

---

# DeCLUTR: Deep Contrastive Learning for Unsupervised Textual Representations

---

**John M. Giorgi**

University of Toronto  
Department of Computer Science  
The Donnelly Centre  
Vector Institute  
Toronto, ON M5G 1M1, Canada  
john.giorgi@utoronto.ca

**Osvald Nitski**

University of Toronto  
Department of Mechanical and Industrial Engineering  
Peter Munk Cardiac Center, University Health Network  
Toronto, ON M5S 3G8, Canada  
osvald.nitski@mail.utoronto.ca

**Gary D. Bader**

University of Toronto  
Department of Molecular Genetics  
Department of Computer Science  
The Donnelly Centre  
Toronto, ON M5S 3E1, Canada  
gary.bader@utoronto.ca

**Bo Wang**

University of Toronto  
Peter Munk Cardiac Center, University Health Network  
CIFAR AI Chair, Vector Institute  
Toronto, ON M5G 1M1, Canada  
bowang@vectorinstitute.ai

## Abstract

We present DeCLUTR: Deep Contrastive Learning for Unsupervised Textual Representations, a self-supervised method for learning universal sentence embeddings that transfer to a wide variety of natural language processing (NLP) tasks. Our objective leverages recent advances in deep metric learning (DML) and has the advantage of being conceptually simple and easy to implement, requiring no specialized architectures or labelled training data. We demonstrate that our objective can be used to pretrain transformers to state-of-the-art performance on SentEval, a popular benchmark for evaluating universal sentence embeddings, outperforming existing supervised, semi-supervised and unsupervised methods. We perform extensive ablations to determine which factors contribute to the quality of the learned embeddings. Our code will be publicly available and can be easily adapted to new datasets or used to embed unseen text.

## 1 Introduction

Due to the limited amount of labelled training data available for many natural language processing (NLP) tasks, transfer learning has become ubiquitous [1]. For some time, transfer learning in NLP was limited to pretrained *word*-level embeddings, such as word2vec [2] or GloVe [3]. Recent work has demonstrated strong transfer task performance using pretrained *sentence*-level embeddings. These fixed-length vectors, often referred to as “universal” sentence embeddings, are typically learned on a large text corpus and then transferred for use in a variety of downstream tasks (such as sentiment analysis or semantic search). Indeed, high-quality universal sentence embeddings have become an area of focus, and many supervised [4], semi-supervised [5, 6, 7] and unsupervised [8, 9, 10, 11, 12] approaches have been proposed. However, the highest performing solutions require at least some *labelled* data, limiting their usefulness to languages and domains where labelled data exists for the chosen pretraining tasks. Therefore, closing the gap in performance between unsupervised and supervised universal sentence embedding methods is an important goal.

Pretraining transformer-based language models has become the primary method for learning textual representations from unlabelled corpora [13, 14, 15, 16]. This success has primarily been driven by masked language modelling (MLM), a self-supervised, *token*-level objective that requires the model to predict the identity of some randomly masked tokens from the input sequence. In addition to MLM, some of these models have mechanisms for learning *sentence*-level embeddings via self-supervision. In BERT [14], for instance, a special classification token is prepended to every input sequence, and its representation is used in a binary classification task to predict whether one textual segment follows another in the training corpus, denoted Next Sentence Prediction (NSP). However, recent work has called in to question the effectiveness of NSP [17, 18, 19] and in RoBERTa (Robustly Optimized BERT Pretraining Approach) [16], the authors demonstrated that removing NSP during pretraining leads to the same or slightly improved performance on downstream tasks (including question answering, semantic text similarity and natural language inference). In ALBERT (A Lite BERT) [20], the authors hypothesize that, because the textual segments of negative examples are sampled from different documents, NSP conflates *topic* prediction and *coherence* prediction. They propose swapping NSP for a Sentence-Order Prediction objective (SOP), which they suggest better models inter-sentence coherence. In preliminary evaluations, we found that neither objective produces good universal sentence embeddings (see Supplementary Material, ??). Drawing from recent advances in metric learning, we propose a simple and effective self-supervised, sentence-level objective for pretraining transformer-based language models.

Metric learning, a type of representation learning, aims to learn an embedding space where the embedded vector representations of similar data are mapped close together, and those of dissimilar data far apart [21, 22, 23]. In the computer vision (CV) literature, deep metric learning (DML) has been used to great effect for learning *visual* representations [24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34]. Generally speaking, DML is approached as follows: a *pretext* task (often self-supervised, e.g. colourization or inpainting) is carefully designed and used to train deep neural networks, with the goal of generating useful feature representations. Here, “useful” means a representation that is easily adaptable to other *downstream* tasks, unknown at training time. Downstream tasks (e.g. object recognition) are then used to evaluate the quality of the learned features, often by training a linear classifier on the task using these features as input. The most successful approach to-date has been to design a pretext task for learning with a pair-based *contrastive* loss function. For a given *anchor* data point, contrastive losses attempt to make the distance between the anchor and some *positive* data points (i.e. those that are similar) smaller than the distance between the anchor and some *negative* data points (i.e. those that are dissimilar) [35]. The highest-performing methods generate anchor-positive pairs by randomly augmenting the *same* image (e.g. using crops, flips and colour distortions); anchor-negative pairs are randomly chosen, augmented views of *different* images [36, 37, 38].

Inspired by this setup, we propose a self-supervised, contrastive objective that can be used alongside MLM to pretrain a transformer. Our objective learns universal sentence embeddings by training an encoder to minimize the distance between the embeddings of textual segments randomly sampled from nearby in the same document. We demonstrate that a model pretrained with our objective obtains state-of-the-art results on SentEval [39] – a benchmark of 28 tasks designed to evaluate the quality of universal sentence embeddings – outperforming existing supervised, semi-supervised and unsupervised solutions (subsection 5.1). We perform extensive ablations to determine which factors are important for learning high-quality embeddings (section 5). Finally, we open-source our solution and provide detailed instructions for training it on new data or embedding unseen text.

## 2 Related Work

Previous works on universal sentence embeddings can broadly be grouped into supervised, semi-supervised or unsupervised approaches.

**Supervised or semi-supervised.** The most successful universal sentence embedding methods are pretrained on the (human-labelled) natural language inference (NLI) datasets Stanford NLI (SNLI) [40] and MultiNLI [41]. NLI is the task of classifying a pair of sentences (denoted the “hypothesis” and the “premise”) into one of three relationships: *entailment*, *contradiction* or *neutral*. The effectiveness of NLI for training universal sentence encoders was demonstrated by the supervised method InferSent [4]. Google’s Universal Sentence Encoder (USE) is semi-supervised, augmenting supervised learning on SNLI with a mix of unsupervised objectives trained on unlabelled text.

The recently published Sentence Transformers [7] method fine-tunes pretrained, transformer-based language models like BERT [14] using labelled NLI datasets.

**Unsupervised** Skip-thought [10] and FastSent [11] are two popular unsupervised techniques that learn sentence embeddings by using an encoding of a sentence to predict words in neighbouring sentences. However, in addition to being computationally expensive, this generative objective forces the model to reconstruct the *surface* form of a sentence, which may produce representations that capture aspects irrelevant to the *meaning* of a sentence. QuickThoughts [12] addresses both of these shortcomings with a simple discriminative objective; given a sentence and its context (adjacent sentences), it learns sentence representations by training a classifier to distinguish context sentences from non-context sentences. The unifying theme of these unsupervised approaches is that they exploit the distributional hypothesis, namely that the meaning of a word (and by extension, a sentence) is characterized by the word-context in which it appears.

Our overall approach is most similar to Sentence Transformers – we pretrain a transformer-based language model to produce useful sentence embeddings – but our proposed objective is *self-supervised*, allowing us to exploit the vast amount of unlabelled text on the web, without being restricted to languages or domains where labelled data exist. Our objective most closely resembles QuickThoughts, with three distinctions: we relax our sampling to textual *segments* (rather than natural *sentences*), we sample one or more positive segments per anchor (rather than strictly one), and we allow these segments to be adjacent, overlapping or subsuming (rather than strictly adjacent; see Figure 1, B).

### 3 Model

#### 3.1 Self-supervised contrastive loss

Our method learns textual *representations* via a contrastive loss by maximizing agreement between textual *segments* (referred to as “spans” in the rest of the paper) sampled from nearby in the same document. As illustrated in Figure 1, this approach comprises the following components:

- A data loading step which randomly samples paired anchor-positive spans from each document in a minibatch of size  $N$ . Let  $A$  be the number of anchor spans sampled *per document*,  $P$  be the number of positive spans sampled *per anchor* and  $i \in \{1 \dots AN\}$  be the index of an arbitrary anchor span. Then we denote an anchor span and its corresponding  $p \in \{1 \dots P\}$  positive spans as  $s_i$  and  $s_{i+pAN}$  respectively. The sampling procedure is designed to maximize the chance of sampling semantically similar anchor-positive pairs (see subsection 3.1.1).
- An encoder  $f(\cdot)$ , which maps each token in the input spans to a word embedding. Although our method places no constraints on the choice of encoder, we restrict ourselves to transformer-based language models, as this represents the state-of-the-art for text encoders (see subsection 3.2).
- A pooler  $g(\cdot)$ , which maps the encoded spans  $f(s_i)$  and  $f(s_{i+pAN})$  to fixed-length embeddings  $e_i = g(f(s_i))$  and its corresponding mean positive embedding

$$e_{i+AN} = \frac{1}{P} \sum_{p=1}^P g(f(s_{i+pAN}))$$

Similar to [7], we found that choosing  $g(\cdot)$  to be the mean of the token-level embeddings (referred to as “mean pooling” in the rest of the paper) performs well (see Supplementary Material, ??). Pairing each anchor embedding with the *mean* of multiple positive embeddings was proposed in [42], who demonstrated both theoretical and empirical improvements when compared to using a single positive example for every anchor.

- A contrastive loss function defined for a contrastive prediction task. Given a set of embedded spans  $\{e_k\}$  including a positive pair of examples  $e_i$  and  $e_{i+AN}$ , the contrastive prediction task aims to identify  $e_{i+AN}$  in  $\{e_k\}_{k \neq i}$  for a given  $e_i$ .

During training, we randomly sample a minibatch of  $N$  documents from the train set and define the contrastive prediction task on anchor-positive pairs  $e_i, e_{i+AN}$  derived from the  $N$  documents,

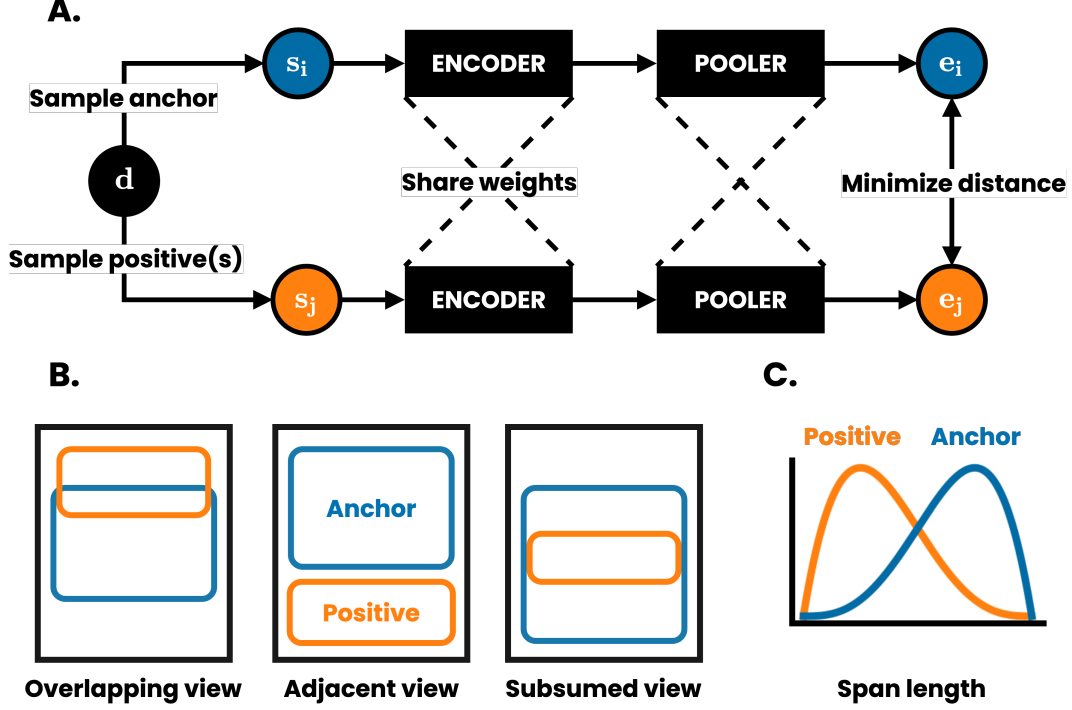


Figure 1: Overview of the self-supervised contrastive objective. (A) For each document  $d$  in a minibatch of size  $N$ , we sample  $A$  anchor spans per document and  $P$  positive spans per anchor. For simplicity, we illustrate the case where  $A = P = 1$  and denote the anchor-positive span pair as  $s_i, s_j$ . Both spans are fed through the same encoder  $f(\cdot)$  and pooler  $g(\cdot)$  to produce the corresponding embeddings  $e_i = g(f(s_i)), e_j = g(f(s_j))$ . The encoder and pooler are trained to minimize the distance between embeddings via a contrastive prediction task (where the other embeddings in a minibatch are treated as negatives, omitted here for simplicity). (B) Positive spans can overlap with, be adjacent to or be subsumed by the sampled anchor span. (C) The length of anchors and positives are randomly sampled from beta distributions, skewed toward longer and shorter spans, respectively.

resulting in  $2AN$  data points. As proposed in [43], we treat the other  $2(AN - 1)$  instances within a minibatch as negative examples. The cost function takes the following form

$$\mathcal{L}_{\text{contrastive}} = \sum_{i=1}^{AN} \ell(i, i + AN) + \ell(i + AN, i) \quad (1)$$

$$\ell(i, j) = -\log \frac{\exp(\text{sim}(\mathbf{e}_i, \mathbf{e}_j)/\tau)}{\sum_{k=1}^{2AN} \mathbb{1}_{[i \neq k]} \cdot \exp(\text{sim}(\mathbf{e}_i, \mathbf{e}_k)/\tau)} \quad (2)$$

where  $\text{sim}(\mathbf{u}, \mathbf{v}) = \mathbf{u}^T \mathbf{v} / \|\mathbf{u}\|_2 \|\mathbf{v}\|_2$  denotes the cosine similarity of two vectors  $\mathbf{u}$  and  $\mathbf{v}$ ,  $\mathbb{1}_{[i \neq k]} \in \{0, 1\}$  is an indicator function evaluating to 1 if  $i \neq k$ , and  $\tau > 0$  denotes the temperature hyperparameter. This is exactly the InfoNCE loss used in previous works [44, 45, 46] and denoted normalized temperature-scale cross-entropy loss or “NT-Xent” in [38]. Notice that when  $\tau = 1$  and  $\text{sim}(\mathbf{u}, \mathbf{v}) = \mathbf{u}^T \mathbf{v}$ , the loss function closely resembles the loss used in QuickThoughts.

To embed text with a trained model, we simply pass batches of tokenized text through the model, without sampling spans. Therefore, the computational cost of our method at test time is the cost of the encoder,  $f(\cdot)$ , plus the cost of the pooler,  $g(\cdot)$ , which is negligible when using mean pooling.

### 3.1.1 Span sampling

We start by choosing a minimum and maximum span length; in this paper,  $\ell_{\min} = 32$  and  $\ell_{\max} = 512$ , the maximum input size for many pretrained transformers. Next, a document  $d$  is tokenized to produce a sequence of  $n$  tokens  $\mathbf{x}^d = (x_1, x_2 \dots x_n)$ . To sample an anchor span  $s_i$  from  $\mathbf{x}^d$ , we first sample its length  $\ell_{\text{anchor}}$  from a beta distribution and then randomly (uniformly) sample its starting position  $s_i^{\text{start}}$ . We then sample  $p \in \{1 \dots P\}$  corresponding positive spans  $s_{i+pAN}$  independently following a similar procedure:

$$\begin{aligned} \ell_{\text{anchor}} &= \lfloor p_{\text{anchor}} \times (\ell_{\max} - \ell_{\min}) + \ell_{\min} \rfloor & \ell_{\text{positive}} &= \lfloor p_{\text{positive}} \times (\ell_{\max} - \ell_{\min}) + \ell_{\min} \rfloor \\ s_i^{\text{start}} &\sim \{0 \dots n - \ell_{\text{anchor}}\} & s_{i+pAN}^{\text{start}} &\sim \{s_i^{\text{start}} - \ell_{\text{positive}} \dots s_i^{\text{end}}\} \\ s_i^{\text{end}} &= s_i^{\text{start}} + \ell_{\text{anchor}} & s_{i+pAN}^{\text{end}} &= s_{i+pAN}^{\text{start}} + \ell_{\text{positive}} \\ \mathbf{s}_i &= \mathbf{x}_{s_i^{\text{start}}:s_i^{\text{end}}}^d & \mathbf{s}_{i+pAN} &= \mathbf{x}_{s_{i+pAN}^{\text{start}}:s_{i+pAN}^{\text{end}}}^d \end{aligned}$$

where  $p_{\text{anchor}} \sim \text{Beta}(\alpha = 4, \beta = 2)$ , which skews the anchor sampling towards longer spans, and  $p_{\text{positive}} \sim \text{Beta}(\alpha = 2, \beta = 4)$ , which skews the positive sampling towards shorter spans (Figure 1, C). In practice, we restrict the sampling of anchor spans from the same document such that they are a minimum of  $\ell_{\max}$  tokens apart. In Supplementary Materials ??, we show several examples of text that has been sampled by our method.

We note several carefully considered decisions in the design of our sampling procedure:

- Sampling span lengths from a distribution clipped at  $\ell_{\min} = 32$  and  $\ell_{\max} = 512$  encourages the model to produce good embeddings for text ranging from sentence- to paragraph-length. Thus, at test time, we expect our model to be able to embed up-to paragraph-length texts.
- We found that sampling longer lengths for the anchor span than the positive spans improves performance in downstream tasks (we did not find performance to be sensitive to the specific choice of  $\alpha$  and  $\beta$ ). The rationale for this is twofold: it enables the model to learn global-to-local view prediction as in [47, 36, 38] (referred to as “subsumed view” in Figure 1, B); when  $P > 1$ , it encourages diversity among positives by lowering the probability of repeated text between positive spans.
- The restriction that positives must be sampled nearby to the anchor exploits the distributional hypothesis and increases the chances of sampling valid (i.e. semantically related) anchor-positive pairs.
- By sampling multiple anchors per document (i.e.  $A > 1$ ), each anchor-positive pair is contrasted against both *easy* negatives (anchors and positives sampled from *other* documents in a minibatch) and *hard* negatives (anchors and positives sampled from the *same* document).

In conclusion, the sampling procedure produces *three* types of positives: positives that partially overlap with the anchor, positives adjacent to the anchor, and positives subsumed by the anchor (Figure 1, B) and *two* types of negatives: easy negatives sampled from a different document than the anchor, and hard negatives sampled from the same document as the anchor. Thus, our stochastically generated training set and contrastive loss implicitly define a family of predictive tasks which can be used to train a model, independent of any specific encoder architecture.

### 3.2 Continued MLM pretraining

We use our objective to pretrain a transformer-based language model [48], as this represents the state-of-the-art text encoder in NLP. We implement the MLM objective as described in [14] on each anchor span in a minibatch and sum the losses from the MLM and contrastive objectives before backpropagating.

$$\mathcal{L} = \mathcal{L}_{\text{contrastive}} + \mathcal{L}_{\text{MLM}} \quad (3)$$

This is similar to the approaches of other pretraining strategies, such as those used in [14, 20], in which an MLM loss is paired with a sentence-level loss objective (e.g. NSP or SOP) and used to

pretrain the model. To make the computational requirements feasible, we do not train from scratch, but rather we continue training a model that has been pretrained with the MLM objective. Specifically, we use both RoBERTa-base [16] and DistilRoBERTa [49] (a distilled version of RoBERTa-base) in our experiments. In the rest of the paper, we refer to our method as DeCLUTR-small (when our method is used to pretrain DistilRoBERTa) and DeCLUTR-base (when our method is used to pretrain RoBERTa-base).

## 4 Experimental setup

### 4.1 Dataset, training, and implementation

**Dataset** We collected all documents with a minimum token length of 2048 from an open-access subset of the OpenWebText corpus [50], yielding 495,243 documents in total. For reference, Google’s USE was trained on 570,000 human-labelled sentence pairs from the SNLI dataset. InferSent and Sentence Transformer models were trained on both SNLI and MultiNLI, for a total of 1 million human-labelled sentence pairs.

**Implementation** We implemented our model in PyTorch [51] using AllenNLP [52]. We used the NT-Xent loss function implemented by the PyTorch Metric Learning library [53] and the pretrained transformer architecture and weights from the Transformers library [54]. All models were trained on up to four NVIDIA Tesla V100 16 or 32GB GPUs.

**Training** Unless specified otherwise, we train for 1 epoch over the 495,243 documents with a minibatch size of 16 and a temperature  $\tau = 5 \times 10^{-2}$  using the AdamW optimizer [55] with a learning rate (LR) of  $5 \times 10^{-5}$  and a weight decay of 0.1. For every document in a minibatch, we sample two anchor spans ( $A = 2$ ), and two positive spans per anchor ( $P = 2$ ). We use the Slanted Triangular LR scheduler [56] with a number of train steps equal to training instances and a cut fraction of 0.1. All gradients are scaled to a vector norm of 1.0 before backpropagating.

### 4.2 Evaluation

We evaluate the quality of our learned sentence embeddings on the SentEval benchmark, a widely-used toolkit for evaluating general-purpose, fixed-length sentence representations. SentEval is divided into 18 *downstream* tasks – representative NLP tasks such as sentiment analysis, natural language inference, paraphrase detection and image-caption retrieval – and 10 *probing* tasks, which are designed to evaluate what linguistic properties are encoded in a sentence representation (see Supplementary Materials ?? for details). We report averaged scores obtained by our model and the relevant baselines on both the downstream and probing tasks. In Supplementary Material ??, we report a detailed breakdown of the scores for each task.

#### 4.2.1 Baselines

We compare to the highest performing and most popular sentence encoding methods, i.e. InferSent, Google’s USE and Sentence Transformers. For InferSent, we compare to the latest model (V2) trained with fastText word vectors.<sup>1</sup> We use the latest “large” USE model<sup>2</sup> and the “roberta-base-nli-mean-tokens”<sup>3</sup> Sentence Transformers model for comparison, as they are most similar in terms of architecture and number of parameters to our own. We include the performance of averaged GloVe<sup>4</sup> and fastText<sup>5</sup> word vectors as weak baselines. Despite a good-faith attempt, we were not able to evaluate the pretrained QuickThought models against the full SentEval benchmark. We cite the scores from the paper directly in Supplementary Material ?. Finally, we evaluate the performance of the pretrained transformer model *before* it is subjected to training with our contrastive objective, denoted “Transformer-\*”. We use mean pooling on the pretrained transformers token-level output to produce fixed-length sentence embeddings – the same pooling strategy used in our method.

<sup>1</sup><https://dl.fbaipublicfiles.com/infersent/infersent2.pkl>

<sup>2</sup><https://tfhub.dev/google/universal-sentence-encoder-large/5>

<sup>3</sup><https://github.com/UKPLab/sentence-transformers#english-pretrained-models>

<sup>4</sup><http://nlp.stanford.edu/data/glove.840B.300d.zip>

<sup>5</sup><https://dl.fbaipublicfiles.com/fasttext/vectors-english/crawl-300d-2M.vec.zip>

Table 1: Results on the downstream and probing tasks from the test set of the SentEval benchmark. USE: Google’s Universal Sentence Encoder. Transformer-small and Transformer-base are pretrained DistilRoBERTa and RoBERTa-base models respectively, using mean pooling. DeCLUTR-small and DeCLUTR-base are pretrained DistilRoBERTa and RoBERTa-base models respectively after continued pretraining with our method. Bold: best scores.  $\Delta$ : difference to our methods (DeCLUTR-base) average score.

| Model                                    | Parameters | Embed. dim. | SentEval     |              |              |          |
|--|------------|-------------|--------------|--------------|--------------|----------|
|  |            |             | Downstream   | Probing      | Avg.         | $\Delta$ |
| <i>Bag-of-words (BoW) weak baselines</i> |            |             |              |              |              |          |
| GloVe                                    | –          | 300         | 65.50        | 62.42        | 63.96        | -12.02   |
| fastText                                 | –          | 300         | 68.57        | 63.16        | 65.87        | -10.11   |
| <i>Supervised and semi-supervised</i>    |            |             |              |              |              |          |
| InferSent                                | 38M        | 4096        | 76.46        | 72.58        | 74.52        | -1.46    |
| USE                                      | 147M       | 512         | <b>79.13</b> | 66.70        | 72.91        | -3.06    |
| Sentence Transformers                    | 125M       | 768         | 77.59        | 63.22        | 70.40        | -5.57    |
| <i>Unsupervised</i>                      |            |             |              |              |              |          |
| Transformer-small                        | 82M        | 768         | 72.69        | <b>74.27</b> | 73.48        | -2.50    |
| Transformer-base                         | 125M       | 768         | 72.22        | 73.38        | 72.80        | -3.18    |
| DeCLUTR-small (ours)                     | 82M        | 768         | 76.80        | 73.84        | 75.32        | -0.66    |
| DeCLUTR-base (ours)                      | 125M       | 768         | 78.16        | 73.80        | <b>75.98</b> | –        |

## 5 Results

In subsection 5.1, we compare the performance of our model against the relevant baselines. We use the remaining sections to explore which components contribute to the quality of the learned embeddings.

### 5.1 Comparison to existing methods

Except for InferSent, we find that existing methods perform poorly on the probing tasks of the SentEval benchmark (Table 1). Sentence Transformers, which begins with a pretrained transformer model and fine-tunes it on NLI datasets, scores approximately 10% lower on the probing tasks than the model it fine-tunes (see Supplementary Material, ??). In contrast, both DeCLUTR-small and DeCLUTR-base significantly boost downstream task performance while maintaining high probing task performance, producing the highest average score. Although our method sometimes underperforms existing supervised solutions on average downstream performance, we found that this is partially explained by the fact that these methods are trained on the SNLI corpus, which is included as a downstream evaluation task in SentEval. If the average downstream performance is computed *without* considering SNLI, the difference in downstream performance between our method and the supervised methods shrinks considerably (see Supplementary Material, ??).

### 5.2 Ablation of the sampling procedure

We ablate several components of the sampling procedure, including the number of anchors sampled per document  $A$ , the number of positives sampled per anchor  $P$ , and the sampling strategy for those positives (Figure 2). First, we note that sampling multiple anchors per document has a large positive impact on the quality of the learned embeddings. We hypothesize this is because the difficulty of the contrastive objective increases when  $A > 1$ . Recall that a minibatch is composed of random documents, and each anchor-positive pair sampled from a document is contrasted against all other anchor-positive pairs in the minibatch. When  $A > 1$ , anchor-positive pairs will be contrasted against other anchors and positives from the same document, increasing the difficulty of the contrastive objective, thus leading to better representations. Secondly, we find that a positive sampling strategy that allows positives to be adjacent to *and* subsumed by the anchor outperforms a strategy which only allows adjacent *or* subsuming views, suggesting that the information captured by these views is

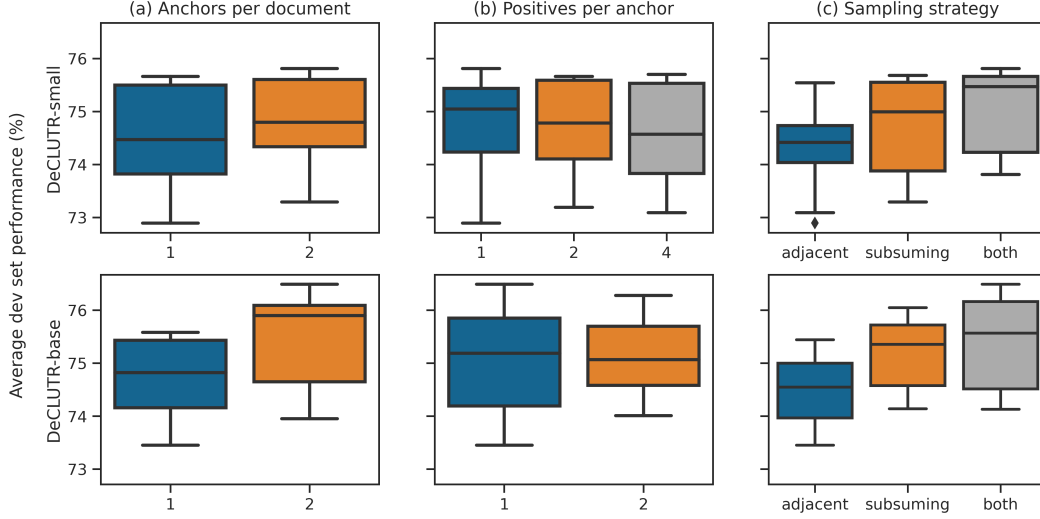


Figure 2: Effect of the number of anchor spans sampled per document (a), the number of positive spans sampled per anchor (b), and the sampling strategy (c) on SentEval performance. Performance is computed over a grid of hyperparameters and plotted as a distribution. Hyperparameters are chosen from the grid defined by all permutations of number of anchors  $A = \{1, 2\}$ , number of positives  $P = \{1, 2, 4\}$ , and temperatures  $\tau = \{5 \times 10^{-4}, 5 \times 10^{-2}\}$ .  $P = 4$  is omitted for DeCLUTR-base as these experiments did not fit into GPU memory.

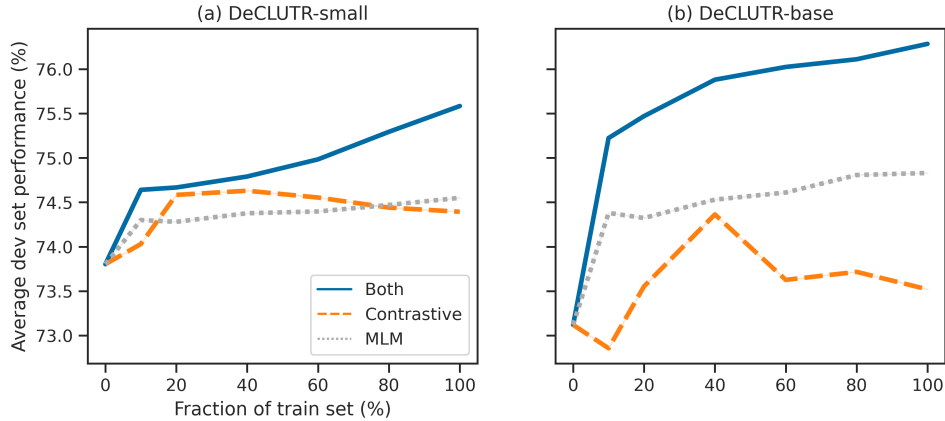


Figure 3: Effect of training objective, train set size and model capacity on SentEval performance. DeCLUTR-small has 6 layers and  $\sim 82\text{M}$  parameters; DeCLUTR-base has 12 layers and  $\sim 125\text{M}$  parameters. Average scores are reported from the development set of the SentEval benchmark. 100% corresponds to 1 epoch of training with all 495,243 documents from our OpenWebText subset.

complementary. Finally, we note that sampling multiple positives per anchor ( $P > 1$ ) has minimal impact on performance. This is in contrast to [42], who found both theoretical and empirical improvements when multiple positives are averaged and paired with a given anchor.

### 5.3 Training objective, train set size and model capacity

To determine the importance of the training objectives, train set size, and model capacity on the quality of the learned embeddings, we trained two sizes of the model while artificially reducing the size of the train set (Figure 3). First, we note that pretraining the model with both the MLM and contrastive objectives improves performance over training with either objective alone. In particular, the performance of the model on SentEval when using only the contrastive objective actually peaks at



40% of the train set. In contrast, including the MLM objective alongside the contrastive objective leads to monotonic improvement as the train set size is increased. Our results suggest that the quality of the learned embeddings could be further improved by scaling the train set size; because the training method is completely self-supervised, this would simply involve collecting more unlabelled text.

## 6 Discussion and conclusion

In this paper, we proposed a self-supervised objective for learning universal sentence representations. Our objective is conceptually simple, easy to implement, and applicable to any text encoder. We demonstrated the effectiveness of our objective by evaluating the learned sentence representations on the SentEval benchmark, which contains a total of 28 tasks designed to evaluate the transferability and linguistic properties of sentence representations. When used to pretrain a transformer-based language model, our objective achieves new state-of-the-art performance, outperforming existing supervised, semi-supervised and unsupervised methods. Our experiments suggest that the quality of the learned embeddings can be further improved by scaling the model capacity and train set size. Finally, we will release our model publicly in the hopes that it will be extended to new domains (e.g. biomedical literature) and non-English languages.

## References

- [1] Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [2] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.
- [3] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.
- [4] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*, 2017.
- [5] Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher J Pal. Learning general purpose distributed sentence representations via large scale multi-task learning. *arXiv preprint arXiv:1804.00079*, 2018.
- [6] 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. Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, Brussels, Belgium, November 2018. Association for Computational Linguistics.
- [7] Nils Reimers and Iryna Gurevych. 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 2019.
- [8] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196, 2014.
- [9] Yacine Jernite, Samuel R Bowman, and David Sontag. Discourse-based objectives for fast unsupervised sentence representation learning. *arXiv preprint arXiv:1705.00557*, 2017.
- [10] Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302, 2015.
- [11] Felix Hill, Kyunghyun Cho, and Anna Korhonen. Learning distributed representations of sentences from unlabelled data. *arXiv preprint arXiv:1602.03483*, 2016.
- [12] Lajanugen Logeswaran and Honglak Lee. An efficient framework for learning sentence representations. *ArXiv*, abs/1803.02893, 2018.

- [13] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. URL [https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsupervised/language\\_understanding\\_paper.pdf](https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsupervised/language_understanding_paper.pdf), 2018.
- [14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [15] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764, 2019.
- [16] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
- [17] Guillaume Lample and Alexis Conneau. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*, 2019.
- [18] Yang You, Jing Li, Jonathan Hseu, Xiaodan Song, James Demmel, and Cho-Jui Hsieh. Reducing bert pre-training time from 3 days to 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019.
- [19] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, 2020.
- [20] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.
- [21] David G Lowe. Similarity metric learning for a variable-kernel classifier. *Neural computation*, 7(1):72–85, 1995.
- [22] Sebastian Mika, Gunnar Ratsch, Jason Weston, Bernhard Scholkopf, and Klaus-Robert Mullers. Fisher discriminant analysis with kernels. In *Neural networks for signal processing IX: Proceedings of the 1999 IEEE signal processing society workshop (cat. no. 98th8468)*, pages 41–48. Ieee, 1999.
- [23] Eric P Xing, Michael I Jordan, Stuart J Russell, and Andrew Y Ng. Distance metric learning with application to clustering with side-information. In *Advances in neural information processing systems*, pages 521–528, 2003.
- [24] Paul Wohlhart and Vincent Lepetit. Learning descriptors for object recognition and 3d pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3109–3118, 2015.
- [25] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*, pages 499–515. Springer, 2016.
- [26] Ziming Zhang and Venkatesh Saligrama. Zero-shot learning via joint latent similarity embedding. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6034–6042, 2016.
- [27] Maxime Bucher, Stéphane Herbin, and Frédéric Jurie. Improving semantic embedding consistency by metric learning for zero-shot classification. In *European Conference on Computer Vision*, pages 730–746. Springer, 2016.
- [28] Laura Leal-Taixé, Cristian Canton-Ferrer, and Konrad Schindler. Learning by tracking: Siamese cnn for robust target association. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 33–40, 2016.
- [29] Ran Tao, Efstratios Gavves, and Arnold WM Smeulders. Siamese instance search for tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1420–1429, 2016.
- [30] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

- [31] Xinwei He, Yang Zhou, Zhichao Zhou, Song Bai, and Xiang Bai. Triplet-center loss for multi-view 3d object retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1945–1954, 2018.
- [32] Alexander Grabner, Peter M Roth, and Vincent Lepetit. 3d pose estimation and 3d model retrieval for objects in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3022–3031, 2018.
- [33] Sasi Kiran Yelamarthi, Shiva Krishna Reddy, Ashish Mishra, and Anurag Mittal. A zero-shot framework for sketch based image retrieval. In *European Conference on Computer Vision*, pages 316–333. Springer, 2018.
- [34] Rui Yu, Zhiyong Dou, Song Bai, Zhaoxiang Zhang, Yongchao Xu, and Xiang Bai. Hard-aware point-to-set deep metric for person re-identification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 188–204, 2018.
- [35] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*, volume 2, pages 1735–1742. IEEE, 2006.
- [36] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems*, pages 15509–15519, 2019.
- [37] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.
- [38] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.
- [39] Alexis Conneau and Douwe Kiela. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*, 2018.
- [40] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.
- [41] Adina Williams, Nikita Nangia, and Samuel Bowman. 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, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics, 2018.
- [42] Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*, 2019.
- [43] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*, pages 1857–1865, 2016.
- [44] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*, pages 1857–1865, 2016.
- [45] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018.
- [46] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.
- [47] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *Proceeding of the International Conference on Learning Representations*, 2019.
- [48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.
- [49] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

- [50] Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. <http://Skylion007.github.io/OpenWebTextCorpus>, 2019.
- [51] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.
- [52] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia, July 2018. Association for Computational Linguistics.
- [53] Kevin Musgrave, Ser-Nam Lim, and Serge Belongie. Pytorch metric learning. <https://github.com/KevinMusgrave/pytorch-metric-learning>, 2019.
- [54] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R’emi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface’s transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.
- [55] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [56] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia, July 2018. Association for Computational Linguistics.

---

# Supplementary Material for *DeCLUTR: Deep Contrastive Learning for Unsupervised Textual Representations*

---

**John M. Giorgi**

University of Toronto  
Department of Computer Science  
The Donnelly Centre  
Vector Institute  
Toronto, ON M5G 1M1, Canada  
john.giorgi@utoronto.ca

**Osvald Nitski**

University of Toronto  
Department of Mechanical and Industrial Engineering  
Peter Munk Cardiac Center, University Health Network  
Toronto, ON M5S 3G8, Canada  
osvald.nitski@mail.utoronto.ca

**Gary D. Bader**

University of Toronto  
Department of Molecular Genetics  
Department of Computer Science  
The Donnelly Centre  
Toronto, ON M5S 3E1, Canada  
gary.bader@utoronto.ca

**Bo Wang**

University of Toronto  
Peter Munk Cardiac Center, University Health Network  
CIFAR AI Chair, Vector Institute  
Toronto, ON M5G 1M1, Canada  
bowang@vectorinstitute.ai

This document contains supplementary materials for the paper *DeCLUTR: Deep Contrastive Learning for Unsupervised Textual Representations*.

## 1 Pretrained transformers make poor universal sentence encoders

Some pretrained transformers, such as BERT (Bidirectional Encoder Representations from Transformers) [1] and ALBERT (A Lite BERT) [2], already have mechanisms for learning sequence-level embeddings via self-supervision. These models prepend every input sequence with a special classification token (often “[CLS]” is used) and its representation is learned using a simple classification task, such as Next Sentence Prediction (NSP) or Sentence-Order Prediction (SOP) (see [1] and [2] respectively for more information on these tasks).

However, during preliminary experiments, we noticed that these models are not good universal sentence encoders, as measured by their performance on SentEval [3]. As a simple experiment, we evaluated three pretrained transformer models on SentEval: one trained with the NSP loss (BERT), one trained with the SOP loss (ALBERT) and one trained with neither, RoBERTa (Robustly Optimized BERT Pretraining Approach) [4]. We did not find that the CLS embeddings produced by models trained against the NSP or SOP losses to outperform that of a model trained without either loss and sometimes failed to outperform a bag-of-words (BoW) weak baseline (Table S1). Furthermore, we find that pooling token embeddings via averaging (referred to as “mean pooling” in the main paper) outperforms pooling via the CLS classification token. Our results are corroborated by [4], who find that removing NSP loss leads to the same or better results on downstream tasks and [5], who find that directly using the output of BERT as sentence embeddings leads to poor performances on a suite of semantic similarity tasks.

Table S1: Results on the downstream and probing tasks from the development set of the SentEval benchmark. We compare models trained with the Next Sentence Prediction (NSP) and Sentence-Order Prediction (SOP) losses to a model trained with neither, using two different pooling strategies: "\*-CLS", where the special classification token prepended to every input is used as its sentence-level representation, and "\*-mean", where each sentence is represented by the mean of its token embeddings.

| Model  | SentEval   |             |              |              |              |
|--|------------|-------------|--------------|--------------|--------------|
|  | Parameters | Embed. dim. | Downstream   | Probing      | Avg.         |
| <i>Bag-of-Words (BoW) weak baselines</i>                 |            |             |              |              |              |
| GloVe  | –          | 300         | 66.05        | 62.93        | 64.49        |
| fastText   | –          | 300         | 68.75        | 63.46        | 66.11        |
| <i>Trained with Next Sentence Prediction (NSP) loss</i>  |            |             |              |              |              |
| BERT-base-CLS  | 110M       | 768         | 63.54        | 69.57        | 66.55        |
| BERT-base-mean   | 110M       | 768         | 72.00        | 73.37        | 72.68        |
| <i>Trained with Sentence-Order Prediction (SOP) loss</i> |            |             |              |              |              |
| ALBERT-base-V2-CLS                                       | 11M        | 768         | 58.81        | 69.88        | 64.35        |
| ALBERT-base-V2-mean                                      | 11M        | 768         | 69.63        | <b>74.83</b> | 72.23        |
| <i>Trained with neither NSP or SOP losses</i>            |            |             |              |              |              |
| RoBERTa-base-CLS   | 125M       | 768         | 68.04        | 66.35        | 67.20        |
| RoBERTa-base-mean  | 125M       | 768         | <b>72.39</b> | 73.85        | <b>73.12</b> |

## 2 Examples of sampled spans

In Table S2, we present examples of anchor-positive and anchor-negative pairs generated by our sampling procedure. We show one example for each possible view of a sampled positive, e.g. positives adjacent to, overlapping with, or subsumed by the anchor. For each anchor-positive pair, we show examples of both a *hard* negative (derived from the same document) and an *easy* negative (derived from another document). Recall that a minibatch is composed of random documents, and each anchor-positive pair sampled from a document is contrasted against all other anchor-positive pairs in the minibatch. Thus, hard negatives, as we have described them here, are generated only when sampling multiple anchors per document ( $A > 1$ ).

Table S2: Examples of text spans generated by our sampling procedure. During training, we randomly sample one or more anchors from every document in a minibatch. For each anchor, we randomly sample one or more positives adjacent to, overlapping with, or subsumed by the anchor. All anchor-positive pairs are contrasted with every other anchor-positive pair in the minibatch. This leads to *easy* negatives (anchors and positives sampled from *other* documents in a minibatch) and *hard* negatives (anchors and positives sampled from the *same* document). Here, examples are capped at a maximum length of 64 word-tokens. During training, we sample spans up-to a length of 512 word tokens.

| Anchor   | Positive  | Hard negative  | Easy negative  |
|--|---|--|--|
| <i>Overlapping view</i>  |   |  |  |
| immigrant-rights advocates and law enforcement professionals were skeptical of the new program. Any effort by local cops to enforce immigration laws, they felt, would be bad for community policing, since immigrant victims or witnesses of crime wouldn't feel comfortable talking to police.   | feel comfortable talking to police. Some were skeptical that ICE's intentions were really to protect public safety, rather than simply to deport unauthorized immigrants more easily.     | liberal parts of the country with large immigrant populations, like Santa Clara County in California and Cook County in Illinois, agreed with the critics of Secure Communities. They worried that implementing the program would strain their relationships with immigrant residents. | that a new location is now available for exploration. A good area, in my view, feels like a natural progression of a game world it doesn't seem tacked on or arbitrary. That in turn needs it to relate  |
| <i>Adjacent view</i>   |   |  |  |
| if the ash stops belching out of the volcano then, after a few days, the problem will have cleared, so that's one of the factors. "The other is the wind speed and direction." At the moment the weather patterns are very volatile which is what is making it quite difficult, unlike last year, to predict                                   | where the ash will go. "The public can be absolutely confident that airlines are only able to operate when it is safe to do so." Ryanair said it could not see any ash cloud              | A British Airways jumbo jet was grounded in Canada on Sunday following fears the engines had been contaminated with volcanic ash   | events are processed in FIFO order. When this nextTickQueue is emptied, the event loop considers all operations to have been completed for the current phase and transitions to the next phase.  |
| <i>Subsumed view</i>   |   |  |  |
| Far Cry Primal is an action-adventure video game developed by Ubisoft Montreal and published by Ubisoft. It was released worldwide for PlayStation 4 and Xbox One on February 23, 2016, and for Microsoft Windows on March 1, 2016. The game is a spin-off of the main Far Cry series. It is the first Far Cry game set in the Mesolithic Age. | by Ubisoft. It was released worldwide for PlayStation 4 and Xbox One on February 23, 2016, and for Microsoft Windows on March 1, 2016. The game is a spin-off of the main Far Cry series. | Players take on the role of a Wenja tribesman named Takkar, who is stranded in Oros with no weapons after his hunting party is ambushed by a Saber-tooth Tiger.  | to such feelings. Fawkes cried out and flew ahead, and Albus Dumbledore followed. Further along the Dementors' path, people were still alive to be fought for. And no matter how much he himself was hurting, while there were still people who needed him he would go on. For |

### 3 SentEval evaluation details

SentEval is a benchmark for evaluating the quality of fixed-length sentence embeddings. It is divided into 18 *downstream* tasks, and 10 *probing* tasks. Sentence embedding methods are evaluated on these tasks via a simple interface<sup>1</sup>, which standardizes training, evaluation and hyperparameters. For most tasks, the method to evaluate is used to produce fix-length sentence embeddings, and a simple logistic regression (LR) or multi-layer perception (MLP) model is trained on the task using these embeddings as input. For other tasks (namely several semantic text similarity tasks), the embeddings are used as-is without any further training.

In subsection 3.1, we present the individual tasks of the SentEval benchmark. In subsection 3.2, we explain our method for computing the average scores across all downstream and all probing tasks presented in the main paper. Finally, in subsection 3.3, we present the scores obtained by our method and the baselines across all 28 SentEval tasks.

#### 3.1 SentEval tasks

The downstream tasks of SentEval are representative NLP tasks used to evaluate the transferability of fixed-length sentence embeddings. We give a brief overview of the broad categories that divide the tasks below (see [3] for more details):

- **Binary and multi-class classification:** These tasks cover various types of sentence classification, including sentiment analysis (MR [6], SST2 and SST5 [7]), question-type (TREC) [8], product reviews (CR) [9], subjectivity/objectivity (SUBJ) [10] and opinion polarity (MPQA) [11].
- **Entailment and semantic relatedness:** These tasks cover multiple entailment datasets (also known as natural language inference or NLI), including SICK-E [12] and the Stanford NLI dataset (SNLI) [13] as well as multiple semantic relatedness datasets including SICK-R and the STS Benchmark [14].
- **Semantic textual similarity** These tasks (STS12 [15], STS13 [16], STS14 [17], STS15 [18] and STS16 [19]) are similar to the semantic relatedness tasks, except the embeddings produced by the encoder are used as-is in a cosine similarity to determine the semantic similarity of two sentences. No additional model is trained on top of the encoder’s output.
- **Paraphrase detection** Evaluated on the Microsoft Research Paraphrase Corpus (MRPC) [20], this binary classification task is comprised of human-labelled sentence pairs, annotated according to whether they capture a paraphrase/semantic equivalence relationship.
- **Caption-Image retrieval** This task is comprised of two sub-tasks: ranking a large collection of images by their relevance for some given query text (Image Retrieval) and ranking captions by their relevance for some given query image (Caption Retrieval). Both tasks are evaluated on data from the COCO dataset [21]. Each image is represented by a pretrained, 2048-dimensional embedding produced by a ResNet-101 [22].

The probing tasks are designed to evaluate what linguistic properties are encoded in a sentence representation. All tasks are binary or multi-class classification. We give a brief overview of each task below (see [23] for more details):

- **Sentence length (SentLen):** A multi-class classification task where a model is trained to predict the length of a given input sentence (binned into six possible length ranges).
- **Word content (WC):** A multi-class classification task where, given 1000 words as targets, the goal is to predict which of the target words appears in a given input sentence. Each sentence contains a single target word, and the word occurs exactly once in the sentence.
- **Tree depth (TreeDepth):** A multi-class classification task where the goal is to predict the maximum depth (with values ranging from 5 to 12) of a given input sentence’s syntactic tree.
- **Bigram Shift (BShift):** A multi-class classification task where the goal is to predict whether two consecutive tokens within a given sentence have been inverted.

---

<sup>1</sup><https://github.com/facebookresearch/SentEval>



Table S3: Results on the downstream tasks from the test set of the SentEval benchmark. USE: Google’s Universal Sentence Encoder. Bold: best scores.

| Model                                    | SentEval (downstream) |              |              |                        |              |              |              |                    |              |
|--|-----------------------|--------------|--------------|------------------------|--------------|--------------|--------------|--------------------|--------------|
|  | CR                    | MR           | MPQA         | SUBJ                   | SST2         | SST5         | TREC         | MRPC               | SNLI         |
| <i>Bag-of-words (BoW) weak baselines</i> |                       |              |              |                        |              |              |              |                    |              |
| GloVe                                    | 78.78                 | 77.70        | 87.76        | 91.25                  | 80.29        | 44.48        | 83.00        | 73.39/81.45        | 65.85        |
| fastText                                 | 79.18                 | 78.45        | 87.88        | 91.53                  | 82.15        | 45.16        | 83.60        | 74.49/82.44        | 68.79        |
| <i>Supervised and semi-supervised</i>    |                       |              |              |                        |              |              |              |                    |              |
| InferSent                                | 84.37                 | 79.42        | 89.04        | 93.03                  | 84.24        | 45.34        | 90.80        | 76.35/83.48        | 84.16        |
| USE                                      | 85.70                 | 79.38        | 88.89        | 93.11                  | 84.90        | 46.11        | <b>95.00</b> | 72.41/82.01        | 83.25        |
| Sentence Transformers                    | <b>91.05</b>          | 84.52        | 89.10        | 92.65                  | <b>90.55</b> | <b>52.76</b> | 87.40        | <b>77.33/84.06</b> | <b>84.18</b> |
| <i>Unsupervised</i>                      |                       |              |              |                        |              |              |              |                    |              |
| Transformer-small                        | 86.81                 | 82.51        | 87.20        | 94.69                  | 87.75        | 49.64        | 91.60        | 73.80/81.14        | 71.81        |
| Transformer-base                         | 88.35                 | 84.07        | 79.90        | <b>95.21</b>           | 88.80        | 50.27        | 93.20        | 74.49/82.46        | 71.62        |
| DeCLUTR-small (ours)                     | 88.03                 | 82.54        | <b>89.86</b> | 94.94                  | 87.59        | 48.42        | 92.40        | 75.54/82.68        | 73.72        |
| DeCLUTR-base (ours)                      | 90.15                 | <b>84.89</b> | 89.39        | 95.19                  | 89.68        | 50.18        | 93.60        | 75.25/82.80        | 74.18        |
| Model                                    | SICK-E                | SICK-R       | STSBenchmark | Caption-ImageRetrieval | STS12        | STS13        | STS14        | STS15              | STS16        |
| GloVe                                    | 78.89                 | 72.30        | 62.86        | 0.40                   | 53.44        | 51.24        | 55.71        | 59.62              | 57.93        |
| fastText                                 | 79.01                 | 72.98        | 68.26        | 0.40                   | 58.85        | 58.83        | 63.42        | 69.05              | 68.24        |
| InferSent                                | <b>86.30</b>          | <b>83.06</b> | 78.48        | <b>65.84</b>           | 62.90        | 56.08        | 66.36        | 74.01              | 72.89        |
| USE                                      | 85.37                 | 81.53        | <b>81.50</b> | 62.42                  | <b>68.87</b> | <b>71.70</b> | <b>72.76</b> | <b>83.88</b>       | <b>82.78</b> |
| Sentence Transformers                    | 82.97                 | 79.17        | 73.31        | 60.97                  | 64.11        | 65.69        | 69.83        | 74.77              | 72.85        |
| Transformer-small                        | 81.98                 | 77.48        | 71.26        | 60.65                  | 54.95        | 46.11        | 57.08        | 65.56              | 63.80        |
| Transformer-base                         | 80.76                 | 76.08        | 69.02        | 60.97                  | 54.32        | 46.41        | 55.64        | 64.04              | 62.77        |
| DeCLUTR-small (ours)                     | 82.69                 | 78.35        | 77.26        | 61.22                  | 64.10        | 65.80        | 67.89        | 75.63              | 72.84        |
| DeCLUTR-base (ours)                      | 82.95                 | 79.39        | 78.84        | 61.92                  | 64.21        | 70.40        | 69.99        | 77.51              | 75.35        |

- **Top Constituents (TopConst):** A multi-class classification task where the goal is to predict the top constituents (from a choice of 19) immediately below the sentence (S) node of the sentence’s syntactic tree.
- **Tense:** A binary classification task where the goal is to predict the tense (past or present) of the main verb in a sentence.
- **Subject number (SubjNum):** A binary classification task where the goal is to predict the number (singular or plural) of the subject of the main clause.
- **Object number (ObjNum):** A binary classification task, analogous to SubjNum, where the goal is to predict the number (singular or plural) of the direct object of the main clause.
- **Semantic odd man out (SOMO):** A binary classification task where the goal is to predict whether a sentence has had a single randomly picked noun or verb replaced with another word with the same part-of-speech.
- **Coordinate inversion (CoordInv):** A binary classification task where the goal is to predict whether the order of two coordinate clauses in a sentence have been inverted.

### 3.2 Computing an average score

In the main paper, we present averaged downstream and probing scores. Computing averaged probing scores was straightforward; each of the 10 probing tasks reports a simple accuracy, which we averaged. To compute an averaged downstream score, we do the following:

- If a task reports Spearman correlation (SICK-R, STS Benchmark) we use this score when computing the average downstream task score. If the task reports a mean Spearman correlation for multiple subtasks (STS12, STS13, STS14, STS15, STS16), we use this score.
- If a task reports both an accuracy and an F1-score (MRPC), we use the average of these two scores.
- For the Caption-Image Retrieval task, we report the average of the Recall@K, where  $K \in \{1, 5, 10\}$  for the Image and Caption retrieval tasks (a total of six scores). This is the default behaviour of SentEval.
- Otherwise, we use the reported accuracy.

### 3.3 Detailed SentEval scores

In Table S3, we report the scores obtained by our method and the relevant baselines for every downstream task in the SentEval benchmark. Similarly in Table S4, we report the scores obtained for the probing tasks.

Table S4: Results on the probing tasks from the test set of the SentEval benchmark. USE: Google’s Universal Sentence Encoder. Bold: best scores.

| Model                                    | SentEval (probing) |              |              |              |              |              |              |              |              |              |
|--|--------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
|  | SentLen            | WC           | TreeDepth    | TopConst     | BShift       | Tense        | SubjNum      | ObjNum       | SOMO         | CoordInv     |
| <i>Bag-of-words (BoW) weak baselines</i> |                    |              |              |              |              |              |              |              |              |              |
| GloVe                                    | 57.82              | 81.10        | 31.41        | 62.70        | 49.74        | 83.58        | 78.39        | 76.31        | 49.55        | 53.62        |
| fastText                                 | 55.46              | 82.10        | 32.74        | 63.32        | 50.16        | 86.68        | 79.75        | 79.81        | 50.21        | 51.41        |
| <i>Supervised and semi-supervised</i>    |                    |              |              |              |              |              |              |              |              |              |
| InferSent                                | 78.76              | <b>89.50</b> | 37.72        | <b>80.16</b> | 61.41        | <b>88.56</b> | <b>86.83</b> | 83.91        | 52.11        | 66.88        |
| USE                                      | 73.14              | 69.44        | 30.87        | 73.27        | 58.88        | 83.81        | 80.34        | 79.14        | 56.97        | 61.13        |
| Sentence Transformers                    | 69.21              | 51.79        | 30.08        | 50.38        | 69.70        | 83.02        | 79.74        | 77.85        | 60.10        | 60.33        |
| <i>Unsupervised</i>                      |                    |              |              |              |              |              |              |              |              |              |
| Transformer-small                        | <b>89.16</b>       | 64.50        | <b>41.21</b> | 73.50        | 88.33        | 87.97        | 85.44        | <b>84.17</b> | 63.80        | 64.65        |
| Transformer-base                         | 82.19              | 57.60        | 38.92        | 70.38        | <b>89.34</b> | 88.32        | 84.70        | 83.33        | <b>68.09</b> | 70.94        |
| DeCLUTR-small (ours)                     | 85.83              | 74.51        | 38.04        | 73.78        | 85.26        | 87.72        | 85.48        | 83.08        | 62.06        | 62.61        |
| DeCLUTR-base (ours)                      | 79.74              | 69.98        | 37.25        | 72.96        | 87.50        | 88.29        | 85.72        | 82.92        | 65.60        | <b>68.03</b> |

Table S5: Results on the downstream and probing tasks from the development set of the SentEval benchmark for a selection of pretrained transformer models, before and after they have been fine-tuned by the Sentence Transformers method. For all models, we use mean pooling on their token-level embeddings to produce fixed-length sentence embeddings. We use the “\*-nli-mean-tokens” pretrained models obtained from the Sentence Transformers GitHub: <https://github.com/UKPLab/sentence-transformers>.  $\Delta$ : difference to the probing task performance of the model before fine-tuning with the Sentence Transformers objective.

| Model   | Parameters | Embed. dim. | SentEval   |         |       |          |
|---|------------|-------------|------------|---------|-------|----------|
|   |            |             | Downstream | Probing | Avg.  | $\Delta$ |
| <i>Before fine-tuning with Sentence Transformer objective</i> |            |             |            |         |       |          |
| BERT-base   | 110M       | 768         | 72.00      | 73.37   | 72.68 | –        |
| BERT-large  | 350M       | 1024        | 71.27      | 71.37   | 71.32 | –        |
| RoBERTa-base  | 125M       | 768         | 72.39      | 73.85   | 73.12 | –        |
| RoBERTa-large   | 355M       | 1024        | 72.02      | 75.05   | 73.54 | –        |
| <i>After fine-tuning with Sentence Transformer objective</i>  |            |             |            |         |       |          |
| BERT-base   | 110M       | 768         | 77.78      | 67.37   | 72.57 | 6.00     |
| BERT-large  | 340M       | 1024        | 79.12      | 64.57   | 71.85 | 6.80     |
| RoBERTa-base  | 125M       | 768         | 77.61      | 63.73   | 70.67 | 10.12    |
| RoBERTa-large   | 355M       | 1024        | 79.23      | 64.02   | 71.63 | 11.03    |

#### 4 Sentence Transformers trades downstream task performance for probing task performance

Sentence Transformers [5] is a method which begins with a pretrained, transformer-based language model and fine-tunes it against NLI datasets to learn universal sentence embeddings. While this approach leads to good performance on the *downstream* tasks of the SentEval benchmark, we noticed that this improvement comes at the cost of significant degradation in performance on the *probing* tasks. In Table S5, we compare the performance of multiple pretrained transformer models on SentEval, before and after fine-tuning with the Sentence Transformers objective, and find that the fine-tuning procedure consistently reduces average probing task performance by at least 6%.

We note that the purpose of the probing tasks is *not* the development of ad-hoc models that attain top performance on them. However, the performance of several *probing* tasks, particularly WC, SOMO and CoordInv is positively correlated with performance on one or more *downstream* tasks [23]. Thus, one of our aims in this paper was to develop a sentence encoder that performs well on *both* types of tasks.

Table S6: Results on the downstream tasks from the development set of the SentEval benchmark. Averaged scores computed *without* considering the performance on the Stanford Natural Language Inference (SNLI) dataset are shown in parenthesis. USE: Google’s Universal Sentence Encoder.  $\Delta$ : difference to our methods (DeCLUTR-base) average score.

| Model                                 | Parameters | Embed. dim. | SentEval (w/o SNLI) |               |
|---------------------------------------|------------|-------------|---------------------|---------------|
|                                       |            |             | Downstream          | $\Delta$      |
| <i>Supervised and semi-supervised</i> |            |             |                     |               |
| InferSent                             | 38M        | 4096        | 76.80 (76.33)       | -1.47 (-2.20) |
| USE                                   | 147M       | 512         | 79.76 (79.56)       | 1.49 (1.03)   |
| Sentence Transformers                 | 125M       | 768         | 77.61 (77.21)       | -0.66 (-1.32) |
| <i>Unsupervised</i>                   |            |             |                     |               |
| DeCLUTR-small (ours)                  | 82M        | 768         | 76.97 (77.18)       | -1.30 (-1.35) |
| DeCLUTR-base (ours)                   | 125M       | 768         | 78.27 (78.53)       | –             |

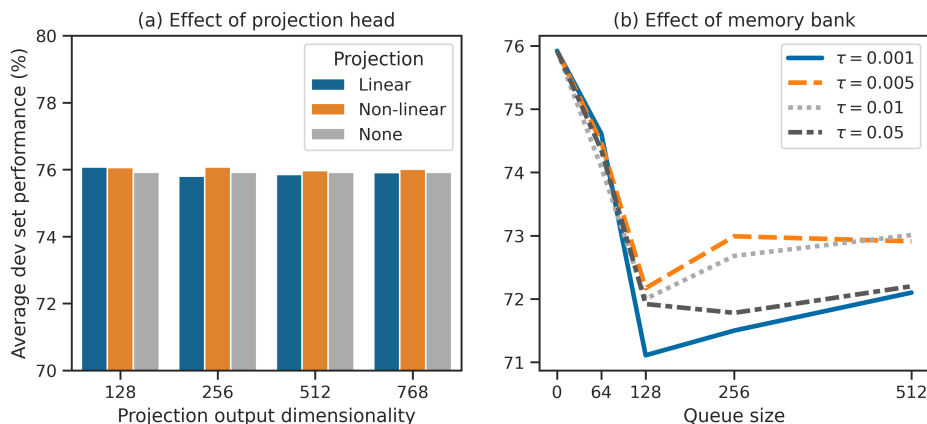


Figure S1: Effect of adding a projection head (a) and a memory bank (b) on SentEval performance. Projection heads are two-layer multi-layer perceptrons (MLPs) inserted between the pooler and the contrastive loss, with either linear or non-linear (ReLU) activations. The projection head is used only during training and is discarded at test time. A memory bank stores instances from previous minibatches, accumulating these instances until the selected queue size is reached. All examples in the memory bank are used as additional negatives during each minibatch. Once full, the oldest examples in the memory bank are replaced with new instances from the current minibatch.  $\tau > 0$ : temperature parameter of the NT-Xent contrastive loss function.

## 5 Controlling for SNLI in downstream performance

The supervised methods we compare to (InferSent, USE and Sentence Transformers) are all trained on (in addition to other data) the Stanford NLI dataset (SNLI) [13], which is included as a downstream evaluation task in SentEval. To control for this, we simply re-computed average downstream performance without including SNLI (see Table S6). In all cases, we found the difference between the supervised methods performance and our unsupervised method to improve. For example, after controlling for SNLI, our method performs only  $\sim 1\%$  worse than Google’s USE. Similarly, our improvement in downstream performance over InferSent and Sentence Transformers improves  $\sim 33\%$  and  $\sim 50\%$ , respectively.

## 6 Methods popular in contrastive learning for computer vision

In this section, we explore two methods that have been highly successful for contrastive learning approaches in computer vision (CV), namely adding a *projection head* between the encoder and

contrastive loss function (subsection 6.1), and increasing the number of negative examples via a *memory bank* (subsection 6.2).

### 6.1 Projection head

SimCLR, a recently proposed method for contrastive learning of visual representations, [24] demonstrated that introducing a non-linear *projection head* between their encoder (a ResNet [22]) and their contrastive loss function (NT-Xent, the same loss function used in our work) substantially improved the performance in a downstream ImageNet [25] Top-1 linear evaluation task. The projection head is a two-layer MLP which maps the output of the encoder  $e_i$  (or pooler, in our case) to a new representation  $z_i$

$$z_i = g(e_i) = W^{(2)}\sigma(W^{(1)}e_i) \quad (1)$$

where  $W^{(1)}, W^{(2)}$  are trainable weight matrices and  $\sigma$  is a ReLU non-linearity. This representation,  $z_i$ , is then used to compute a contrastive loss during training. At test time, the projection head is discarded, and the representation of an image is taken to be  $e_i$ . In a simple experiment, SimCLR demonstrated that the introduction of a projection head at train time improved performance at test time regardless of the projections output dimensionality, and even when the non-linearity was replaced with a linear activation. They conjecture that because  $z_i$  is trained to be *invariant* to data transformation (SimCLR learns to minimize the distance between embeddings of two randomly augmented views of the same image),  $e_i$  is allowed to form and maintain more information. They verify this hypothesis by using either  $z_i$  or  $e_i$  to predict which transformations have been applied to an image (e.g. has the image been rotated? Grayscaled?) and find that  $e_i$  indeed performs substantially better.

In a simple experiment, illustrated in Figure S1 (a), we explore the effect of introducing a projection head at train time on the performance at test time. Our experiment is similar to SimCLR; we add the same projection head (Equation 1) with and without a non-linear activation and compare it to the identity mapping (no projection). We find that the introduction of a non-linear projection head marginally improves performance at every projection dimensionality we tested (up to +0.16%). A linear projection head also marginally improves performance, but only when the output dimensionality is set to 128. We hypothesize that the improvement is so small due to the fact that (besides the sampling of spans), we do not augment or transform our data in any way. Thus, the motivation of the projection head as a mapping that is trained to be invariant to the applied data transformations does not apply to our approach. Due to the marginal improvement and increased number of parameters during training (from  $\sim 100,000$  to  $\sim 1,000,000$  depending on projected output dimensionality), we elected to omit the projection head from our method and main results.

### 6.2 Memory bank

A consistent source of improvement of contrastive learning methods in CV has been to increase the number of negative examples used to compute a pair-based contrastive loss [26, 27, 24]. Often, negative instances for each anchor-positive pair in a minibatch are taken to be all *other* anchors and positives in that minibatch, which couples the number of negative instances to the batch size. One solution is to dramatically scale up compute in order to increase the minibatch size and, therefore, the number of negatives per anchor-positive pair, as was done in SimCLR. Another approach is to introduce a *memory bank* [28], which maintains a *queue* of embedded instances across minibatches. During training, the instances in the queue from the oldest minibatch are progressively replaced with instances from the current minibatch in order to maintain some fixed number of examples (typically denoted the “queue size”). At every minibatch, the contrastive loss is computed using both the negatives from the minibatch *plus* the negatives currently stored in the queue, decoupling the number of negative instances from the minibatch size.

To investigate the impact of scaling up the number of negative instances on our approach, we pair our method with the Cross-Batch Memory (XBM) [27] module. This is a simple memory bank which works as described above: it collects embedded instances across minibatches, progressively replacing instances from the oldest minibatch with the instances from the current minibatch in order to maintain a fixed queue size. During training, we compute the contrastive loss by contrasting each anchor-positive against all negatives instances from the current minibatch *plus* all negative instances

in the queue. We conduct our experiment for queue sizes up to 512 (larger queue sizes did not fit into GPU memory) and for a range of temperature values:  $\tau = \{1 \times 10^{-3}, 5 \times 10^{-3}, 1 \times 10^{-2}, 5 \times 10^{-2}\}$ . Perhaps surprisingly, we find that a queue of any size performs worse than using no queue at all (see Figure S1, (b)). One possible explanation for this is provided by [29], who present a theoretical framework for contrastive learning, where every training instance belongs to some *latent* class. Using this framework, they demonstrate that a large number of negative examples can lead to frequent “class collisions” – scenarios where the anchor-positive pair and corresponding negative example come from the same latent class – which hurts the models ability to learn good representations.

## References

- [1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [2] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.
- [3] Alexis Conneau and Douwe Kiela. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*, 2018.
- [4] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
- [5] Nils Reimers and Iryna Gurevych. 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 2019.
- [6] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 115–124. Association for Computational Linguistics, 2005.
- [7] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.
- [8] Ellen M Voorhees and Dawn M Tice. Building a question answering test collection. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 200–207, 2000.
- [9] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177, 2004.
- [10] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, page 271. Association for Computational Linguistics, 2004.
- [11] Janyce Wiebe, Theresa Wilson, and Claire Cardie. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3):165–210, 2005.
- [12] Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, Roberto Zamparelli, et al. A sick cure for the evaluation of compositional distributional semantic models. In *LREC*, pages 216–223, 2014.
- [13] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.
- [14] Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*, 2017.

- [15] Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. Semeval-2012 task 6: A pilot on semantic textual similarity. In *\* SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, 2012.
- [16] Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. \* sem 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (\* SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43, 2013.
- [17] Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. Semeval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 81–91, 2014.
- [18] Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, et al. 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, 2015.
- [19] Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez Agirre, Rada Mihalcea, German Rigau Claramunt, and Janyce Wiebe. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *SemEval-2016. 10th International Workshop on Semantic Evaluation; 2016 Jun 16-17; San Diego, CA. Stroudsburg (PA): ACL; 2016. p. 497-511. ACL (Association for Computational Linguistics)*, 2016.
- [20] Bill Dolan, Chris Quirk, and Chris Brockett. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th international conference on Computational Linguistics*, page 350. Association for Computational Linguistics, 2004.
- [21] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.
- [22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [23] Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. *arXiv preprint arXiv:1805.01070*, 2018.
- [24] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.
- [25] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.
- [26] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.
- [27] Xun Wang, Haozhi Zhang, Weilin Huang, and Matthe R. Scott. Cross-batch memory for embedding learning. *arXiv preprint arXiv:1912.06798*, 2020.
- [28] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018.
- [29] Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*, 2019.