Unsupervised Attention-based Sentence-Level Meta-Embeddings from Contextualised Language Models

Keigo Takahashi, Danushka Bollegala

Tokyo Metropolitan University, University of Liverpool

takahashi-keigo@ed.tmu.ac.jp, danushka@liverpool.ac.uk

Abstract

A variety of contextualised language models have been proposed in the NLP community, which are trained on diverse corpora to produce numerous Neural Language Models (NLMs). However, different NLMs have reported different levels of performances in downstream NLP applications when used as text representations. We propose a sentence-level meta-embedding learning method that takes independently trained contextualised word embedding models and learns a sentence embedding that preserves the complementary strengths of the input source NLMs. Our proposed method is unsupervised and is not tied to a particular downstream task, which makes the learnt meta-embeddings in principle applicable to different tasks that require sentence representations. Specifically, we first project the token-level embeddings obtained by the individual NLMs and learn attention weights that indicate the contributions of source embeddings towards their token-level meta-embeddings. Next, we apply mean and max pooling to produce sentence-level meta-embeddings from token-level meta-embeddings. Experimental results on semantic textual similarity benchmarks show that our proposed unsupervised sentence-level meta-embedding method outperforms previously proposed sentence-level meta-embedding methods as well as a supervised baseline.

1. Introduction

Meta-embedding (Yin and Schütze, 2016a; Bollegala et al., 2018) is the task of producing embeddings from a given set of pretrained text embeddings such that the complementary strengths of the input (source) embeddings are preserved or further enhanced to obtain the best performance on a downstream NLP task. Numerous methods have been proposed for creating a meta-embedding from a given set of static word embeddings such as GloVe (Pennington et al., 2014), Skip-gram with Negative Sampling (SGNS) (Mikolov et al., 2013b) or Continuous Bag-of-Words (CBOW) (Mikolov et al., 2013a) as the source embeddings. In contrast to static word embeddings, contextualised word embeddings obtained from Neural Language Models (NLMs) such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) etc. represent the same word with different vectors in different contexts, thereby providing a richer context-dependent semantic representations. Moreover, contextualised word embeddings can be pooled and fine-tuned to obtain accurate sentence-level representations. Compared to static word embeddings, contextualised word/sentence embeddings have reported impressive performances in a wide range of NLP tasks.

Challenges: Compared to meta-embedding of static word embeddings, when computing meta-embeddings from contextualised word embeddings (Poerner et al., 2020) one must overcome several challenges.

- First, unlike static word embeddings, which represent the same word using the same embedding vector in all contexts in which the target word occurs, contextualised word embeddings depend on the context of the word. Therefore, the same word will be represented by different embeddings in different contexts by a contextualised word embedding model. This is particularly challenging when creating meta word embeddings because the meta embedding of a word will depend not only on the source embeddings, but also on the context in which the target word occurs.

- Second, static word embeddings represent words using relatively smaller dimensional vector spaces compared to contextualised word embeddings. For example, good performance can be obtained with as small as 300-600 dimensional GloVe static word embeddings, whereas BERT contextualised word embeddings require 768, 1024 or higher dimensionalities. Therefore, methods such as concatenation that have been effective for creating meta-embeddings (Coates and Bollegala, 2018a) for static word embeddings become problematic when applied to contextualised word embeddings. This issue is further aggravated by the fact that contextualised word embedding models require specialised hardware such as GPUs, which have limited memory buffers.

- Third, although supervised approaches have been proposed for meta-embedding of static word embeddings that require labelled data for the target tasks (Xie et al., 2019; He et al., 2020a; Wu et al., 2020), adapting such supervised approaches for contextualised word embeddings remains a challenging task. Because contextualised word embeddings such as BERT typically contain over 100 million parameters[1], it requires significantly more labelled data to fine-tune such models for a

---

[1]For example, the `BERT-small` model trained by Devlin et al. (2019) with 12 layers contain 110 million parameters

given target task. Therefore, unsupervised methods are preferable over supervised ones when creating meta-embeddings from contextualised embedding models.

Proposal: To address the above-mentioned challenges, we propose an unsupervised method to combine multiple contextualised word embedding models (referred to as **source embeddings**) to create a meta-embedding for a given target word in a sentence that it occurs. Furthermore, we create sentence-level embeddings by applying parameter-free operations such as *pooling* to create sentence-level meta-embeddings from the source embeddings.

In our proposed method, we first project token-level embeddings obtained for a word from a contextualised word embedding to a common meta-embedding space using linear transformations. We then compute the linearly-weighted average of the projected source embeddings to compute their token-level meta-embedding. The weighting coefficients can be seen as describing the significance we must impart on a source embedding in the ensemble. Note that contextualised token-embeddings produced by different NLMs can in general have different dimensionalities. Therefore, the source-specific linear projections learnt in the first step ensures all source embeddings are projected to a common meta-embedding dimensionality such that we can compute their linearly-weighted average. Finally, we apply pooling operations such as max and mean pooling over the token-level meta-embeddings for the words in a sentence to create a sentence-level meta-embedding for the corresponding sentence.

Contributions: We evaluate our proposed meta embedding learning method using different types of source embeddings on Semantic Textual Similarity (STS) benchmarks (Cer et al., 2017). We compare our results against multiple baselines such as vector concatenation, averaging, singular value decomposition and the current state-of-the-art unsupervised sentence-level meta-embedding method proposed by Poerner et al. (2020). Experimental results show that our proposed unsupervised sentence-level meta-embedding learning method obtains the highest correlation with human similarity ratings in all STS benchmark datasets, outperforming all methods compared.

2. Related Work

2.1. Sentence-Level Meta-embeddings

Sentence-level meta-embedding learning methods aim to create a single vector representation for a given sentence by combining token-level representations produced by multiple source embeddings. Kiela et al. (2018) proposed dynamic meta-embeddings, which learns task-specific combinations of source embeddings using the attention mechanism (Bahdanau et al., 2015). Poerner et al. (2020) proposed the use of Generalised Canonical Correlation Analysis (GCCA) to learn sentence-level meta-embeddings from contextualised word embeddings. GCCA is an extension of Canonocal Correlation Analysis (CCA) to more than two random variables. Specifically, a word in a sentence can be represented using multiple vectors by using multiple contextualised word embedding models. GCCA learns linear transformations that can be applied to such vectors to maximise the correlation in the projected space. Unlike our proposed method, prior work on sentence-level meta-embedding methods consider sentence embeddings as the input to the meta-embedding algorithms. In contrast, we first create token-level meta-embeddings by combining multiple contextualised embeddings and then create sentence-level meta-embeddings by applying pooling operations (e.g. max and min pooling) over the words in a sentence. GCCA-based meta sentence embedding method proposed by Poerner et al. (2020) is the current state-of-the-art unsupervised sentence-level meta-embedding learning method and is a direct comparison to ours which is also unsupervised method. As we later see in §4, our proposed method outperforms GCCA in multiple benchmark datasets.

2.2. Word-Level Meta-embeddings

Word meta-embedding has been studied more actively compared to sentence-level meta-embedding. Yin and Schütze (2016a) proposed 1TON, by projecting source embeddings to a common space via source-specific linear transformations. This method minimises squared ℓ_2 distance between the meta and each source embedding assuming a common vocabulary. 1TON+ overcomes this limitation by learning pairwise linear transformations between two given sources for predicting the embeddings for out of vocabulary (OOV) words. Both of these methods can be seen as *globally-linear* transformations because *all* the words in a particular source are projected to the meta-embedding space using the *same* transformation matrix.

In contrast, *locally-linear* meta-embedding (LLE) (Bollegala et al., 2018) computes the nearest neighbours for a particular word in each source and then represent a word as the linearly-weighted combination of its neighbours. Next, meta-embeddings are learnt by preserving the neighbourhood constraints. This method does not require all words to be represented by all sources, thereby obviating the need to predict missing source embeddings for OOVs.

Bao and Bollegala (2018) modelled meta-embedding learning as an *autoencoding* problem where information embedded in different sources are integrated at different levels to propose three types of meta-embeddings: Decoupled Autoencoded Meta-Embedding (DAEME) (independently encode each source and concatenate), Concatenated Autoencoded Meta-Embedding (CAEME) (independently decode meta-embeddings to reconstruct each source), and Averaged Autoencoded Meta-Embedding (AAEME) (similar to DAEME but instead of concatenation use average). In comparison to meth-

ods that learn globally or locally linear transformations (Bollegala et al., 2018; Yin and Schütze, 2016a), autoencoders learn nonlinear transformations. O'Neill and Bollegala (2018) further extend this approach using squared cosine proximity loss as the reconstruction loss. Vector concatenation has been used as a baseline for producing meta-embeddings (Bao and Bollegala, 2018; Yin and Schütze, 2016a; Bollegala et al., 2018). Goikoetxea et al. (2016) concatenated independently learnt word embeddings from a corpus and the WordNet. Moreover, applying Principal Component Analysis on the concatenation further improved their performance on similarity tasks. Interestingly, Coates and Bollegala (2018b) showed that accurate meta-embeddings can be produced by averaging source embeddings that exist in *different* vector spaces. Specifically, they showed that when word embeddings in each source are approximately orthogonal (they empirically validate for pre-trained embeddings), averaging can approximate concatenation. Although averaging does not increase the dimensionality of the meta-embedding space as with concatenation, it does not consistently outperform concatenation, especially when the orthogonality condition does not hold. Recent work (He et al., 2020b; Jawanpuria et al., 2020) show that learning orthogonal transformations prior to averaging can further improve accuracy.

3. Method

Let us denote a sentence $s = w_1, w_2, \ldots, w_{|s|}$ consisting of tokens $w_1, w_2, \ldots, w_{|s|}$ in that order. In the case where we have applied a subtokenisation method such as Byte Pair Encoding Sennrich et al. (2016) to tokenise sentences into subtokens, we will assume w_j to be subtokens instead of tokens. In the following discussion, we use the term *word* to cover both tokens as well as subtokens unless otherwise explicitly stated. Moreover, let us assume that we are given n independently pretrained contextualised word embeddings of which the i-th source embedding is denoted by $\boldsymbol{f}_i(w, s) \in \mathbb{R}^{d_i}$, and returns a d_i dimensional word embedding of w in a given context s (e.g. a sentence or a contextual window). In particular, static word embeddings can be seen as a special case of this formulation where the word embedding does not depend on s (i.e. $\forall s \ \boldsymbol{f}_i(w, s) = \boldsymbol{f}_i(w)$). Our goal in this paper is to create a sentence-level meta-embedding, $\boldsymbol{m}(s)$, representing the sentence s using all word-level source embeddings, $\{\boldsymbol{f}_i(w, s)\}_{i=1}^n$, for the words w in s. Following the convention used in prior work on meta-embedding learning, we refer to individual $\boldsymbol{f}_i$ as source embeddings.

Note that in general the individual source embeddings will have different dimensionalities d_i. To transform each source embedding to the same meta-embedding space with dimensionality d_m, we first transform each source embedding by a source-specific projection matrix $\mathbf{A}_i \in \mathbb{R}^{d_m \times d_i}$ as given by (1).

$$\boldsymbol{\phi}_i(w, s) = \mathbf{A}_i \boldsymbol{f}_i(w, s) \tag{1}$$

Here, the projected source embeddings, $\boldsymbol{\phi}(w, s) \in \mathbb{R}^{d_m}$, will be in the same common d_m dimensional meta-embedding space for all source embeddings. The total number of learnable parameters in this model is $\sum_{i=1}^n d_m d_i$, which corresponds to the sum of elements in all projection matrices $\{\mathbf{A}_i\}_{i=1}^n$.

Different source embeddings will have complementary strengths and weaknesses. Therefore, we propose to assign weights to the source embeddings when creating a meta-embedding of those source embeddings as given in (2).

$$\boldsymbol{\phi}_i(w_j, s_k) = \alpha_i \mathbf{A}_i \boldsymbol{f}_i(w_j, s_k) \tag{2}$$

Here, α_i is the weight associated with the i-th source embedding such that $\sum_i \alpha_i = 1$. Moreover, to reduce the degrees of freedom of the model, we require cross-correlations in projection matrices to be small. This requirement can be implemented as a regularisation on the projection matrices such that $\left|\left|\mathbf{A}_i^\top \mathbf{A}_i - \mathbf{I}\right|\right|_F^2$ is minimised, where $||\cdot||_F$ denotes the Frobenius norm.

We can consider the individual projection matrices $\mathbf{A}_i$ to be concerned with projecting source embeddings with different dimensionalities to a common meta-embedding space, whereas α_i determines the weight imparted on the projected source embedding in the meta-embedding space. However, in the unsupervised meta-embedding learning setting that we consider in this paper, we do not assume the availability of any labelled data for downstream tasks nor our goal is to learn a meta-embedding that is specific to a particular task. On the other hand, task specialisation of meta-embedding can be done as a subsequent task akin to fine-tuning of foundation models. Therefore, we learn both α_i and $\mathbf{A}_i$ via an unsupervised learning method as described later. Given n distinct source embeddings, $\{\boldsymbol{f}_i(w, s)\}_{i=1}^n$, of a word w in a sentence s, we compute the word's meta-embedding, $\boldsymbol{m}_{\text{word}}(w, s)$, as the average of the projected source embeddings as given by (3).

$$\boldsymbol{m}_{\text{word}}(w, s) = \frac{1}{n} \sum_{i=1}^n \boldsymbol{\phi}_i(w, s) \tag{3}$$

Recall that the projection matrices ensure that source embeddings with different dimensionalities are projected to the same d_m dimensional meta-embedding space such that they can be averaged to obtain a reliable estimate of the final word-level meta-embedding.

Armed with the word-level meta-embeddings, $\boldsymbol{m}_{\text{word}}(w, s)$, for a word w in a sentence s, we compute two types of sentence-level meta-embeddings, $\boldsymbol{m}_{\text{mean}}(s)$ (defined by (5)) and $\boldsymbol{m}_{\text{max}}(s)$ (defined by (4)) by applying respectively mean or max pooling over the word embeddings $\boldsymbol{m}_{\text{word}}(w, s)$ corresponding to the words in the sentence s.

$$\boldsymbol{m}_{\text{mean}}(s) = \frac{1}{|s|} \sum_{w \in s} \boldsymbol{m}_{\text{word}}(w, s) \tag{4}$$

$$\boldsymbol{m}_{\text{max}}(s) = \max_{w \in s} \boldsymbol{m}_{\text{word}}(w, s) \tag{5}$$

The pooling operations are conducted elementwise over the set of word-level meta-embeddings. We compare the performances of meta sentence embeddings created using mean and max pooling in §4.

What remains in our model is to devise a method to learn the projection matrices, $\{\mathbf{A}_i\}_{i=1}^n$, and the weights $\{\alpha_i\}_{i=1}^n$. Recall that in this paper we are considering *unsupervised* sentence embeddings. Therefore, we do not assume the availability of labelled data such as human-annotated similarity ratings used in supervised sentence embedding learning. For this reason, next we derive an unsupervised training objective.

Let us consider the projected source embeddings of two words $w \in s$, $w' \in s'$ in two different sentences s and s' to be given respectively by $\phi_i(w, s)$ and $\phi_j(w', s')$ using two different sources f_i and f_j. We assume the following four criteria to be satisfied by these projected source embeddings.

Criterion 1: *The projected source embeddings $\phi_i(w, s)$ and $\phi_j(w, s)$ of the same word in the same sentence must be similar.*

This requirement stems from the definition of meta-embedding process because the purpose of the projection matrix in the first place was to ensure that different source embeddings of the same word are closer to each other in the meta-embedding space. This requirement can be formulated as the minimisation of the squared ℓ_2 distance between the projected source embeddings given by (6).

$$\left|\left|\phi_i(w, s) - \phi_j(w, s)\right|\right|_2^2 \qquad (6)$$

Criterion 2: *The projected source embeddings $\phi_i(w, s)$ and $\phi_j(w', s)$ of two different words w and w' that co-occur in the same sentence s must be dissimilar.*

This requirement is important to make the meta-embeddings of two distinct words to be sufficiently dissimilar such that we can discriminate them in the meta-embedding space. We formalise this requirement as the minimisation of the squared ℓ_2 distance given by (7)

$$\left|\left|\phi_i(w, s) - \phi_j(w', s)\right|\right|_2^2 \qquad (7)$$

Criterion 3: The projected source embeddings $\phi_i(w, s)$ and $\phi_j(w, s')$ of the same word w in two different sentences s and s' must be dissimilar.

This is particularly true for contextualised source embeddings, which consider both the word as well as the context in which the word occurs when representing the word. Moreover, we are considering embeddings produced by two distinct source embeddings here. This requirement is useful for creating different meta-embeddings for different sentences eventhough they might share some words in common. We formalise this requirement as the

minimisation of the squared ℓ_2 distance given by (8).

$$\left|\left|\phi_i(w, s) - \phi_j(w, s')\right|\right|_2^2 \qquad (8)$$

Criterion 4: *The projected source embeddings $\phi_i(w, s)$, $\phi_j(w', s')$ of two different words w, w' each occurring in different sentences s, s' must be dissimilar.*

This requirement can be seen as the most extreme dissimilar case involving distinct sources, distinct sentences and distinct words. It can also be seen as imposing a regularisation on the parameter learning objective that prevents degenerated solutions such as projecting all words and sentences to the same point in the meta-embedding space. We formalise this requirement as the minimisation of the squared ℓ_2 loss given by (9).

$$\left|\left|\phi_i(w, s) - \phi_j(w', s')\right|\right|_2^2 \qquad (9)$$

Finally, we combine the above-mentioned four constraints into a single training objective including the regulariser for the projection matrices as in (10).

$$
\begin{aligned}
&L(\{\mathbf{A}_i\}_{i=1}^n, \{\alpha_i\}_{i=1}^n) \\
&= \left|\left|\phi_i(w, s) - \phi_j(w, s)\right|\right|_2^2 \\
&\quad - \lambda \left|\left|\phi_i(w, s) - \phi_j(w', s)\right|\right|_2^2 \\
&\quad - \mu \left|\left|\phi_i(w, s) - \phi_j(w, s')\right|\right|_2^2 \\
&\quad - \nu \left|\left|\phi_i(w, s) - \phi_j(w', s')\right|\right|_2^2 \\
&\quad + \xi \sum_{i=1}^n \left|\left|\mathbf{A}_i^\top \mathbf{A}_i - \mathbf{I}\right|\right|_F^2 \qquad (10)
\end{aligned}
$$

Here, the regularisation coefficients λ, μ, ν and ξ are all optimised during training via backpropagation.

We randomly initialise all projection matrices and attention vectors and use Stochastic Gradient Descent (SGD) to optmise. Further details on learning settings are provided in §4.4.

Note that we do not assume any constraints for the projections of a single source embedding such as for two words in the same sentence, or the same word in different sentence for two reasons. First, many source embedding learning methods already use these constraints during pretraining. For example, masked language models such as BERT, masks a word in a sentence and requires that it can be predicted correctly using the embeddings of the other words in the sentence. This requirement forces the words in the same sentence to have similar embeddings. Second, during meta-embedding learning, we would like to preserve much information as possible in the original sources, without imposing constraints for the individual sources, beyond what they have been already trained with.

Note that we require no human labelled data such as labels for sentences or similarity ratings for creating sentence-level meta-embedding from the above-described proposed method. Therefore, we denote our unsupervised proposed method by **UNSUP** in the remainder of the paper.

4. Experiments

4.1. Source Embeddings

We use two source embeddings in our experiments – BERT[2] and RoBERTa.[3] All the models are pretrained by Reimers and Gurevych (2019) and Wolf et al. (2020). Although we use two source embeddings as a proof-of-concept for the proposed method, note that the proposed method itself is not limited to only two sources. For example, given n sources, we can consider pairwise combinations between any two of those n sources to compute the four criteria described in §3. Moreover, the BERT and RoBERTa sources we use in our evaluations have equal (i.e. 1024) dimensionalities. However, the proposed method *does not* require the dimensionalities of the source embeddings to be equal. In fact, the projection matrices will ensure that the averaging operation given by (3) can be carried out even if the dimensionalities of the sources are different. Further analysis on more than two sources and involving unequal dimensionalities are deferred to future work.

4.2. Data and Evaluation Metrics

Given that our goal in this paper is to create sentence-level meta-embeddings, we use Semantic Textual Similarity (STS) prediction as an evaluation task. In STS, we are given two sentences s and s' between which we must predict their semantic similarity. For example, we can compute the cosine similarity between s and s' using the sentence embeddings computed using a meta sentence embedding method. There exist several benchmark datasets where human annotators have provided similarity ratings, $r(s, s')$, between s and s', which can be considered as ground truth for evaluating sentence embedding methods. We can then compute the agreements between human similarity ratings and the predicted cosine similarities using correlation measures such as the Spearman and Pearson correlations. The SentEval tool (Conneau and Kiela, 2018) provides a unified framework for evaluating sentence embedding methods and reports the class-weighted averages of Pearson and Spearman correlation coefficients as the evaluation metric, which we use in this work. A high degree of correlation between human similarity ratings and cosine similarity scores computed using the sentence embeddings produced by a meta sentence embedding

method is considered as desirable, and accurately capturing the meaning of the sentences. We use the following STS datasets in our experiments: STS-15 (Agirre et al., 2015), STS-16 (Agirre et al., 2016)) and STS-B (STS benchmark) (Cer et al., 2017). Note that the purpose of evaluating on STS benchmark dataset is *not* to compare against the state-of-the-art for sentence embedding methods but to compare against the unsupervised sentence-level meta-embedding learning methods. With this objective in mind, we emphasise that the results reported in this paper must not be compared against general purpose sentence embedding learning methods, which are not combining multiple source embeddings as required by the meta-embedding learning methods we consider in this paper.

4.3. Baselines Methods

We compare our proposed method against several baseline methods, which we describe next.

Single Source Embeddings (SSE): We create sentence embeddings from each source embedding separately by applying either max or mean pooling over the word embeddings produced by that source for the words in a sentence. These baselines will demonstrate the level of performance that we can expect to obtain if we had used a single source embedding. This can be seen as a lower baseline for meta-embedding because one would expect a meta-embedding to always outperform the individual source embeddings used in it. We use both mean and max pooling given respectively by (4) and (5) to create sentence embeddings from the source embeddings in our evaluations.

Concatenation (CONC): In prior work of meta-embedding, it has been reported that by simply concatenating the different source embeddings of a word, one can obtain a surprisingly accurate meta-embedding for that word (Yin and Schütze, 2016b; Coates and Bollegala, 2018a). We extend this method to create sentence-level meta-embeddings by first creating word-level meta-embeddings for each word in a given sentence by the concatenation of each source embedding of that word. Next, we apply max or mean pooling over those concatenated word-level meta-embeddings. Specifically, for a given sentence s, its the sentence-level meta-embeddings, $\text{conc}_{\max}(s)$ and $\text{conc}_{\text{mean}}(s)$ are computed using respectively max and mean pooling methods as follows:

$$\text{conc}_{\max}(s) = \max_{w \in s} \bigoplus_{i=1}^{n} \boldsymbol{f}_i(w, s) \qquad (11)$$

$$\text{conc}_{\text{mean}}(s) = \frac{1}{|s|} \sum_{w \in s} \bigoplus_{i=1}^{n} \boldsymbol{f}_i(w, s) \qquad (12)$$

Here, $\bigoplus$ denotes the vector concatenation. Note that token pooling methods are applied *after* the concatenation operation.

One disadvantage of concatenation as a meta-embedding method is that because it adds up the dimensionalities of the source embeddings, when the number of sources increases or when the dimensionality of the source embeddings increases, thereby creating high-dimensional meta-embeddings. High-dimensional meta-embeddings require large storage spaces and are also slower to conduct operations such as computing cosine similarity between words/sentences. This is particularly problematic when creating meta-embeddings of contextualised embeddings produced by NLMs such as BERT and RoBERTa, which tend to have larger dimensionalities (e.g. 1024, 2048 etc.).

Averaging (AVG): Averaging is also a simple method to compute meta-embeddings that has performed surprisingly well in downstream tasks (Coates and Bollegala, 2018a; Poerner et al., 2020). Unlike concatenation, averaging *does not* increases the dimensionality of the meta-embedding when the number of source embeddings increases. This overcomes the computational and storage related disadvantages of concatenation described above. Specifically, for a given sentence s, its the sentence-level meta-embeddings, $\mathrm{avg}_{\mathrm{max}}(s)$ and $\mathrm{avg}_{\mathrm{mean}}(s)$ are computed using respectively max and mean pooling methods as follows:

$$\mathrm{avg}_{\mathrm{max}}(s) = \max_{w \in s} \sum_{i=1}^{n} \boldsymbol{f}_i(w, s) \qquad (13)$$

$$\mathrm{avg}_{\mathrm{mean}}(s) = \frac{1}{|s|} \sum_{w \in s} \sum_{i=1}^{n} \boldsymbol{f}_i(w, s) \qquad (14)$$

Although one cannot simply add (or average) vectors that lie in different vector spaces, Coates and Bollegala (2018a) showed that in the case of static word embeddings, such operations can be carried out because the source word embeddings are found to be approximately orthogonal in practice. However, this requirement might not be readily satisfied by a set of arbitrary source embeddings and prior work have trained orthogonal projections to explicitly enforce this orthogonality requirement, which has improved the performance of the resultant meta-embeddings (Jawanpuria et al., 2020; He et al., 2020a). Moreover, all source embeddings must have equal dimensionalities in order for us to compute their average. When this requirement is not satisfied, prior work have padded zeros as necessary prior to computing the average. Note that our proposal to learn projection matrices for individual source embeddings as given in (2) automatically solves both of those issues. Specifically, the projection matrices $\mathbf{A}_i$ convert source embeddings with potentially different dimensionalities to the same meta-embedding space (thereby obviating the need for zero padding), and simultaneously learn orthogonal projections.

SVD: Singular Value Decomposition was used as a method for creating sentence-level meta-embeddings by Yin and Schütze (2016b). Given a matrix $\mathbf{A} \in \mathbb{R}^{d_m \times d_n}$, its SVD is given by $\mathbf{A} = \mathbf{V} \mathbf{S} \mathbf{U}^{\top}$ where $\mathbf{V}, \mathbf{S}, \mathbf{U} \in \mathbb{R}^{d_m \times d_m}, \mathbb{R}^{d_m \times d_n}, \mathbb{R}^{d_n \times d_n}$. We follow Poerner et al. (2020) and apply SVD on the matrix $\mathbf{A}$ formed by row-wise zero-mean concatenation matrix of the source embeddings. We then use $\mathbf{U}^{\top}$ as the meta-sentence embedding. SVD overcomes the increasing dimensionality problem associated with CONC because the dimensionality is reduced in the SVD process by limiting to the largest eigenvectors.

GCCA: Generalised Canonical Correlation Analysis (Horst, 1961) (GCCA) was used as a method for creating sentence-level meta-embeddings by Poerner et al. (2020). Given two random vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, the Canonical Correlation Analysis (CCA) finds linear projections $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ such that $\boldsymbol{\lambda}^{\top} \boldsymbol{x}$ and $\boldsymbol{\mu}^{\top} \boldsymbol{y}$ are maximally correlated. GCCA extends this approach to three or more random vectors. We follow Poerner et al. (2020) and use the genealised eigenvalue decomposition-based approach proposed by Bach and Jordan (2002) to compute GCCA. GCCA is the current state-of-the-art method for creating meta sentence embeddings.

Supervised Meta-Embedding (SUP): Note that all of the above-mentioned baselines are all unsupervised meta-embedding methods, similar to our proposed method. To simulate the level of performance that we can hope to achieve if we had access to some labelled data for the target task, we propose a supervised baseline as follows. For this purpose we consider predicting semantic textual similarity between two given sentence as the target task. First, given a pair of sentences (s, s') and their human similarity rating $r(s, s')$, we compute a meta sentence embedding separately for s and s' using either mean pooling (given by (4)) and max pooling (given by (5)). Let us denote the meta sentence embedding for s and s' respectively by $\boldsymbol{m}(s)$ and $\boldsymbol{m}(s')$. We then compute the cosine similarity, $\cos(\boldsymbol{m}(s), \boldsymbol{m}(s'))$ using $\boldsymbol{m}(s)$ and $\boldsymbol{m}(s')$ and minimise the squared loss between the predicted cosine similarity and the human similarity rating, $r(s, s')$, between s and s' as given by (15).

$$L(\{\mathbf{A}_i\}_{i=1}^{n}, \{\alpha_i\}_{i=1}^{n})$$
$$= (\cos(\boldsymbol{m}(s), \boldsymbol{m}(s')) - r(s, s'))^2 \qquad (15)$$

Note that projection matrices $\{\mathbf{A}_i\}_{i=1}^{n}$ and weights $\{\alpha_i\}_{i=1}^{n}$ enter into (15) via (2) and get updated during backpropagation.

We use the training sentence pairs in the STS-B dataset (Cer et al., 2017) and minimise the loss given by (15) over all such sentence pairs. The original human similarity ratings in STS-B are in the range $[0, 5]$. We linearly transform both cosine similarities and human similarity ratings to $[-1, 1]$ by $\frac{x - \min(x)}{\max(x) - \min(x)}$ before computing the loss given in (15).

Table 1: Results of meta-sentence embeddings by using BERT and RoBERTa from sBERT (Reimers and Gurevych, 2019). We show both Pearson r and Spearman ρ correlation coefficients, separated by /.

Method	STS-15	STS-16	STS-B
SSE			
-BERT[a]	87.15/87.34	81.98/83.05	82.40/82.74
-BERT[b]	88.29/88.22	83.24/84.02	83.66/84.05
-RoBERTa[a]	75.94/78.69	72.96/75.20	82.46/83.75
-RoBERTa[b]	87.72/87.88	84.55/84.93	82.46/83.26
Meta			
CONC[a]	82.46/83.86	78.33/79.88	75.12/76.21
CONC[b]	88.64/88.59	84.27/84.88	84.69/85.14
AVG[a]	82.23/83.72	77.99/79.68	83.89/84.53
AVG[b]	88.47/88.45	83.97/84.63	84.21/84.47
SVD	88.66/88.65	84.07/84.63	83.98/84.61
GCCA	88.58/88.58	84.00/84.53	83.36/84.14
SUP	89.34/89.30	85.11/85.72	65.21/64.79
UNSUP	**88.76/88.85**	**85.06/85.33**	**85.33/86.08**

[a] max-pooling for word-to-sentence pooling.

[b] mean-pooling for word-to-sentence pooling.

4.4. Training Details

To train UNSUP, we use use the set of sentences that appear in the sentence pairs in the STS-B training split. Note that we *do not* use any human similarity ratings for the sentence pairs in the STS-B training split during the training of UNSUP. In fact, we do not even require pairs of sentences as they appear in STS-B for training UNSUP. We randomly pick two sentences that appear in (could be the same or different sentence pairs) and compute the loss given by (10), and update the projection matrices and weights such that the loss is minimised.

To train SUP, we use the sentence pairs and their human similarity ratings from the training sentence pairs in STS-B dataset.

We use SGD optimiser with weight decay 10^{-4}, learning rate 10^{-2} for both UNSUP and SUP methods, and a batch size of 512. We apply an early stopping as a regularisation technique such that, if the Pearson correlation does not improve for more than five consecutive epochs, computed over the development set in STS-B we stop further training. All the parameters SUP and UNSUP are randomly initialised in the $[-1, 1]$ range.

4.5. Results

Table 1 summarises the main results for the proposed unsupervised method (**UNSUP**) and the baseline methods described in §4.3. We use 1024-dimensional BERT and RoBERTa from sBERT (Reimers and Gurevych, 2019) as the source embeddings in all the experiments. The dimensionality in all meta-embedding methods compared in Table 1 are 1024, except for CONC, which is 2048 dimensional. Where applicable, we show the results obtained for the sentence embeddings created by max and mean pooling methods.

From Table 1, we see that for single source embeddings

Table 2: Ablation of attention and pooling for meta sentence embeddings. Here, BERT and RoBERTa are used as the source embeddings. The column corresponding to Pooling denotes the pooling method used for creating sentence-level meta-embeddings from word-level meta-embeddings as given by (4) and (5). Weight column denotes the weights being used. Specifically, w/ represents using the weights α_i in (2), whereas w/o represents not using the weights as in (1).

Method	Weight	Pooling	Pearson	Spearman
UNSUP	w/	mean	75.08	75.03
UNSUP	w/	max	**85.33**	**86.08**
UNSUP	w/o	mean	80.06	80.41
UNSUP	w/o	max	81.98	83.00

(SSE) the performance is consistently better when we use mean-pooling instead of max-pooling, except for RoBERTa in STS-B, where there is no significant difference between the two pooling methods. Next, in Table 1, we compare the performance of different baseline methods described in §4.3. We evaluate CONC and AVG using both max and mean pooling methods for creating sentence embeddings in Table 1. We see that CONC consistently reporting better performances than the best between the two individual source embedding in all datasets. This shows that CONC is still a strong baseline not only when computing meta embeddings from static word embeddings as reported in prior work (Coates and Bollegala, 2018a), but also in the case of contextualised word embeddings.

AVG on the other hand closely follows the performance of CONC but never outperforms CONC in any of the datasets. Here again we see that the mean pooling variants of CONC and AVG outperform their max pooling counterparts. Although SVD slightly outperforms CONC in terms of Spearman correlation in STS-15, it reports consistently lower performance than CONC in all other settings. This shows that although SVD can reduce the dimensionality compared to CONC, the performance drops due to possible information loss during the dimensionality reduction process. GCCA consistently under performs compared to SVD in all datasets.

The proposed **UNSUP** and the supervised baseline (**SUP**) are shown at the bottom two rows in Table 1. We see that SUP outperforms all baselines including CONC in both STS-15 and STS-16 datasets but not on STS-B. More importantly USNUP outperforms all methods compared in Table 1 and obtains the best performance on all datasets in terms of both Spearman and Pearson correlation coefficients. In particular, UNSUP outperforms the current state-of-the-art meta sentence embedding method, GCCA, proposed by Poerner et al. (2020).

4.6. Effect of Weighting and Pooling

We study the effect of using weights α_i with max and mean pooling methods in Table 2. Specifically, w/ represents using the weights α_i in (2), whereas w/o represents not using the weights as in (1). From Table 2, we see that the best performance is obtained with UNSUP when we use max pooling and with weighting.

5. Conclusion

We propose an unsupervised method to create sentence-level meta-embeddings from multiple contextualised word embeddings. Our proposed method outperformed several competitive baselines and a previous proposed state-of-the-art sentence-level unsupervised meta-embedding method. In particular, both weighted linear transformation and pooling operations contributed to the optimal performance of the proposed method.

Bibliographic 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 *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 252–263, Denver, Colorado, June. Association for Computational Linguistics.

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 *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 497–511, San Diego, California, June. Association for Computational Linguistics.

Bach, F. R. and Jordan, M. I. (2002). Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, July.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. *Proc. of ICLR*.

Bao, C. and Bollegala, D. (2018). Learning word meta-embeddings by autoencoding. In *Proc. of COLING*, pages 1650–1661.

Bollegala, D., Hayashi, K., and Kawarabayashi, K.-i. (2018). Think globally, embed locally — locally linear meta-embedding of words. In *Proc. of IJCAI-EACI*, pages 3970–3976.

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 *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada, August. Association for Computational Linguistics.

Coates, J. and Bollegala, D. (2018a). Frustratingly easy meta-embedding – computing meta-embeddings by averaging source word embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 194–198, New Orleans, Louisiana, June. Association for Computational Linguistics.

Coates, J. and Bollegala, D. (2018b). Frustratingly easy meta-embedding – computing meta-embeddings by averaging source word embeddings. In *Proc. of NAACL-HLT*, pages 194–198.

Conneau, A. and Kiela, D. (2018). SentEval: An Evaluation Toolkit for Universal Sentence Representations.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL-HLT*, pages 4171–4186.

Goikoetxea, J., Agirre, E., and Soroa, A. (2016). Single or multiple? combining word representations independently learned from text and wordnet. In *Proc. of AAAI*, pages 2608–2614.

He, J., Tsiolis, K., Kenyon-Dean, K., and Cheung, J. C. K. (2020a). Learning efficient task-specific meta-embeddings with word prisms. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1229–1241, Barcelona, Spain (Online), December. International Committee on Computational Linguistics.

He, J., Tsiolis, K., Kenyon-Dean, K., and Cheung, J. C. K. (2020b). Learning Efficient Task-Specific Meta-Embeddings with Word Prisms. In *Proc. of COLING*.

Horst, P. (1961). Generalized canonical correlations and their applications to experimental data. In *Clinical Psychology*, volume 17, pages 331–347.

Jawanpuria, P., N T V, S. D., Kunchukuttan, A., and Mishra, B. (2020). Learning geometric word meta-embeddings. In *Proc. of RepL4NLP*, pages 39–44.

Kiela, D., Wang, C., and Cho, K. (2018). Dynamic meta-embeddings for improved sentence representations. In *Proc. of EMNLP*, pages 1466–1477.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. In Yoshua Bengio et al., editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, et al., editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

O'Neill, J. and Bollegala, D. (2018). Angular-Based Word Meta-Embedding Learning.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: global vectors for word representation. In *Proc. of EMNLP*, pages 1532–1543.

Poerner, N., Waltinger, U., and Schütze, H. (2020). Sentence meta-embeddings for unsupervised semantic textual similarity. In *Proc. of ACL*, pages 7027–7034.

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11.

Sennrich, R., Haddow, B., and Birch, A. (2016). Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August. Association for Computational Linguistics.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October. Association for Computational Linguistics.

Wu, X., Cai, Y., Kai, Y., Wang, T., and Li, Q. (2020). Task-oriented domain-specific meta-embedding for text classification. In *Proc. of EMNLP*, pages 3508–3513.

Xie, Y., Hu, Y., Xing, L., and Wei, X. (2019). Dynamic task-specific factors for meta-embedding. In *International Conference on Knowledge Science, Engineering and Management*, pages 63–74. Springer.

Yin, W. and Schütze, H. (2016a). Learning word meta-embeddings. In *Proc. of ACL*, pages 1351–1360.

Yin, W. and Schütze, H. (2016b). Learning word meta-embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1351–1360, Berlin, Germany, August. Association for Computational Linguistics.