TSDAE: Using Transformer-based Sequential Denoising Auto-Encoder for Unsupervised Sentence Embedding Learning

Kexin Wang, Nils Reimers, Iryna Gurevych
Ubiquitous Knowledge Processing Lab (UKP-TUDA)
Department of Computer Science, Technical University of Darmstadt
www.ukp.tu-darmstadt.de

Abstract

Learning sentence embeddings often requires a large amount of labeled data. However, for most tasks and domains, labeled data is seldom available and creating it is expensive. In this work, we present a new state-of-the-art unsupervised method based on pre-trained Transformers and Sequential Denoising Auto-Encoder (TSDAE) which outperforms previous approaches by up to 6.4 points. It can achieve up to 93.1% of the performance of in-domain supervised approaches. Further, we show that TSDAE is a strong domain adaptation and pre-training method for sentence embeddings, significantly outperforming other approaches like Masked Language Model.[1]

A crucial shortcoming of previous studies is the narrow evaluation: Most work mainly evaluates on the single task of Semantic Textual Similarity (STS), which does not require any domain knowledge. It is unclear if these proposed methods generalize to other domains and tasks. We fill this gap and evaluate TSDAE and other recent approaches on four different datasets from heterogeneous domains.

1 Introduction

Sentence embedding techniques encode sentences into a fixed-sized, dense vector space such that semantically similar sentences are close. The most successful previous approaches like InferSent (Conneau et al., 2017), Universial Sentence Encoder (USE) (Cer et al., 2018) and SBERT (Reimers and Gurevych, 2019) heavily relied on labeled data to train sentence embedding models. However, for most tasks and domains, labeled data is not available and data annotation is expensive. To overcome this limitation, unsupervised approaches have been proposed which learn to embed sentences just using an unlabeled corpus for training.

We propose a new approach: Transformer-based Sequential Denoising Auto-Encoder (TSDAE). It significantly outperforms previous methods via an encoder-decoder architecture. During training, TSDAE encodes corrupted sentences into fixed-sized vectors and requires the decoder to reconstruct the original sentences from this sentence embedding. For good reconstruction quality, the semantics must be captured well in the sentence embedding from the encoder. Later, at inference, we only use the encoder for creating sentence embeddings.

A crucial shortcoming of previous unsupervised approaches is the evaluation. Often, approaches are mainly evaluated on the Semantic Textual Similarity (STS) task from SemEval (Li et al., 2020; Giorgi et al., 2021; Carlsson et al., 2021; Gao et al., 2021). As we argue in Section 4, we perceive this as an insufficient evaluation. The STS datasets do not include sentences with domain specific knowledge, i.e., it remains unclear how methods will perform on more specific domains. Further, STS datasets have an artificial score distribution, and the performance on STS datasets does not correlate with downstream task performances (Reimers et al., 2016). In conclusion, it remains unclear, how well unsupervised sentence embedding methods will perform on domain specific tasks.

To answer this question, we compare TSDAE with previous unsupervised sentence embedding approaches on three different tasks (Information Retrieval, Re-Ranking and Paraphrase Identification), for heterogeneous domains and different text styles. We show that TSDAE can outperform other state-of-the-art unsupervised approaches by up to 6.4 points. TSDAE is able to perform on-par or even outperform existent supervised models like USE-large, which had been trained with a lot of labeled data from various datasets.

Further, we demonstrate that TSDAE works well for domain adaptation and as a pre-training task. We observe a significant performance improvement

[1]Code available at: https://github.com/UKPLab/sentence-transformers/

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 671–688
November 7–11, 2021.

compared to other pre-training tasks like Masked Language Model (MLM).

Our contributions are three-fold:

- We propose a novel unsupervised method, TSDAE based on denoising auto-encoders. We show that it outperforms the previous best approach by up to 6.4 points on diverse datasets.
- To the best of our knowledge, we are the first to compare recent unsupervised sentence embedding methods for various tasks on heterogeneous domains.
- TSDAE outperforms other methods including MLM by a large margin as a pre-training and domain adaptation method.

2 Related Work

Supervised sentence embeddings utilize labels for sentence pairs which provide the information about the relation between the sentences. Since sentence embeddings are usually applied to measure the similarity of a sentence pair, the most direct way is to label this similarity for supervised training (Henderson et al., 2017). Many studies also find that natural language inference (NLI), question answering and conversational context datasets can successfully be used to train sentence embeddings (Conneau et al., 2017; Cer et al., 2018). The recently proposed Sentence-BERT (Reimers and Gurevych, 2019) introduced pre-trained Transformers to the field of sentence embeddings. Although high-quality sentence embeddings can be derived via supervised training, the labeling cost is a major obstacle for practical usage, especially for specialized domains.

Unsupervised sentence embeddings utilize only an unlabeled corpus during training. Recent work combined pre-trained Transformers with different training objectives to achieve state-of-the-art results on STS tasks. Among them, Contrastive Tension (CT) (Giorgi et al., 2021) simply views the identical and different sentences as positive and negative examples, resp. and train two independent encoders; BERT-flow (Li et al., 2020) trains model via debiasing embedding distribution towards Gaussian; SimCSE (Gao et al., 2021) is based on contrastive learning (Hadsell et al., 2006; Chen et al., 2020) and views the identical sentences with different dropout mask as the positive examples. For more details, please refer to Section 5. All of them

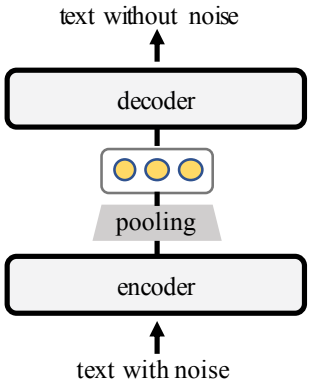


Figure 1: Architecture of TSDAE.

requires only independent sentences. By contrast, DeCLUTR (Giorgi et al., 2021) utilizes sentence-level contexts and requires long documents (2048 tokens at least) for training. This requirement is hardly met for many cases, e.g. tweets or dialogues. Thus, in this work we only consider methods which uses only single sentences during training.

Most previous work mainly evaluate only on Semantic Textual Similarity (STS) from the SemEval shared tasks. As we show in Section 4, the unsupervised approaches perform much worse than the out-of-the-box supervised pre-trained models even though they were not specifically trained for STS. Further, a good performance on STS does not necessarily correlate with the performance on downstream tasks (Reimers et al., 2016). It remains unclear how these methods perform on specific tasks and domains. To answer this, we compare three recent powerful unsupervised methods based on pre-trained Transformers including CT, SimCSE, BERT-flow and our proposed TSDAE on different tasks of heterogeneous domains.

3 Sequential Denoising Auto-Encoder

Although Sequential Denoising Auto-Encoder (SDAE) (Vincent et al., 2010; Goodfellow et al., 2016; Hill et al., 2016) is a popular unsupervised method in machine learning, how to combine it with pre-trained Transformers for learning sentence embeddings remains unclear. In this section, we first introduce the training objective of TSDAE and then give the optimal configuration of TSDAE.

3.1 Training Objective

Figure 1 illustrates the architecture of TSDAE. TSDAE train sentence embeddings by adding a certain type of noise (e.g. deleting or swapping words) to

input sentences, encoding the damaged sentences into fixed-sized vectors and then reconstructing the vectors into the original input. Formally, the training objective is:

$$\begin{aligned} J_{\text{SDAE}}(\theta) &= \mathbb{E}_{x\sim D}[\log P_\theta(x|\tilde{x})] \\ &= \mathbb{E}_{x\sim D}[\sum_{t=1}^{l} \log P_\theta(x_t|\tilde{x})] \\ &= \mathbb{E}_{x\sim D}[\sum_{t=1}^{l} \log \frac{\exp(h_t^T e_t)}{\sum_{i=1}^{N} \exp(h_t^T e_i)}] \end{aligned}$$

where D is the training corpus, $x = x_1 x_2 \cdots x_l$ is the input sentence with l tokens, $\tilde{x}$ is the corresponding damaged sentence, e_t is the word embedding of x_t, N is the vocabulary size and h_t is the hidden state at decoding step t.

An important difference to original transformer encoder-decoder setup presented in Vaswani et al. (2017) is the information available to the decoder: Our decoder decodes only from a fixed-size sentence representation produced by the encoder. It does not have access to all contextualized word embeddings from the encoder. This modification introduces a bottleneck, that should force the encoder to produce a meaningful sentence representation.

3.2 TSDAE

The model architecture of TSDAE is a modified encoder-decoder Transformer where the key and value of the cross-attention are both confined to the sentence embedding only. Formally, the formulation of the modified cross-attention is:

$$H^{(k)} = \text{Attention}(H^{(k-1)}, [s^T], [s^T])$$

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$$

where $H^{(k)} \in \mathbb{R}^{t\times d}$ is the decoder hidden states within t decoding steps at the k-th layer, d is the size of the sentence embedding, $[s^T] \in \mathbb{R}^{1\times d}$ is a one-row matrix including the sentence embedding vector and Q, K and V are the query, key and value, respectively. By exploring different configurations on the STS benchmark dataset (Cer et al., 2017), we discover that the best combination is: (1) adopting deletion as the input noise and setting the deletion ratio to 0.6, (2) using the output of the `[CLS]` token as fixed-sized sentence representation (3) tying the encoder and decoder parameters during training. For the detailed tuning process, please refer to Appendix A.

4 Evaluation

Previous unsupervised sentence embedding learning approaches (Giorgi et al., 2021; Carlsson et al., 2021; Li et al., 2020; Su et al., 2021; Gao et al., 2021) primarily evaluated on the task of Semantic Textual Similarity (STS) with data from SemEval using Pearson or Spearman's rank correlation.

We find the (sole) evaluation on STS problematic. As shown in (Reimers et al., 2016), performance on the STS dataset does not correlate with downstream task performance, i.e. an approach working well on the STS tasks must not be a good choice for downstream tasks. We confirm this with our experiments, the performance on the STS tasks does not correlate with the performance on other (real-world) tasks. See Section 6.1 for more details on this.

This has multiple reasons: First, the STS datasets consists of sentences which do not require domain-specific knowledge, they are primarily from news and image captions. It is unclear how approaches will work for domain-specific tasks. Second, the STS datasets have an artificial score distribution - dissimilar and similar pairs appear roughly equally. For most real-word tasks, there is an extreme skew and only a tiny fraction of pairs are considered similar. Third, to perform well on the STS datasets, a method must rank dissimilar pairs and similar pairs equally well. In contrast, most real-world tasks, like duplicate questions detection, related paper finding, or paraphrase mining, only require to identify the few similar pairs out of a pool of millions of irrelevant combinations.

A further shortcoming of previous evaluation setups is just testing the case of unsupervised learning, ignoring labeled data that potentially exists. In many scenarios, some labeled data exists, either directly from the specific task or from other (similar) tasks. A good approach should also work if some labeled data is available.

Hence, we propose to evaluate unsupervised sentence embedding approaches in following three setups:

Unsupervised Learning: We assume we just have unlabeled sentences from the target task and tune our approaches based on these sentences.

Domain Adaptation: We assume we have unlabeled sentences from the target task and labeled sentences from NLI (Bowman et al., 2015; Williams et al., 2018) and STS benchmark (Cer et al., 2017) datasets. We test two setups: 1) Training on

NLI+STS data, then unsupervised training to the target domain, 2) Unsupervised training on the target domain, then supervised training on NLI + STS.

Pre-Training: We assume we have a larger collection of unlabeled sentences from the target task and a smaller set of labeled sentences from the target task.

4.1 Datasets

We evaluate these three settings on different tasks from heterogeneous (specialized) domains. The tasks include Re-Ranking (RR), Information Retrieval (IR) and Paraphrase Identification (PI). In detail, the datasets used are as follows[2]:

AskUbuntu (RR task) is a collection of user posts from the technical forum AskUbuntu (Lei et al., 2016). Models are required to re-rank 20 candidate questions according to the similarity given an input post. The candidates are obtained via BM25 term-matching (Robertson et al., 1994). The evaluation metric is Mean Average Precision (MAP).

CQADupStack (IR task) is a question retrieval dataset of forum posts on various topics from StackExchange (Hoogeveen et al., 2015). In detail, it has 12 forums including Android, English, gaming, geographic information system, Mathematica, physics, programmers, statistics, Tex, Unix, Webmasters and WordPress. Models are required to retrieve duplicate questions from a large candidate pool. The metric is MAP@100. We train a single model for all forums.

TwitterPara (PI task) consists of two similar datasets: the Twitter Paraphrase Corpus (PIT-2015) (Xu et al., 2015) and the Twitter News URL Corpus (noted as TURL) (Lan et al., 2017). The dataset consists of pairs of tweets together with a crowd-annotated score if the pair is a paraphrase. The evaluation metric is Average Precision (AP) over the gold confidence scores and the similarity scores from the models.

SciDocs (RR task) is a benchmark consisting of multiple tasks about scientific papers (Cohan et al., 2020). In our experiments, we use the tasks of *Cite*: Given a paper title, identify the titles the paper is citing; *Co-Cite (CC)*, *Co-Read (CR)*, and *Co-View (CV)*, for which we must find papers that are frequently co-cited/-read/-viewed for a given paper title. For all these tasks, given one query paper title, models are required to identify up to 5 relevant papers titles from up to 30 candidates. The negative examples were selected randomly. The evaluation metric is MAP.

For evaluation, sentences are first encoded into fixed-sized vectors and cosine similarity is used for sentence similarity. Since we focus on embeddings for sentences, we just use the titles from the AskUbuntu, CQADupStack and SciDocs datasets. For the datasets with sub-datasets or sub-tasks including CQADupStack, TwitterPara and SciDocs, the final score is derived by averaging the scores from each sub-dataset or sub-task.

For unsupervised training, we just use the sentences from the training split without any labels. The statistics for each dataset are shown in Table 1.

5 Experiments

In this section, we compare our proposed TSDAE with other unsupervised counterparts and out-of-the-box supervised pre-trained models on the above mentioned tasks. For comparison, we include three recent state-of-the-art unsupervised approaches: CT, SimCSE, and BERT-flow. We use the proposed hyper-parameters from the respective paper. Without other specification, *BERT-base-uncased*[3] is used as the base Transformer model. To eliminate the influence of randomness, we report the scores averaged over 5 random seeds. For other details, please refer to Appendix B.

5.1 Baseline Methods

We compare the approaches against avg. GloVe embeddings (Pennington et al., 2014) and Sent2Vec (Pagliardini et al., 2018). The former generates sentence embeddings by averaging word embeddings trained on a large corpus from the general domain; the latter is also a bag-of-words model but trained on the in-domain unlabeled corpus. The unsupervised baseline of *BERT-base-uncased* with mean pooling is also in comparison. We further compare against existent pre-trained models: Universial Sentence Embedding (USE) (Yang et al., 2020), which was trained on multiple supervised datasets including NLI and community question answering. From the Sentence-Transformers package, we use *SBERT-base-nli-v2* and *SBERT-base-nli-stsb-v2*: These models were trained on SNLI + MultiNLI data using the Multiple-Negative Ranking Loss (MNRL) (Henderson et al., 2017) and the

[2]The dataset splits and the evaluation toolkit are available at: https://github.com/UKPLab/useb

[3]Results for other checkpoints is reported in Appendix C

Dataset	Task	#queries	Avg. #relevant	Avg. #candidates	Avg. length	Size of unsupervised training set	Size of supervised training set
AskUbuntu	RR	200	5.9/5.4	20	9.2	165K	23K
CQADupStack	IR	3K	1.1/1.1	39K	8.6	44K	13K
SciDocs	RR	4K	5	30	12.5	312K	380K

Dataset	Task	#paraphrase	#non-paraphrase	Avg. length	Size of unsupervised training set	Size of supervised training set
TwitterPara	PI	–/2K	–/9K	13.9	53K	23K

Table 1: Dataset statistics. The slash symbol '/' separates the numbers for development and test. Multiple sub-datasets are included in CQADupStack, SciDocs and TwitterPara. CQADupStack has one sub-dataset for each of the 12 forums. The avg. #relevant, avg. #candidates and avg. length are all general statistics without distinguishing the sub-datasets.

Mean Square Error (MSE) loss on the STS benchmark train set. Further we include BM25 using Elasticsearch for comparison.

To better understand the relative performance of these unsupervised methods, we also train SBERT models in an in-domain supervised manner and view their scores as the upper bound. For AskUbuntu, CQADupStack and SciDocs, where the relevant sentence pairs are labeled, the in-domain SBERT models are trained with MNRL. MNRL is a cross-entropy loss with in-batch negatives. For a batch of relevant sentences pairs $\{x^{(i)}, y^{(i)}\}_{i=1}^{M}$, MNRL views the labeled pairs as positive and the other in-batch combinations as negative. Formally, the training objective for each batch is:

$$J_{\text{MNRL}}(\theta) = \frac{1}{M}\sum_{i=1}^{M} \log \frac{\exp \sigma(f_\theta(x^{(i)}), f_\theta(y^{(i)}))}{\sum_{j=1}^{M} \exp \sigma(f_\theta(x^{(i)}), f_\theta(y^{(j)}))}$$

where σ is a certain similarity function for vectors and f_θ is the sentence encoder that embeds sentences. For TwitterPara, whose relevant scores are labeled, the MSE loss is adopted to train the in-domain models.

5.2 MLM

Masked-Language-Model (MLM) is a fill-in-the-blank task originally introduced by BERT: Words are masked from the input and the transformer network must predict the missing words. We use the original setup in Devlin et al. (2019) except the number of training steps (100K), the batch size (8) and the learning rate (5e-5). To derive a sentence embedding, we perform mean-pooling of the output token embeddings.

5.3 Contrastive Tension (CT)

CT (Carlsson et al., 2021) finetunes pre-trained Transformers in a contrastive-learning fashion. For each sentence, it construct a binary cross-entropy loss by viewing the same sentence as the relevant and samples K random sentences as the irrelevant. To make the training process stable, for each sentence pair (a, b), CT uses two independent encoders f_{θ_1} and f_{θ_2} from the same initial parameter point to encode the sentence a and b, respectively. Formally, the learning objective is:

$$J_{\text{CT}}(\theta_1, \theta_2) = \mathbb{E}_{(a,b)\sim D}[y \log \sigma(f_{\theta_1}(a)^T f_{\theta_2}(b)) + (1-y)\log(1 - \sigma(f_{\theta_1}(a)^T f_{\theta_2}(b))]$$

where $y \in \{0, 1\}$ represents whether sentence a is identical to sentence b and σ is the Logistic function. Despite its simplicity, CT achieves state-of-the-art unsupervised performance on the Semantic Textual Similarity (STS) datasets.

5.4 SimCSE

Similar to CT, SimCSE (Gao et al., 2021) also views the identical sentences as the positive examples. The main difference is that SimCSE samples different dropout masks for the same sentence to generate a embedding-level positive pair and uses in-batch negatives. Thus, this learning objective is equivalent to feeding each batch of sentences to the shared encoder twice and applying the MNRL-loss.

5.5 BERT-flow

Instead of fine-tuning the parameters of the pre-trained Transformers, BERT-flow (Li et al., 2020) aims at fully exploiting the semantic information encoded by these pre-trained models themselves via distribution debiasing. The paper of BERT-flow

claims that the BERT word embeddings are highly relevant to the word frequency, which in turn influences the hidden states via the Masked Language Modeling (MLM) pre-training. This finally leads to biased sentence embeddings generated by the pooling over these hidden states. To solve this problem, BERT-flow inputs the biased sentence embedding into a trainable flow network f_ϕ (Kingma and Dhariwal, 2018) for debiasing via fitting a standard Gaussian distribution, while keeping the parameters of the BERT model unchanged. Formally, the training objective is:

$$J_{\text{BERT-flow}}(\phi) = \mathbb{E}_{x\sim D}[\log p_{\mathcal{U}}(u)] \quad (1)$$

$$= \mathbb{E}_u[\log(p_{\mathcal{Z}}(f_\phi^{-1}(u))|\det \frac{\partial f_\phi^{-1}(u)}{\partial u}|)] \quad (2)$$

where u is the biased embedding of sentence x and $z = f_\phi^{-1}(u)$ is the debiased sentence embedding which follows a standard Gaussian distribution. Equation 2 is derived by applying the change-of-variables theorem to Equation 1.

As BERT-flow does not update the parameters of the underlying Transformer network, we just reports scores for BERT-flow for *unsupervised learning* and *domain adaptation NLI+STS → target task*. It is not suitable for the other evaluation setups we used. We re-implemented BERT-flow under the Pytorch framework, which can reproduce the reported results in the original paper.[4]

6 Results

Unsupervised learning: The results in Table 2 show that TSDAE can outperform the previous best approach (CT) by up-to 6.4 points (on SciDocs on average) and 2.6 points on average over all tasks. Surprisingly, a simple Masked-Language-Modeling (MLM) approach with mean pooling, which performs badly when evaluated on STS data, is the second best unsupervised approach, outperforming more recent approaches like CT, SimCSE, and BERT-flow on the selected tasks. TSDAE and MLM both removes words from the input, forcing the network to produce robust embeddings. In contrast, the input sentences for CT and SimCSE are not modified, resulting in less stable embeddings. Our experiments also show that out-of-the-box pre-trained models (SBERT-base-nli-stsb-v2 and USE-large) achieve strong results on our tasks without any domain-specific fine-tuning, outperforming recent proposed unsupervised learning approaches.

Domain Adaptation: For all unsupervised methods, we find that first training on the target domain, and then training with labeled NLI+STS achieves better results than the opposite direction. For all methods, we observe a performance increase compared to only training on the target domain. On average, the performance improves by 1.3 points for TSDAE, 3.0 points for MLM, 0.6 points for CT, and 1.8 points for SimCSE. CT and SimCSE perform in this setting only slightly better than the out-of-the-box model SBERT-base-nli-stsb-v2.

Pre-training: In Figure 2 we compare the pre-training performance of the tested approaches: We first pre-train on all available unlabeled sentences and then perform in-domain supervised training with different labeled training set sizes. Scores are reported by evaluation on the development sets. TSDAE outperforms MLM by a significant margin for all datasets except for AskUbuntu. There, MLM works slightly better. For the other datasets, TSDAE shows a clear out-performance to other pre-training strategies. The difference is quite consistent also for larger labeled training sets. We conclude, that TSDAE works well as pre-training method and can significantly improve the performance for later supervised training even for larger training datasets. CT and SimCSE don't perform well for pre-training, the results are far worse than using TSDAE/MLM or even starting from the pre-trained SBERT-nli-stsb model.

6.1 Results on STS data

We sample sentences from Wikipedia as done by Carlsson et al. (2021) and train a *BERT-base-uncased* model on this dataset with the different unsupervised training methods. In Table 3, we show the performance (Spearman's rank correlation) on the test set of the STS benchmark[5] along with the avg. performance on our four domain-specific tasks. See Appendix F for results on other STS datasets.

We observe quite different behaviour when evaluating on STS data compared to evaluating on our domain specific tasks. On STS data, CT and SimCSE perform strongly, outperforming MLM and TSDAE by a large margin. However, when ap-

[4]Code available at: https://github.com/UKPLab/pytorch-bertflow

[5]In the original paper of BERT-flow, the mean pooling over the first and the last layer is used, which causes the discrepancy on the STS scores. However, for a comparable setting, as the choice of most of the previous work, we only consider the pooling over the last layer.

Method	AskU.	CQADup.	TwitterP.			SciDocs					Avg.
Sub-task/-dataset			**TURL**	**PIT**	**Avg.**	**Cite**	**CC**	**CR**	**CV**	**Avg.**	
Unsupervised learning based on BERT-base											
TSDAE	59.4†	14.5†	76.8†	69.2	73.0	71.4†	73.9†	75.0†	75.6†	74.0†	55.2†
MLM	54.3	11.7	71.9	69.7	70.8	71.2	75.8	75.1	76.2	74.6	52.9
CT	56.3	13.3	74.6	70.4	72.5	63.4	67.1	70.1	69.7	67.6	52.4
SimCSE	55.9	12.4	74.5	62.5	68.5	62.5	65.1	67.7	67.6	65.7	50.6
BERT-flow	53.7	9.2	72.8	65.7	69.3	61.3	62.8	66.7	67.1	64.5	49.2
Domain adaptation: NLI+STS → target task											
TSDAE	58.7	13.6	75.8	66.2	71	69.9†	73.8†	75†	75.7†	73.6†	54.2†
MLM	54.4	9.7	69.8	68.1	69	67.1	71.8	72.6	72.9	71.1	51.1
CT	57.9	14.2	75.6	70.6	73.1	62.3	66.2	68.5	68.9	66.5	52.9
SimCSE	56.6	13.8	73.4	65.9	69.7	61.8	63.7	67.01	66.7	64.8	51.2
BERT-flow	58.2	13.9	76.5	67.4	72	62.2	64.8	68.1	68	65.8	52.5
Domain adaptation: target task → NLI+STS											
TSDAE	59.4†	14.4†	75.8	**73.1**†	**74.5**†	**75.6**†	**78.6**†	**78.1**†	**78.2**†	**77.6**†	**56.5**†
MLM	**60.6**	14.3	75.0	68.6	71.8	74.7	78.2	77.0	77.6	76.9	55.9
CT	56.4	13.4	75.9	68.9	72.4	66.5	69.6	70.6	72.2	69.7	53.0
SimCSE	56.2	13.1	75.5	67.3	71.4	65.5	68.5	70.0	71.4	68.9	52.4
Other previous unsupervised approaches											
BM25	53.4	13.3	71.9	70.5	71.2	58.9	61.3	67.3	66.9	63.6	50.4
Avg. GloVe	51.0	10.0	70.1	52.1	61.1	58.8	60.6	64.2	65.4	62.2	46.1
Sent2Vec	49.0	3.2	47.5	39.9	43.7	61.6	66.0	66.1	66.7	65.1	40.2
BERT-base-uncased	48.5	6.5	69.1	61.7	65.4	59.4	65.1	65.4	68.6	64.6	46.3
Out-of-the-box supervised pre-trained models											
SBERT-base-nli-v2	53.4	11.8	75.4	69.9	72.7	66.8	70.0	70.7	72.8	70.1	52.0
SBERT-base-nli-stsb-v2	54.5	12.9	75.9	68.5	72.2	66.2	69.2	69.9	72.3	69.4	52.3
USE-large	(*59.3*)	(***15.9***)	**77.1**	69.8	73.5	67.1	69.5	71.4	72.6	70.2	54.7
In-domain supervised training (upper bound)											
SBERT-supervised	63.8	16.3	81.6	75.8	78.7	90.4	91.2	86.2	83.6	87.9	61.6

Table 2: Evaluation using average precision. Results are averaged over 5 random seeds. The best results excluding the upper bound are bold. USE-large was trained with in-domain training data for AskUbuntu and CQADupStack (scores in italic). Our proposed TSDAE significantly outperforms other unsupervised and supervised out-of-the-box approaches.† marks the cases where TSDAE outperforms both CT and SimCSE in all 5 runs.

Method	STSb	Specific Tasks
Unsupervised method		
TSDAE	66.0	**55.2**
MLM	47.3	52.9
CT	73.9	52.4
SimCSE	73.8	50.6
BERT-flow	48.9	49.2
Out-of-the-box supervised pre-trained models		
SBERT-base-nli-v2	83.9	52.0
SBERT-base-nli-stsb-v2	**87.3**	52.3
USE-large	80.9	54.7

Table 3: Performance (Spearman's rank correlation) on the STS benchmark test set. Specific tasks: Average performance from Table 2.

plied to domain-specific real-world tasks, TSDAE and MLM are outperforming CT and SimCSE. We think these are due to the reasons mentioned in Section 4. Overall, we conclude that a strong performance on STS data is not a good indicator for good performance on domain-specific tasks.

7 Analysis

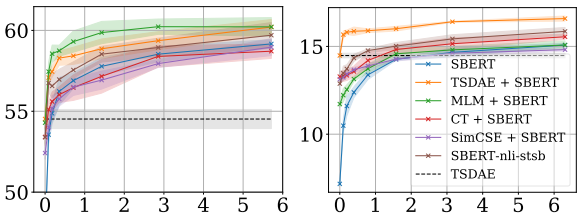
We analyze how many training sentences are needed and if relevant content words are identified.

For all the datasets except TwitterPara, the analysis is carried out on the development set. For TwitterPara, the test set is used, as it has no development split released by the original paper. All the hyper-parameters are chosen up-front without tuning to a particular dataset.

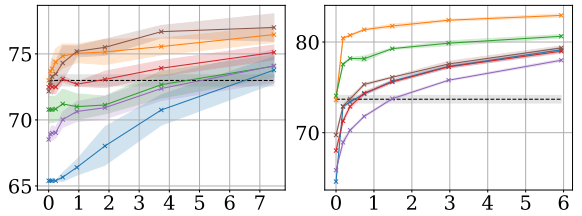
7.1 Influence of Corpus Size

In certain domains, getting a sufficiently high number of (unlabeled) sentences can be challenging. Hence, data efficiency and deriving good sentence embeddings even with little unlabeled training data can be important.

In order to study this, we train the unsupervised approaches with different corpus sizes: Between 128 and 65,536 sentences. For each experiment, we train a bert-base-uncased model with 10 epochs up to 100k training steps. The models are evaluated

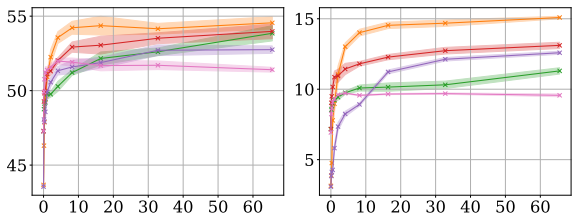


(a) AskUbuntu (b) CQADupStack

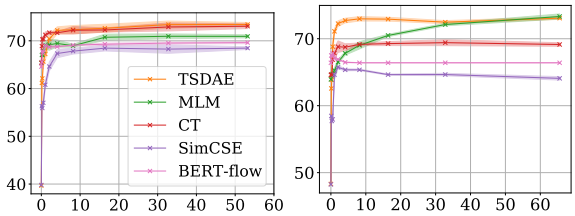


(c) TwitterPara (d) SciDocs

Figure 2: Comparison of different pre-training approaches (TSDAE/MLM/CT/SimCSE+SBERT) with increasing sizes of labeled training data (in thousands). SBERT: Training from the standard *BERT-base-uncased* checkpoint. TSDAE: Unsupervised baseline. Larger plots: Appendix E.



(a) AskUbuntu (b) CQADupStack



(c) TwitterPara (d) SciDocs

Figure 3: The influence of the number of training sentences (in thousands) on the model performance. Larger plots: Appendix G.

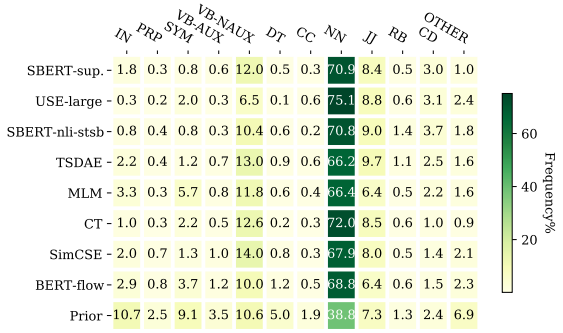


Figure 4: POS tag for the most relevant content word in a sentence, i.e. the word that mostly influences if a sentence pair is considered as similar. VB-AUX/-NAUX represents auxiliary/non-auxiliary verbs.

at the end of each epoch and the best score on the development set is reported.

The results are shown in Figure 3. We observe that TSDAE is outperforming previous unsupervised learning methods often with as little as 1000 unlabeled sentences. With 10K unlabeled sentences, the downstream performance usually stagnates for all tested unsupervised sentence embedding methods. The only exception where more training data is helpful is for the CQADupStack task. This is expected, as the CQADupStack consists of 12 vastly different StackExchange forums, hence, requiring more unlabeled data to represent all domains well.

We conclude that comparatively little unlabeled data of ∼10K sentences is needed to tune pre-trained transformers to a specific domain.

7.2 Relevant Content Words

Not all word types play an equal role in determining the semantics of a sentence. Often, nouns are the critical content words in a sentence, while e.g. prepositions are less important and can be add / removed from a sentences without changing the content too much.

In this section, we investigate which word types are the most relevant for the different sentence embedding methods, i.e., which words (part-of-speech tags) mainly influence if a sentence pair is perceived as similar or not. We are especially interested if we observe differences between in-domain supervised approaches (SBERT-sup.), out-of-the-box pre-trained approaches, and unsupervised approaches.

To measure this, we select a sentence pair (a, b) that is labeled as relevant and find the word that maximally reduces the cosine-similarity score for the pair (a, b):

$$\hat{w} = \text{argmax}_w \big(\text{cossim}(a, b) - \min(\text{cossim}(a \setminus w, b), \text{cossim}(a, b \setminus w))\big)$$

among all words w that appear in either a or b. Then, we record the POS tag for $\hat{w}$ and compute the distribution of POS tags across all sentence pairs.

Checkpoint	AskU.	CQADup.	TwitterP.	SciDocs	Avg.
BERT-base	59.4/2.2	14.5/3.4	73.0/2.4	74.0/2.8	55.2/2.7
Scratch	56.6/2.6	8.4/4.2	69.8/3.3	67.2/3.5	50.5/3.4
BART-base	58.5/1.4	9.5/2.0	60.3/1.5	62.0/1.7	47.6/1.7
T5-base	45.6/1.0	2.2/1.4	48.2/1.5	30.8/1.1	31.7/1.3

Table 4: Test performance/training loss of TSDAE models starting from different checkpoints. The results for BERT-base are copied from Table 2.

POS-tags are determined using CoreNLP (Manning et al., 2014).

The result averaged over the four datasets is shown in Figure 4. For the result on each dataset, please refer to Appendix H. Comparing the in-domain supervised model (SBERT-sup.) and the prior distribution of the POS tags, we find that nouns (NN) are by far the most relevant content words in a sentence, while function words such as prepositions (IN) and determinators (DT) have little influence on the model prediction. Surprisingly, we do not perceive significant differences between all the approaches. This is good news for the unsupervised methods (TSDAE, CT, SimCSE and BERT-flow) and show that they can learn which words types are critical in a sentence without access to labeled data. On the down side, unsupervised approaches might have issues for tasks where nouns are not the most critical content words.

8 Discussion

We mainly experiment with pre-trained Transformer encoders in this work. Besides single encoders, there are also pre-trained encoder-decoder models like BART (Lewis et al., 2020) and T5 (Raffel et al., 2020). However, they are already extensively pre-trained with variants of auto-encoder loss on the general domain and they are suspected of overfitting the reconstruction behavior. To verify this idea, we also further train BART-base and T5-base models with TSDAE on the 4 domain-specific datasets. The results are shown in Table 4. We observe that BART and T5 can achieve much lower training loss (1.7 and 1.3 on average, resp.) than from scratch (3.4) or BERT (2.7), but they achieve rather bad test performance, even worse than from scratch. Compared with training from scratch (which is similar to Zhang et al. (2018)), on the other hand, we find starting from BERT can reach to a much better balance point between loss fitting and generalization. Thus, we conclude that TSDAE is more suitable to start from single encoder checkpoints, which can utilize the pre-trained knowledge while avoiding overfitting.

9 Conclusion

In this work, we propose a new unsupervised sentence embedding learning method based on pre-trained Transformers and sequential deoising auto-encoder (TSDAE). We evaluate TSDAE on other, recent state-of-the-art unsupervised learning on four different tasks from heterogeneous (specialized) domains in three different settings: *unsupervised learning*, *domain adaptation*, and *pre-training*.

We observe that TSDAE performs well on the selected tasks and for the different settings, significantly outperforming other approaches.

Further, we show that the current evaluation of unsupervised sentence embedding learning approach, which is primarily done on the Semantic Textual Similarity (STS) task, is insufficient: A strong performance on STS does not correlate with a good performance on specific tasks. Many recent unsupervised approaches are not able to outperform out-of-the-box pre-trained models on the selected tasks.

Acknowledgments

This work has been supported by the German Research Foundation (DFG) as part of the UKP-SQuARE project (grant GU 798/29-1), by the European Regional Development Fund (ERDF) and the Hessian State Chancellery – Hessian Minister of Digital Strategy and Development under the promotional reference 20005482 (TexPrax), and by the German Federal Ministry of Education and Research and the Hessian Ministry of Higher Education, Research, Science and the Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE.

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 632–642. The Association for Computational Linguistics.

Fredrik Carlsson, Magnus Sahlgren, Evangelia Gogoulou, Amaru Cuba Gyllensten, and Erik Ylipää Hellqvist. 2021. Semantic re-tuning with contrastive

tension. In *International Conference on Learning Representations, ICLR 2021, Vienna, Austria, May 3-7, 2021*, pages 1–21. International Conference on Learning Representations.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 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. Association for Computational Linguistics.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for english. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pages 169–174. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S. Weld. 2020. SPECTER: document-level representation learning using citation-informed transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2270–2282. Association for Computational Linguistics.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*.

John Giorgi, Osvald Nitski, Bo Wang, and Gary Bader. 2021. DeCLUTR: Deep contrastive learning for unsupervised textual representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 879–895, Online. Association for Computational Linguistics.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), 17-22 June 2006, New York, NY, USA*, pages 1735–1742. IEEE Computer Society.

Matthew L. Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient Natural Language Response Suggestion for Smart Reply. *arXiv preprint arXiv:1705.00652*.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1367–1377. The Association for Computational Linguistics.

Doris Hoogeveen, Karin M. Verspoor, and Timothy Baldwin. 2015. Cqadupstack: A benchmark data set for community question-answering research. In *Proceedings of the 20th Australasian Document Computing Symposium, ADCS 2015, Parramatta, NSW, Australia, December 8-9, 2015*, pages 3:1–3:8. ACM.

Diederik P. Kingma and Prafulla Dhariwal. 2018. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 10236–10245.

Wuwei Lan, Siyu Qiu, Hua He, and Wei Xu. 2017. A continuously growing dataset of sentential paraphrases. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 1224–1234. Association for Computational Linguistics.

Tao Lei, Hrishikesh Joshi, Regina Barzilay, Tommi S. Jaakkola, Kateryna Tymoshenko, Alessandro Moschitti, and Lluís Màrquez. 2016. Semi-supervised question retrieval with gated convolutions. In *NAACL HLT 2016, The 2016 Conference of the*

North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016, pages 1279–1289. The Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 9119–9130. Association for Computational Linguistics.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 528–540. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Nils Reimers, Philip Beyer, and Iryna Gurevych. 2016. Task-oriented intrinsic evaluation of semantic textual similarity. In *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 87–96. ACL.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Proceedings of The Third Text REtrieval Conference, TREC 1994, Gaithersburg, Maryland, USA, November 2-4, 1994*, volume 500-225 of *NIST Special Publication*, pages 109–126. National Institute of Standards and Technology (NIST).

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. 2021. Whitening Sentence Representations for Better Semantics and Faster Retrieval. *arXiv preprint arXiv:2103.15316*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Wei Xu, Chris Callison-Burch, and Bill Dolan. 2015. Semeval-2015 task 1: Paraphrase and semantic similarity in twitter (PIT). In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2015, Denver, Colorado, USA, June 4-5, 2015*, pages 1–11. The Association for Computer Linguistics.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernández Ábrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2020. Multilingual universal sentence encoder for semantic retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, ACL 2020, Online, July 5-10, 2020*, pages 87–94. Association for Computational Linguistics.

Minghua Zhang, Yunfang Wu, Weikang Li, and Wei Li. 2018. Learning universal sentence representations with mean-max attention autoencoder. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4514–4523. Association for Computational Linguistics.

A Optimal Configuration of TSDAE

To obtain the optimal configuration, we compare TSDAE models trained and evaluated on the general domain without bias towards any specific domain. The greedy search is applied by sequentially finding the best (1) noise type and ratio (2) pooling method and (3) weight tying scheme. Similar to the choice of CT and BERT-flow, we train the models on the combination of SNLI and MultiNLI without labels and evaluate the models on the STS benchmark with the metric of Spearman rank correlation. The maximum number of training steps is 30K and the models are evaluated every 1.5K training steps, reporting the best validation performance. Scores are obtained by calculating the average over 5 random seeds.

We first compare the scores of different noise types, fixing the noise ratio as 0.3 (i.e. 30% tokens are influenced) and the pooling method as CLS pooling. The results are show in Table 5. This indicates deletion is the best noise type. We then tune the noise ratio of the deletion noise and the results are shown in Table 6. This indicates 0.6 is the best noise ratio.

Type	Delete	Swap	Mask	Replace	Add
Score	**78.33**	76.85	76.56	74.01	72.65

Table 5: Results with different noise types

Ratio	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Score	77.81	77.70	77.75	78.02	78.25	**78.77**	78.19	77.69	75.67

Table 6: Results with different noise ratio.

We then compare different pooling methods with the best setting so far. The results are shown in Table 7. Since there is little difference between CLS and mean pooling and mean pooling loses the position information, the CLS pooling is chosen. Finally, we find that tying the encoder and the decoder can further improve the validation score to 79.15.

Method	CLS	Mean	Max
Score	78.77	**78.84**	78.17

Table 7: Results with different pooling methods.

B Experiment Settings

We implement TSDAE, CT and BERT-flow based on Pytorch and Huggingface's Transformers[6] (version number: v3.1.0). For these three unsupervised methods, following the original papers, the number of training steps is 100K; the batch size is 8; the optimizers are AdamW, RMSProp and AdamW, respectively; the initial learning rates are 3e-5, 1e-5 and 1e-6, resp. The weight decay for BERT-flow is 0.01. The learning rate for CT follows a segmented-constant scheduling scheme: 1e-5 for step 1 to 500; 8e-6 for step 501 to 1000; 6e-6 for step 1001 to 1500; 4e-6 for step 1501 to 2000; 2e-6 for others. The pooling method for CT and BERT-flow is both mean pooling. Since CT trains two independent encoders and we find the second encoder has better performance, we use the second encoder for evaluation. For SimCSE, since its official hyper-parameter setting is very different from the other 3 methods, we use the official code[7] along with the default hyper-parameters. In detail, its hyper-parameters are: 1 epoch of training, batch size of 512, AdamW optimizer with learning rate 5e-5 and a linear layer on the CLS token embedding as the pooling method.

Since in the real-world scenario where the labeled data is expensive to obtain, applying early-stopping with a in-domain development set is impractical. Thus, in our unsupervised experiments, we do not use

[6] https://github.com/huggingface/transformers
[7] https://github.com/princeton-nlp/SimCSE

early-stopping with in-domain labeled data and indicate a fixed number of training steps[8] mentioned above instead.

We use the repository of sentence-transformers[9] (version number: v0.3.8) to train the in-domain supervised models. For them, the number of training epochs is 10; the maximum number of training steps is 20K; the batch size is 64; the similarity function σ is set to cosine similarity; early-stopping is applied by checking the validation performance. To eliminate the influence of randomness, we report the scores averaged over 5 random seeds for all the in-domain unsupervised and supervised models. All the pre-trained checkpoints used are listed in Table 8.

For BM25, we use the implementation available on Elasticsearch[10] with the default settings.

Model Name	**URL**
DeCLUTR-base	https://huggingface.co/johngiorgi/declutr-base
ELECTRA-base	https://huggingface.co/google/electra-base-discriminator
DistilRoBERTa-base	https://huggingface.co/distilroberta-base
RoBERTa-base	https://huggingface.co/roberta-base
DistilBERT-base	https://huggingface.co/distilbert-base-uncased
BERT-base	https://huggingface.co/bert-base-uncased
SBERT-base-nli-v2	https://huggingface.co/kwang2049/SBERT-base-nli-v2
SBERT-base-nli-stsb-v2	https://huggingface.co/kwang2049/SBERT-base-nli-stsb-v2
SDRoBERTa-para	https://huggingface.co/sentence-transformers/paraphrase-distilroberta-base-v1
USE-large	https://tfhub.dev/google/universal-sentence-encoder-multilingual-large/3
BART-base	https://huggingface.co/facebook/bart-base
T5-base	https://huggingface.co/t5-base

Table 8: Model checkpoints used in this work.

C Results of Other Checkpoints

The results of other checkpoints besides *BERT-base-uncased* are shown in Table 9. For all the methods, better results are achieved by using BERT checkpoints, which also makes TSDAE significantly outperforms others. We suppose this advantage comes from the additional pre-training task, next sentence prediction of the BERT models, which guides the model to learn from sentence-level contexts.

[8]For SimCSE, the official code involves early-stopping on the STS-B development set. We do not change this setting for this method, since STS-B is not an in-domain dataset in our task- and domain-specific evaluation.

[9]https://github.com/UKPLab/sentence-transformers

[10]https://www.elastic.co/

Method	AskU.	CQADup.	TwitterP.	SciDocs	Avg.
ELECTRA-base					
TSDAE	56.6 +/- 1.1	8.0 +/- 0.3	69.0 +/- 1.6	66.2 +/- 5.6	49.9 +/- 1.1
CT	50.3 +/- 0.4	5.0 +/- 0.2	66.5 +/- 0.7	46.1 +/- 0.6	41.6 +/- 0.8
SimCSE	50.9 +/- 0.3	6.2 +/- 0.1	61.8 +/- 0.6	49.3 +/- 0.3	42.0 +/- 0.1
BERT-flow	51.3 +/- 0.3	5.2 +/- 0.0	62.4 +/- 0.1	41.2 +/- 0.1	38.4 +/- 3.0
DistilRoBERTa-base					
TSDAE	58.9 +/- 0.5	12.5 +/- 0.1	68.5 +/- 0.5	59.3 +/- 0.6	49.9 +/- 0.4
CT	57.9 +/- 0.8	13.8 +/- 0.3	63.6 +/- 1.4	62.7 +/- 0.5	49.8 +/- 0.4
SimCSE	57.1 +/- 0.2	12.2 +/- 0.1	65.6 +/- 0.8	63.4 +/- 0.4	49.6 +/- 0.2
BERT-flow	56.0 +/- 0.2	11.1 +/- 0.1	68.5 +/- 0.1	53.0 +/- 0.2	46.9 +/- 0.1
RoBERTa-base					
TSDAE	58.3 +/- 0.7	12.2 +/- 0.3	70.0 +/- 0.8	61.4 +/- 0.5	50.3 +/- 0.3
CT	56.7 +/- 0.5	14.2 +/- 0.3	69.4 +/- 1.3	63.1 +/- 0.3	50.5 +/- 0.4
SimCSE	56.6 +/- 0.5	12.4 +/- 0.2	66.8 +/- 0.7	64.4 +/- 0.3	50.1 +/- 0.2
BERT-flow	54.5 +/- 0.2	10.5 +/- 0.1	69.0 +/- 0.1	53.5 +/- 0.2	46.6 +/- 0.1
DistilBERT-base					
TSDAE	59.2 +/- 0.3	**14.6 +/- 0.1**	**73.9 +/- 0.3**	72.3 +/- 0.9	54.9 +/- 0.2
CT	57.7 +/- 0.8	14.0 +/- 0.3	66.4 +/- 0.4	72.2 +/- 0.7	52.3 +/- 0.3
SimCSE	54.8 +/- 0.7	12.3 +/- 0.1	66.8 +/- 0.6	65.9 +/- 0.1	49.9 +/- 0.3
BERT-flow	55.0 +/- 0.2	11.0 +/- 0.0	65.9 +/- 0.0	70.5 +/- 0.1	50.5 +/- 0.1
BERT-base					
TSDAE	**59.4 +/- 0.3**	14.5 +/- 0.1	73.0 +/- 0.4	**74.0 +/- 0.4**	**55.2 +/- 0.2**
CT	56.3 +/- 0.7	13.3 +/- 0.3	72.5 +/- 0.5	67.6 +/- 0.4	52.4 +/- 0.3
SimCSE	55.9 +/- 0.8	12.4 +/- 0.0	68.5 +/- 0.0	65.7 +/- 0.0	50.6 +/- 0.2
BERT-flow	53.7 +/- 0.2	9.2 +/- 0.1	69.3 +/- 0.2	64.5 +/- 0.1	49.2 +/- 0.1

Table 9: Evaluation of different checkpoints using average precision. '+/-' separates the mean value and standard deviation over scores of 5 random seeds. Best results within each group are underlined and the overall best results are bold.

D Equivalent Labeling Work

The goal of unsupervised sentence embedding learning methods is to eliminate the need of labeled training data, which can be expensive in the creation. However, as shown in Section 6, approaches with sufficient in-domain labeled data significantly outperform unsupervised approaches.

As far as we know, previous work did not study the point of intersection between unsupervised and supervised approaches: If you only need few labeled examples to outperform unsupervised approaches, annotating those might be the more viable solution.

To find this intersection point, we train the in-domain supervised SBERT approach with varying size of labeled training data. Results are shown in Figure 5. To estimate the intersection with more precision, we apply binary search. We set the search precision to the standard deviation of the target score over 5 random seeds.

The results are shown in Table 10. To match the performance of TSDAE, 140 - 6k annotated examples are required. CQADupStack and the TwitterParaphrase corpus, which compromise various domains, require more labeled data than AskUbuntu (1 domain). Surprisingly, SciDocs, which includes data from all type of scientific domains, the in-domain supervised approach outperforms unsupervised approaches with just 464 labeled examples. This dataset appears to be especially challenging for unsupervised approaches, as we observe a large performance gap between in-domain supervised and unsupervised approaches.

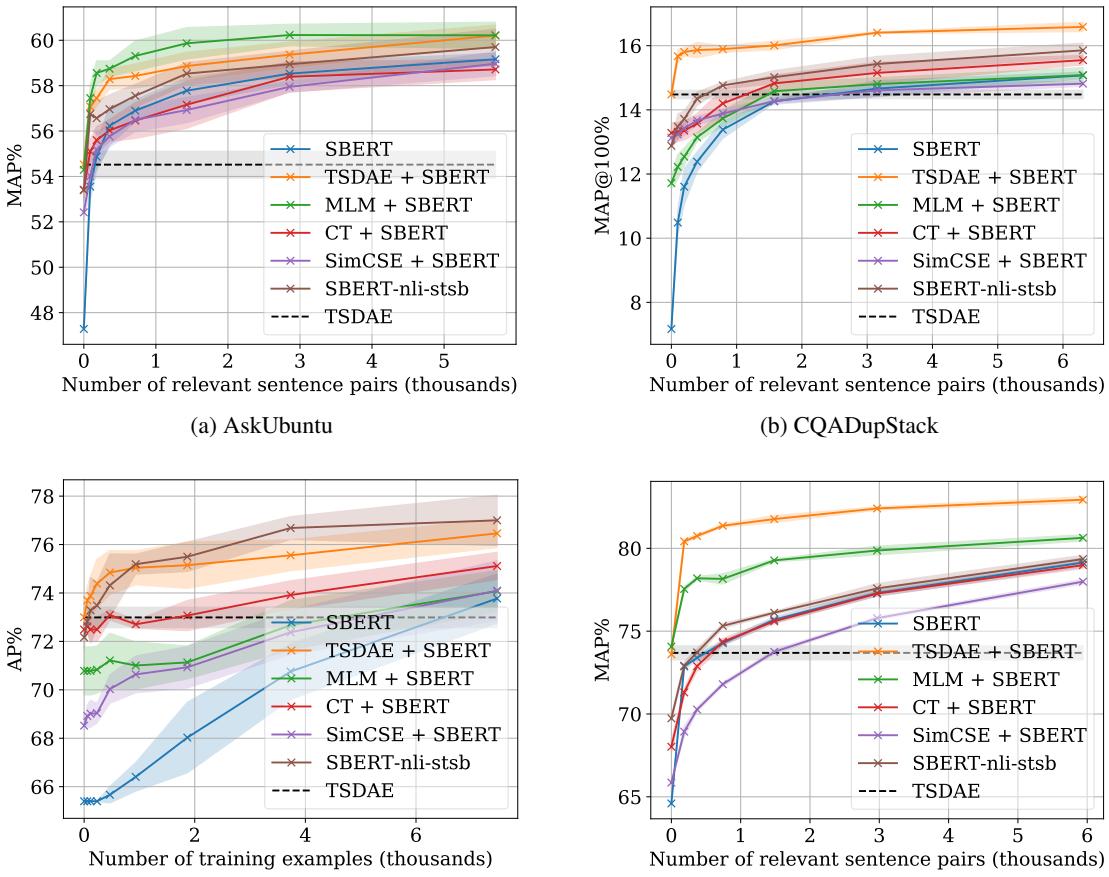
In an annotation experiment on the Twitter dataset, we measured that annotating 100 Tweet pairs takes about 20 minutes for an (experienced) annotator. Hence, the state-of-the-art unsupervised TSDAE approach achieves the same performance as a supervised approach with 0.5 - 20 hours of annotation work for one annotator (2.5h - 100h for 5 crowd annotators).

AskU.	CQADup.	TwitterP.	SciDocs	Avg.
140	2661	6067	464	2333

Table 10: Intersection point (number of labeled sentence pairs) between unsupervised TSDAE and in-domain supervised SBERT.

E Usage for Pre-Training

The pre-training performance on AskUbuntu, CQADupStack and TwitterPara is shown in Figure 5.



(a) AskUbuntu

(b) CQADupStack

(c) TwitterPara

(d) SciDocs

Figure 5: The influence of the number of training sentences on the model performance.

F Detailed Results of Semantic Textual Similarity

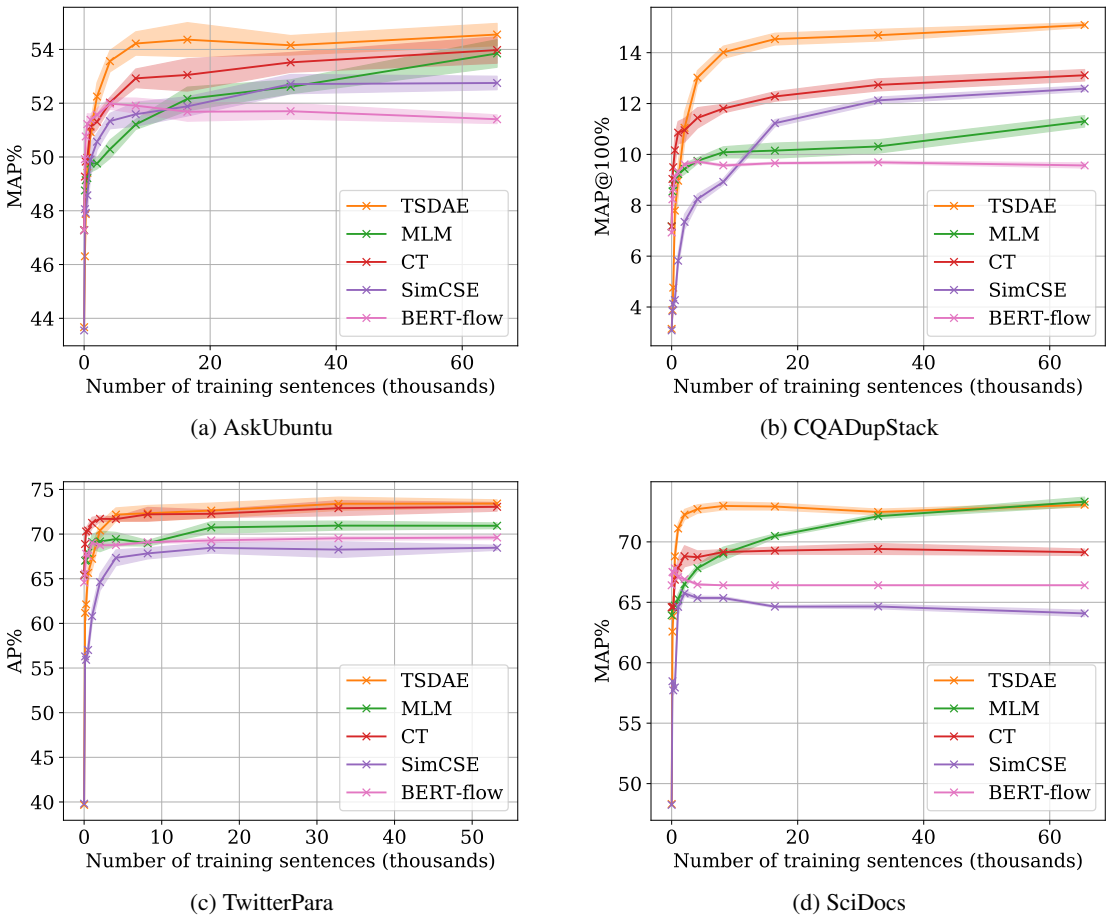
The detailed results of STS on each dataset are shown in Table 11 with the evaluation metric of Spearman's rank correlation. Note that the training set of STSb contains subsets of STS12-16, Thus, we do not include the scores of SBERT-base-nli-stsb-v2 on these datasets for reducing misunderstanding.

Method	STS12	STS13	STS14	STS15	STS16	STSb	SICK-R	Avg.
Unsupervised method based on BERT-base								
TSDAE	55.2	67.4	62.4	74.3	73.0	66.0	62.3	65.8
CT	60.0	76.3	68.2	77.3	75.8	73.9	69.4	71.6
SimCSE	63.6	79.3	69.6	78.2	77.7	73.8	70.1	73.2
BERT-flow	34.1	60.7	48.8	61.9	64.8	48.9	58.4	53.9
MLM	30.9	59.9	47.7	60.3	63.7	47.3	58.2	52.6
Out-of-the-box supervised pre-trained models								
SBERT-base-nli-v2	72.5	**84.8**	**80.2**	**84.8**	**80.0**	83.9	78.0	**80.6**
SBERT-base-nli-stsb-v2	–	–	–	–	–	**87.3**	**80.4**	–
USE-large	**74.3**	71.8	71.4	82.5	77.5	80.9	75.8	76.3

Table 11: Evaluation on the task of STS using Spearman's rank correlation.

G Influence of Corpus Size

The influence of corpus size for AskUbuntu, CQADupStack and TwitterPara is shown in Figure 6.



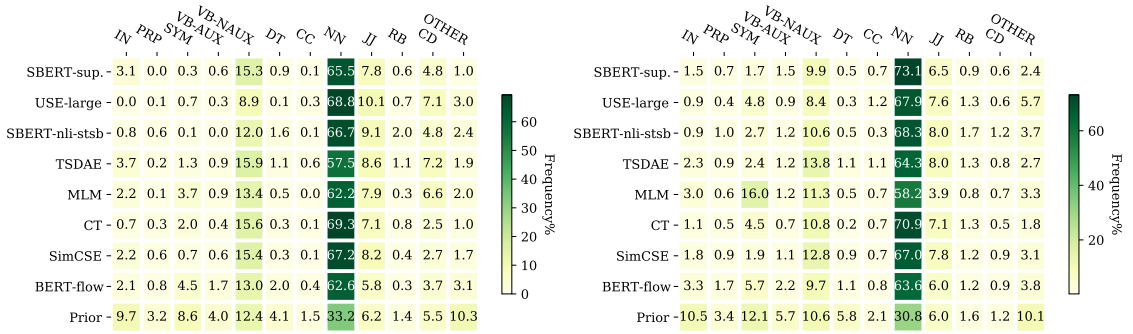
(a) AskUbuntu (b) CQADupStack

(c) TwitterPara (d) SciDocs

Figure 6: The influence of the number of training sentences on the model performance.

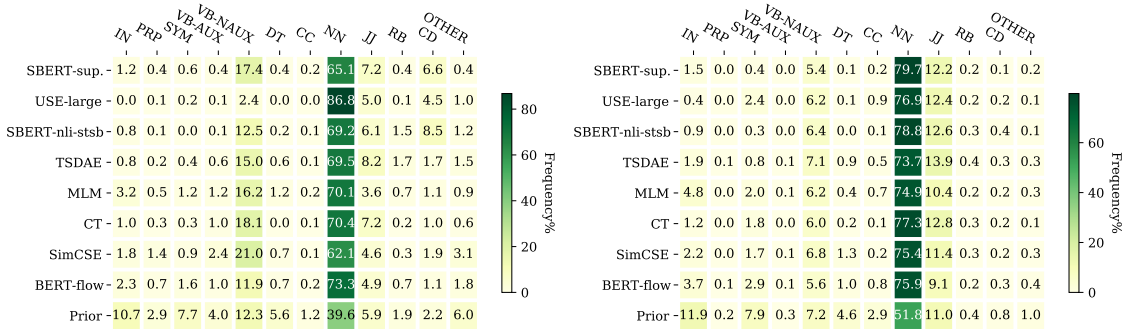
H Influence of Different POS Tags

The influence of different POS tags on the output similarity scores for AskUbuntu, CQADupStack and TwitterPara is shown in Figure 7.



(a) AskUbuntu

(b) CQADupStack



(c) TwitterPara

(d) SciDocs

Figure 7: The influence of different POS tags on the output similarity scores.