training using NNSC loss
Input  : Training corpus, sequence boundaries, and pre-trained word embedding.
Output: $F$
Initialize $F$
foreach $I_t, \boldsymbol{W}(\boldsymbol{N_t}), \boldsymbol{W}(\boldsymbol{N_{r_t}})$ in training corpus do
    Run forward pass on encoder and decoder to compute $\boldsymbol{F}(\boldsymbol{I_t})$
    Compute $\boldsymbol{M}^{\boldsymbol{O_t}} = \arg\min_{\boldsymbol{M}} ||\boldsymbol{F}(\boldsymbol{I_t})\boldsymbol{M} - \boldsymbol{W}(\boldsymbol{N_t})||^2 + \lambda||\boldsymbol{M}||_1 \forall k, j,\ 0 \leq \boldsymbol{M}_{k,j} \leq 1,$
    Compute $\boldsymbol{M}^{\boldsymbol{R_t}} = \arg\min_{\boldsymbol{M}} ||\boldsymbol{F}(\boldsymbol{I_t})\boldsymbol{M} - \boldsymbol{W}(\boldsymbol{N_{r_t}})||^2 + \lambda||\boldsymbol{M}||_1 \forall k, j,\ 0 \leq \boldsymbol{M}_{k,j} \leq 1,$
    Run forward pass to compute $L_t$ in equation 2
    Treat $\boldsymbol{M}^{\boldsymbol{O_t}}$ and $\boldsymbol{M}^{\boldsymbol{R_t}}$ as constants, update $F$ by backpropagation
end
```

Method	Index	Selected Sentence
Ground Truth	NA NA NA	Swedish photographer , Erik Johansson , spends months photographing images to build up to the finished picture . Each image is made up of hundreds of separate shots and painstakingly detailed work by the expert retoucher . Erik , 30 , said : ' Can I put this very weird idea in a photograph and make it look like it was just captured ? '
Lead-3	1 2 3	Thought the black and blue dress was an optical illusion ? It 's nothing compared to these mind - boggling pictures by a Swedish photographer , artist , and Photoshop genius . Erik Johansson , 30 , who is based in Berlin , Germany , says he does n't capture moments , but instead captures ideas .
Our c (K=10)	6 42 1	Swedish photographer , artist , and Photoshop genius , Erik Johansson , has created mind - boggling photos like this inside - out house that look different on each glance . Reverse Opposite is mind - bendig as , with an MC Escher drawing , the car seems both on and under the bridge at the same time . Thought the black and blue dress was an optical illusion ?
Sent_Emb (GloVe)	46 25 18	Although one photo can consist of lots of different images merged into one , he always wants it to look like it could have been captured as a whole picture . He cites Rene Magritte , Salvador Dali and MC Escher as artistic influences . Using Photoshop , he turned the running paint into rolling fields and superimposed a photograph of a house on to the cardboard model , adding a photo of a water wheel to complete the fantastical and dramatic shot of a dreamy , bucolic landscape that seems to be falling over a cliff .
W_Emb (GloVe)	41 23 22	he said . But there are tons of inspiration online . ' I think I get more inspiration from paintings rather than photos .
Sent_Emb (BERT)	18 41 12	Using Photoshop , he turned the running paint into rolling fields and superimposed a photograph of a house on to the cardboard model , adding a photo of a water wheel to complete the fantastical and dramatic shot of a dreamy , bucolic landscape that seems to be falling over a cliff . he said . He said : ' It 's the challenge : can I put this very weird idea in a photograph and make it look like it was just captured ? '
W_Emb (BERT)	5 30 12	Scroll down for video . In Closing Out , interiors and exterior meld as one in this seemingly simple tableau . He said : ' It 's the challenge : can I put this very weird idea in a photograph and make it look like it was just captured ? '

Table 19: Motivating examples for extractive summarization. The sentences come from a document in the validation set of CNN/Daily Mail. Index indicates the sentence order in the document. Ground truth means the summary from humans. The sentences in each method are ranked by its selection order. For example, our method selects the 6th sentence in the document first.

thoughts in Wikipedia 2016 for 2 weeks.

D Randomly Sampled Examples

We visualize the predicted codebook embeddings and the attention weights computed using equation 6 from 10 randomly selected sentences in our validation set (so most of them are unseen in our training corpus).

The first line of each example is always the preprocessed input sentence, where <unk> means an out-of-vocabulary placeholder. The attention weights are visualized using a red background. If one word is more likely to be similar to the words in the nearby sentences, it will get more attention and thus highlighted using a darker red color.

The format of visualized embeddings is similar to Table 1. Each row's embedding is visualized by the nearest five neighbors in a GloVe embedding space and their cosine similarities to the codebook embedding.

Other immobilizing devices such as a Kendrick <unk> Device or a backboard can be used to stabilize the remainder of the spinal column .) <eos>

——————K=10——————
use 0.844 can 0.816 used 0.801
bottom 0.757 front 0.703 sides 0.691
spinal 0.914 nerve 0.739 Spinal 0.700
increases 0.766 decreasing 0.756 increasing 0.749
devices 0.836 device 0.787 wireless 0.703
symptoms 0.726 chronic 0.715 disease 0.692
polymeric 0.674 hydrophilic 0.644 hydrophobic 0.636
backboard 0.899 hoop 0.555 dunks 0.547
column 0.898 columns 0.771 Column 0.584
Kendrick 0.927 Lamar 0.620 Meek 0.611
——————K=3——————
necessary 0.750 use 0.736 can 0.732
spinal 0.801 thoracic 0.713 nerve 0.702
backboard 0.771 ball 0.629 hoop 0.593
——————K=1——————
tissue 0.667 prevent 0.659 pressure 0.642

When she came in , she was always bound to be loud , and boisterous . Carrie got along well with most of the waitresses , most especially Vera Louise Gorman - Novak , and Alice Hyatt . <eos>

——————K=10——————
really 0.893 know 0.877 think 0.869
Carrie 0.901 Christina 0.727 Amanda 0.723
daughter 0.816 mother 0.815 wife 0.781
Carrie 0.908 Christina 0.748 Amanda 0.739
came 0.730 went 0.722 had 0.711
Maureen 0.750 Carolyn 0.749 Joanne 0.740
endearing 0.670 downright 0.643 demeanor 0.632
Vera 0.953 Aloe 0.639 vera 0.585
actor 0.672 starring 0.650 comedy 0.639
dancing 0.647 singing 0.598 raucous 0.593
——————K=3——————
really 0.857 thought 0.853 never 0.846
Carrie 0.881 Amanda 0.801 Rebecca 0.792

waitress 0.596 hostess 0.595 maid 0.591
————————K=1————————
knew 0.806 she 0.793 thought 0.788

The station building is located in the district of <unk> . These services operate on the Eifel Railway (<unk>) . <eos>

————————K=10————————
station 0.989 stations 0.847 Station 0.756
Eifel 0.855 Harz 0.673 Cochem 0.642
railway 0.820 railways 0.794 trains 0.782
Germany 0.813 Berlin 0.766 Munich 0.751
north 0.885 south 0.877 east 0.869
building 0.867 buildings 0.794 construction 0.706
located 0.696 operated 0.669 operates 0.668
line 0.877 lines 0.739 Line 0.712
Railway 0.751 Rail 0.622 Railways 0.591
services 0.909 service 0.879 provider 0.708
————————K=3————————
station 0.918 stations 0.807 railway 0.779
located 0.802 area 0.770 situated 0.734
Aachen 0.706 Eifel 0.701 Freiburg 0.661
————————K=1————————
station 0.853 railway 0.834 stations 0.748

Alfred <unk> <unk> (born January 22 , 1965) is a Ghanaian businessman and a former Honorary Vice Consul of Austria to Ghana and a leading member of the National Democratic Congress . <eos>

————————K=10————————
Ghana 0.928 Zambia 0.807 Cameroon 0.796
Committee 0.824 Council 0.740 Commission 0.733
election 0.810 elections 0.788 elected 0.774
Austria 0.923 Germany 0.793 Austrian 0.739
February 0.868 2011 0.865 2012 0.858
Consul 0.898 consul 0.761 consular 0.596
1995 0.970 1994 0.970 1993 0.969
Party 0.886 party 0.707 Parties 0.649
University 0.858 College 0.728 Graduate 0.727
retired 0.575 born 0.572 emeritus 0.543
————————K=3————————
President 0.778 Chairman 0.765 Committee 0.760
Ghana 0.924 Zambia 0.808 Cameroon 0.806
1999 0.956 1998 0.953 1997 0.951
————————K=1————————
President 0.723 Affairs 0.664 Minister 0.658

ISBN 0 - 8063 - 1367 - 6 Winthrop , John . Winthrop 's Journal , History of New England 1630 - 1649 . New York , NY : Charles Scribner 's Sons , 1908 . <eos>

————————K=10————————
New 0.936 York 0.915 NY 0.807
J. 0.847 R. 0.810 D. 0.807
ISBN 0.934 Paperback 0.821 Hardcover 0.803
History 0.830 Historical 0.694 War 0.681
0 0.933 1 0.739 3 0.663
Winthrop 0.926 Endicott 0.696 Amherst 0.672
1985 0.840 1981 0.836 1982 0.834
University 0.901 College 0.714 Northwestern 0.692
England 0.875 London 0.718 Britain 0.680
842 0.838 794 0.835 782 0.831
————————K=3————————
William 0.841 J. 0.789 Robert 0.779
New 0.887 York 0.860 NY 0.727
1626 0.685 1684 0.676 1628 0.675
————————K=1————————
William 0.686 York 0.676 Charles 0.643

The commune is represented in the Senate by Soledad Alvear (PDC) and Pablo <unk> (UDI) as part of the 8th senatorial constituency (Santiago - East) . Haha Sound is the second album by the British indie electronic band Broadcast . <eos>

————————K=10————————
album 0.919 albums 0.844 songs 0.830
constituency 0.810 election 0.773 elections 0.757
released 0.931 release 0.847 releases 0.783
Santiago 0.859 Juan 0.753 Luis 0.738
Sound 0.890 Audio 0.739 Sounds 0.671
February 0.881 2011 0.878 2010 0.877
Jorge 0.687 Miguel 0.684 Pablo 0.683
Alvear 0.738 Altamirano 0.563 Ruperto 0.552
commune 0.939 communes 0.800 Commune 0.660
# 0.986 Item 0.523 1 0.507
————————K=3————————
commune 0.718 communes 0.638 La 0.596
album 0.948 albums 0.865 songs 0.796
election 0.814 elections 0.791 electoral 0.745
————————K=1————————
album 0.925 albums 0.818 Album 0.745

As a result , the Hellfire Club believed that it would be in their best interests to summon the Phoenix and merge it with Jean Grey via a ritual . <eos>

————————K=10————————
want 0.886 way 0.875 sure 0.871
attack 0.722 kill 0.708 enemy 0.708
beings 0.703 manifestation 0.690 spiritual 0.685
disappeared 0.710 apparently 0.681 initially 0.678
Evil 0.719 Darkness 0.694 Demon 0.693
Phoenix 0.950 Tucson 0.688 Tempe 0.658
Hellfire 0.932 hellfire 0.623 Brimstone 0.530
Grey 0.956 Gray 0.788 Blue 0.771
Jean 0.912 Pierre 0.767 Jacques 0.736
Club 0.903 club 0.804 clubs 0.680

————————K=3————————

even 0.836 somehow 0.823 because 0.816
Hellfire 0.814 Scourge 0.606 Goblin 0.588
Blue 0.738 Grey 0.730 Red 0.693

————————K=1————————

somehow 0.780 even 0.769 nothing 0.747

Skeletal problems , infection , and tumors can also affect the growth of the leg , sometimes giving rise to a one - sided bow - <unk> . <eos>

————————K=10————————

symptoms 0.851 disease 0.826 chronic 0.818
often 0.857 usually 0.854 may 0.851
infection 0.849 infections 0.807 infected 0.772
tumors 0.867 tumours 0.834 tumor 0.791
epithelial 0.676 epithelium 0.672 upregulation 0.667
increasing 0.805 increased 0.799 increases 0.798
legs 0.737 shoulders 0.716 fingers 0.715
leg 0.909 legs 0.785 thigh 0.750
mammalian 0.666 gene 0.664 genes 0.654
bow 0.899 bows 0.775 sash 0.574

————————K=3————————

affect 0.769 significant 0.755 decrease 0.741
disease 0.795 infection 0.782 infections 0.757
epithelial 0.691 extracellular 0.686 epithelium 0.685

————————K=1————————

disease 0.764 abnormal 0.751 tissue 0.743

<unk> is composed of the : Jack Carty may refer to : Jack Carty (musician) (born 1987) , Australian musician Jack Carty (rugby union) (born 1992) , rugby union player from Ireland John Carty (disambiguation) <eos>

————————K=10————————

February 0.910 April 0.906 June 0.904
States 0.746 American 0.745 United 0.735
Australian 0.848 Zealand 0.749 Australia 0.729
born 0.880 Born 0.686 married 0.596
band 0.598 album 0.596 music 0.583
Carty 0.914 Kelleher 0.621 O'Mahony 0.619
1977 0.967 1978 0.964 1974 0.963
football 0.786 player 0.740 soccer 0.740
union 0.841 unions 0.698 Unions 0.631
Jack 0.800 Jim 0.727 Tom 0.715

————————K=3————————

1975 0.740 1981 0.738 1978 0.736
Ireland 0.910 Irish 0.782 Dublin 0.747
rugby 0.766 football 0.759 soccer 0.718

————————K=1————————

England 0.609 Rugby 0.600 Wales 0.596

He designed a system of streets which generally followed the contours of the area 's topography . Residential neighborhoods stretched out from a commercial and service - sector core . <eos>

————————K=10————————

creating 0.737 incorporate 0.726 developing 0.720
north 0.861 south 0.859 east 0.855
sector 0.874 sectors 0.790 economic 0.746
neighborhoods 0.840 streets 0.775 neighbourhoods 0.749
City 0.745 Riverside 0.726 Heights 0.709
buildings 0.691 brick 0.672 building 0.658
topography 0.817 vegetation 0.708 forested 0.702
service 0.889 services 0.809 Service 0.663
Building 0.742 Construction 0.613 Project 0.581
1930 0.856 1920s 0.826 1929 0.815

————————K=3————————

development 0.739 significant 0.720 providing 0.719
streets 0.831 city 0.793 neighborhoods 0.786
County 0.669 Riverside 0.662 City 0.649

————————K=1————————

buildings 0.765 area 0.761 areas 0.753