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.

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.

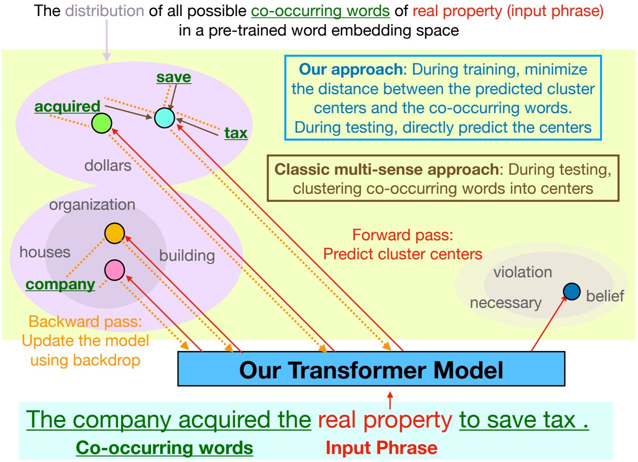
The distribution of all possible co-occurring words of real property (input phrase)
in a pre-trained word embedding space
save
acquired
tax
dollars
organization
houses
building
company
Our approach: During training, minimize the distance between the predicted cluster centers and the co-occurring words. During testing, directly predict the centers
Classic multi-sense approach: During testing, clustering co-occurring words into centers
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 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.

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.

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 the appendix, 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).

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.

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.

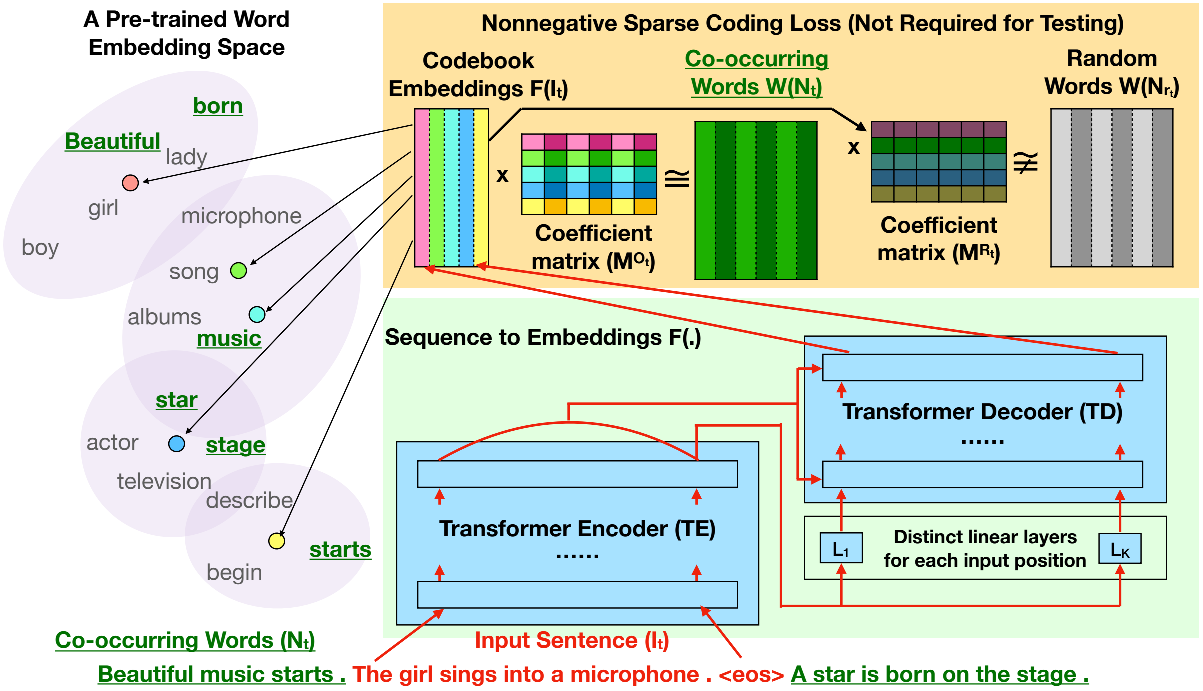
A Pre-trained Word Embedding Space
born
Beautiful
lady
girl
microphone
boy
song
albums
music
star
actor
stage
television
describe
begin
starts
Co-occurring Words (N$_t$)
Nonnegative Sparse Coding Loss (Not Required for Testing)
Codebook Embeddings F(I$_t$)
Co-occurring Words W(N$_t$)
Random Words W(N$_{r_t}$)
x
≅
x
≇
Coefficient matrix (M^{O_t})
Coefficient matrix (M^{R_t})
Sequence to Embeddings F(.)
Transformer Decoder (TD)
......
Transformer Encoder (TE)
......
L$_1$
Distinct linear layers for each input position
L$_K$
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.

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 $\boldsymbol{M}^{\boldsymbol{O_t}}$ after fixing our neural model, while our M-step treats $\boldsymbol{M}^{\boldsymbol{O_t}}$ 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 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, $\boldsymbol{M}^{\boldsymbol{O_t}}$ can be efficiently estimated using convex optimization (our implementation uses RMSprop (Tieleman and Hinton 2012)). Besides, we minimize the L2 distance, $||\boldsymbol{F(I_t)}\boldsymbol{M}^{\boldsymbol{O_t}} - \boldsymbol{W(N_t)}||^2$, 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.

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.

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.

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 2018).

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,

[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.

and their boundaries are extracted by applying simple regular expression rules to POS tags before training. We use 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).

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 the appendix.

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.

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**:

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

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

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

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 the appendix.

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 the appendix, 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.

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).

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 the appendix, the results on hypernym detection 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.

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.

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

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*.

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*.

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*.

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*.

Pagliardini, M.; Gupta, P.; and Jaggi, M. 2018. Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. In *NAACL-HLT*, 528–540.

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*.

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*.

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*.