SimCSE: Simple Contrastive Learning of Sentence Embeddings

Biraj Pandey, Aleksei Sholokhov
Department of Applied Mathematics, University of Washington
`{bpandey, aksh}@uw.edu`

Reproducibility Summary

Scope of Reproducibility

The paper proposes a sentence embedding framework based on contrastive learning that achieves state-of-the-art performance in several benchmark semantic text similarity (STS) tasks in the unsupervised setting. The authors also use quantitative metrics like uniformity and alignment of contrastive embeddings to explain why their approach outerperforms competing methods.

Methodology

The authors provided scripts to download the data required for their unsupervised model. For performance based metrics of their method, we used the authors' code. For quantitative analysis of embeddings and comparison against existing baselines, we implemented parts of their pipeline based on their description. Altogether, it took us around 2 weeks and ≈ 150 GPU hours to reproduce the results. We used four Tesla A40 GPUs which took around 2 hours to train per experiment and around 5 minutes to evaluate for each experiment.

Results

We reproduced the performance-based metrics for the proposed method between $5 - 10\%$ of the reported values; the reproduced models were systematically worse and the difference in performance against baselines were less prominent. Even when taking these differences into accounts, the authors' claim that SimCSE outperforms existing method on STS tasks holds true. We failed to implement a key main result in the paper which argues that the authors' same sentence objective outperforms the competing next sentence objective. We could not implement the competing objective because of its broken code base and lack of implementation details. Finally, we failed to reproduce the results for alignment/uniformity analysis as reported in the paper. In fact, we reached opposite conclusions and argue that this analysis approach fails to explain why their method outperforms other sentence embedding methods.

What was Easy

The authors provided a clean code base with good documentation that was easy to re-implement. They provided replication scripts that delivers the performance-based results for their method. They also showed examples for both single GPU and multi-GPU setups, which was significant for reducing training time.

What was Difficult

The authors did not provide code for data augmentation techniques against which they compared their method. We had to implement several data augmentation methods from scratch relying on vague descriptions. They replicated results for competing baselines but did not publish their implementation code. Due to these issues, we struggled to reproduce a main result because the competing work had a broken code-base and lacked key implementation details. Moreover, we relied on their reported values for the performance of competing methods. Finally, the authors only provided the code to reproduce the performance-based result for their method; they omitted the code for other quantitative analyses.

Communication with Original Authors

We reached out to the original authors on `github` about the specifics of their implementation for word cropping data augmentation technique. They graciously answered our question.

1 Introduction

Natural language embeddings are an important first step for solving a large set of the NLP problems today. Current NLP models utilize embeddings from large pre-trained language models as initial representations and fine-tune their parameters. A popular method of fine-tuning is contrastive learning where embeddings with similar semantic meaning are pushed together while unrelated ones are pushed apart, and the paper by Gao et al. (2021) falls in this subset. It proposes a simple contrastive framework where contrastive training pairs are generated by using using different dropouts masks of the same pre-trained language model, in contrast to constructing the pairs manually by data augmentation. The authors show that embeddings obtained by fine-tuning the pre-trained model in this way improves the uniformity of the embedding i.e. random embeddings are scattered uniformly while maintaining the alignment i.e. semantically similar sentences are close in embedding space. Additionally, they show that the embeddings achieve state-of-the-art performance in several benchmark semantic language similarity tasks.

2 Scope of Reproducibility

This paper introduces a new method to create samples for training sentence-embeddings using contrastive loss. Borrowing an idea from data augmentation, the authors apply two different dropout masks to the same sentence in order to create a pair of embeddings. These two embeddings are used as positive pairs in a contrastive learning framework to fine tune the pre-trained model's weights. The authors then evaluate different methods on how uniformly embeddings are distributed, how closely matched sentence pairs are aligned (as measured by cosine similarity), and how well the model performs on sentence similarity tasks. We were interested in verifying to what extent their particular method of creating positive pairs (dropout noise) is important versus simply the contrastive loss framework in finding embeddings that are correlated with the STS-B benchmarks. To that end, we decided to focus on the unsupervised SimCSE framework and reproduce the claims in the paper outline below.

2.1 Addressed Claims from the Original Paper

1. Using different dropout masks to create contrastive pairs will improve Spearman correlation of the embeddings on the STS-B development dataset compared to other techniques like word cropping, word deletion, synonym replacement, and masked language model.
2. Using dropout with contrastive objective will improve Spearman correlation of the embeddings on STS-B development dataset compared to other unsupervised objectives like Next sentence Logeswaran and Lee (2018), Next 3 sentence and Delete One word. This is true whether one encoder or two encoders are used for obtaining contrastive pair embeddings.
3. In the SimCSE framework, embeddings obtained with $\sim 10\%$ dropout probability on the BERT Devlin et al. (2018) model will lead to the best performance on the STS-B development dataset compared to other dropout probabilities.
4. SimCSE embeddings using $\sim 10\%$ dropout rate show the best uniformity among positive pairs examples compared to embeddings from deleting one word, fixed dropout mask with $\sim 10\%$ dropout, and no dropout. In terms of alignment, it is competitive compared to delete one word approach.
5. SimCSE embeddings from BERT Devlin et al. (2018) and RoBERTa Liu et al. (2019) models show higher Spearman correlation on STS12 — STS16 and SICK-R benchmarks compared to other embeddings methods like GloVe, BERT with different post-processing methods, and other contrastive approaches like InferSent-BERT (IS-BERT) Zhang et al. (2020) and Contrastive Tension (CT) Carlsson et al. (2021).

3 Methodology

3.1 Model Descriptions

Training objective: The paper uses contrastive learning loss function introduced by Hadsell et al. (2006); the authors follow the contrastive framework in Chen et al. (2020) and take a cross-entropy objective with in-batch negatives Chen et al. (2017). Namely, they assume a set of paired examples $\mathcal{D} = \{(x_i, x_i^+)\}_{i=1}^m$ where x_i and x_i^+ are semantically related. Let h_i and h_i^+ denote the representations of x_i and x_i^+ respectively. Then the training objective for (x_i, x_i^+) with a mini-batch of N pairs is:

Model	# Parameters	Source
BERT$_{\text{BASE}}$	110M	Devlin et al. (2018)
BERT$_{\text{LARGE}}$	340M	Devlin et al. (2018)
RoBERTa$_{\text{BASE}}$	125M	Liu et al. (2019)
RoBERTa$_{\text{LARGE}}$	355M	Liu et al. (2019)

Table 1: Number of parameters for each model used in this replication study

Dataset	# train sent. pairs	# test sent. pairs	# dev sent. pairs	Source
STS 2012	N/A	750	N/A	Agirre et al. (2012)
STS 2013	N/A	750	N/A	Agirre et al. (2013)
STS 2014	N/A	750	N/A	Agirre et al. (2014)
STS 2015	N/A	750	N/A	Agirre et al. (2015)
STS 2016	N/A	862	N/A	Agirre et al. (2016)
STS-B	N/A	937	1,500	Cer et al. (2017)
SICK-R	N/A	5000	N/A	Marelli et al. (2014)

Table 2: The following datasets were used for evaluation and hyperparameter tuning in our replication study. All datasets relate to measuring the performance of a language model for Semantic Textual Similarity (STS) task, which involves measuring the semantic similarity between sentence pairs. Only the development set of STS-B is used for hyperparameter tuning. We mark unused parts of each dataset as 'N/A'.

$$\ell_i = -\log \frac{e^{\text{sim}(h_i, h_i^+)/\tau}}{\sum_{j=1}^{N} e^{\text{sim}(h_i, h_j^+)/\tau}} \tag{1}$$

where τ is a temperature hyper-parameter and $\text{sim}(h_1, h_2)$ is a cosine similarity $\frac{h_1^T h_2}{\|h_1\|\|h_2\|}$.

Positive Instances To construct the pairs (x_i, x_i^+) the authors set $x_i = x_i^+$ and construct the pair (h_i, h_i^+) by applying a standard pre-trained language model, such as BERT (Devlin et al. (2018)) or RoBERTa (Liu et al. (2019)), and applying *two different dropout masks* during a forward pass:

$$h_1 = f_\theta(x_1) \tag{2}$$

$$h_1^+ = f_{\theta^+}(x_1) \tag{3}$$

where θ and θ^+ represent the same weights with different dropout masks applied to them. A new dropout mask is sampled randomly for each forward pass.

Model parameters The training fine-tunes the weights of pre-trained model of choice using the contrastive objective from Eq. 1. The models used in this work are standard transformer models namely BERT$_{\text{BASE}}$, BERT$_{\text{LARGE}}$, RoBERTa$_{\text{BASE}}$, and RoBERTa$_{\text{LARGE}}$. BERT$_{\text{BASE}}$ has 12 transformer layers each with hidden size of 768 and 12 attention heads. BERT$_{\text{LARGE}}$ has 24 transformer layers each with hidden size of 1024 and 16 attention heads. Both BERT models are trained using masked language prediction and next sentence prediction objectives simultaneously Devlin et al. (2018). RoBERTa model has a similar architecture as BERT but uses a Byte Pair Encoding (BPE) tokenizing scheme, trains only on the masked language prediction objective and uses larger batch sizes and learning rates Liu et al. (2019). The number of the trainable parameters for each model are given in Table 1.

3.2 Dataset description

Training The authors fine tune pre-trained BERT models using their proposed contrastive framework (Eq. 1 – 3) on 10^6 randomly-sampled sentences from English Wikipedia [1]. The data does not have any labels and contains 19M tokens. The model is fine-tuned on the entire dataset to minimize the contrastive objective.

[1] `https://huggingface.co/datasets/princeton-nlp/datasets-for-simcse`

Category	Original paper	Reproducibility study
Type of hardware	? # of NVIDIA 3090 GPU	4 Tesla A40 GPUs
Memory/GPU	24GB	48 GB
Avg run time for each epoch	? hours	$\approx$ 2 hours
Total number of trials/experiment	1	1
GPU hrs used	? hours	$\approx$ 150 hours
# of training epochs/experiment	1	1

Table 3: We provide the computational details for running the reproducibility study. '?' are details that the authors omitted that we are unable to estimate. Since the authors don't provide the number of GPUs, we cannot estimate their average run time per epoch nor the total number of GPU hours they used.

Evaluation The authors used 7 datasets for evaluating Sematic Textual Similarity (STS); their statistics are summarized in Table 2. The STS12–16 (Agirre et al. (2012)–Agirre et al. (2016)) and STS-B (STS-Benchmark) Cer et al. (2017) datasets contain sentence pairs with ordinal similarity scores ranging from 0 for no meaning overlap to 5 for meaning equivalence. The ground truth similarity scores are given by expert human annotators. The SICK-R (Sentences Involving Compositional Knowledge - Recognizing Textual Entailment) (Marelli et al. (2014)) dataset also contains sentence pairs with real valued semantic relatedness and entailment scores. For SICK-R, only the semantic relatedness scores are used. The performance of a model is measured by the Spearman's correlation of model similarity score with human judgments. The test sets of STS12–16, STS-B, and SICK-R datasets are used for final evaluation. The development set of STS-B dataset are used for model validation and hyperparameter fine-tuning. None of the datasets are used for training. All datasets are open-source and were obtained by running the scripts provided by the authors.

3.3 Hyperparameters

The three main relevant hyperparameters in this work are the dropout probability, learning rate, and batch sizes. Among the three, the dropout probability is key for SimCSE to work properly. We tested several probability values in the range of $(0, 0.5)$. Table 2 contains specific values we tested within that range. We used the same learning rates as that provided by the authors in Appendix A of the paper (Gao et al. (2021)): BERT$_{\text{BASE}}$ ($lr = 3e - 5$), BERT$_{\text{LARGE}}$($lr = 1e - 5$), RoBERTa$_{\text{BASE}}$($1e - 5$), and RoBERTa$_{\text{LARGE}}$($3e - 5$). We used a mini-batch size of 64 for all models.

3.4 Code and Implementation

We use the author's code provided on their GitHub page[2] to conduct the study. The authors provide sufficient resources for replication of their framework. Our reproducibility study is available at `https://github.com/aksholokhov/SimCSE`. We link all the scripts required for dependencies, data download instruction, data pre-processing, training code, evaluation code, and pre-trained models here and in the git repository.

3.5 Computational Requirements

We present the computational details for the original paper and our reproducibility study in Table 3. The authors do not state the number of GPUs they use for training SimCSE. Hence, we are unable to estimate the total number of GPU hours and average run time for each epoch. We can guess that they used more compute than four Tesla A40 GPUs; we ran into out-of-memory issue when training the RoBERTa model with the authors' reported batchsize of 512. This issue persisted for both RoBERTa models. To alleviate this issue, we reduced the batchsize to 64.

4 Results

Our reproducibility study was focused on verifying the authors' claim that SimCSE produces superior sentence embeddings compared to existing sentence embedding methods in the unsupervised regime. To verify their claims, the authors compare SimCSE against existing methods on several Semantic Textural Similarity (STS) tasks. They also use semantic uniformity and alignment measures to compare the quality of their sentence embeddings against that of competing methods. We could ***mostly*** reproduce the performance-based results for SimCSE and even some of the competing methods on STS tasks within a reasonable range of the reported values. The authors claims in regards to superior empirical performance holds true. However, we arrived at different conclusions for the uniformity and alignment analysis.

[2]`https://github.com/princeton-nlp/SimCSE`

Data Augmentation	STS-B (Δ)	SICK-R
None (unsup. SimCSE)	**81.61** (↓ 0.89)	**74.15**
Crop 10%	64.60 (↓ 13.2)	54.24
Crop 20%	59.39 (↓ 12.01)	60.95
Crop 30%	56.97 (↓ 6.63)	61.37
Word Deletion 10%	78.18 (↑ 2.88)	73.21
Word Deletion 20%	79.81 (↑ 7.61)	72.98
Word Deletion 30%	78.31 (↑ 10.11)	71.01
Delete one word	77.10 (↑ 1.2)	71.23
Synonym Replacement	70.88 (↓ 6.52)	68.92
MLM 15%	61.30 (↓ 0.9)	61.37

Table 4: Comparison of data augmentation on STS-B development set (Spearman's correlation). These results correspond to Table 1 from the original paper. The Δ indicates difference in raw values from the original study. The original study conducted this analysis only on the STS-B benchmark. We add an additional SICK-R dataset.

4.1 Dropout masks as data augmentation for contrastive embeddings

While keeping the same contrastive objective, the authors compare their approach of generating contrastive pairs to common data augmentation techniques like crop, word deletion and replacement. They compare the embedding quality of the resulting models on the STS-B development set. We show our reproduction of their results in Table 4. We were able to reproduce the results for SimCSE model up to 1% of the reported value. For the competing baselines, some performed better while others performed worse in our reproduction. We saw improved performance for word deletion by up to 14% points. We saw worse performance for cropping, deleteing one word, synonym replacement, and MLM data augmentation techniques by up to 20%. In particular, the word cropping method performed significantly worse; we elaborate on this in the discussion section. Altogether, our results support the papers' conclusion that SimCSE outperforms all the other data augmentation methods by a wide margin.

4.2 Comparison to other unsupervised training objectives

The authors compare SimCSE, which uses the same sentence to optimize contrastive loss, against QuickThoughts (QT) (Logeswaran and Lee (2018)) which uses next 1 or next 3 sentences to optimize a context-classifying discriminative loss. We were unable to implement the results of QuickThoughts as it lacked a coherent, executable codebase and the paper (in our opinion) lacked several key implementation details. The authors of SimCSE do not provide their implementation of QuickThoughts either. We are unable to support or reject the claims of the author that self-prediction leads to superior embeddings compared to next-sentence prediction.

4.3 Effect of dropout probability on SimCSE embeddings

The authors analyze the relationship between SimCSE's dropout probability and its performance on STS-B development set. We were able to reproduce the performances for most dropout probabilities to within 1% of the reported values (Table 5); we differed significantly in the cases where there is no dropout (diff. by -7%) and where the dropout value is 0.01 (diff. by $+6.8\%$). The authors find the optimal probability to be 0.1 but we find that probabilities between 0.05 to 0.2 all lead to similar performances. Our findings support the paper's conclusion that SimCSE performs significantly worse without any dropout. However, we find that SimCSE is invariant to dropout probabilities within a suitable range.

4.4 Quantitative analysis of SimCSE embeddings

The authors compare the quality of SimCSE's embeddings against other methods like no dropout and delete one word across two metrics, namely *uniformity* and *alignment* based on Wang and Isola (2020). In short, alignment measures the expected distance between embeddings of paired instances while uniformity measures how well the embeddings are uniformly distributed. With every method, the authors find checkpoints of models within every 10 steps of training for the first 100 training iterations. Then, they plot the uniformity and alignment metrics for the resulting checkpoints.

The authors find that SimCSE's uniformity steadily improves while maintaning a constant alignment during the training process. Meanwhile for delete one word, both uniformity and alignment improves over training. Their final conclusion is that SimCSE outperforms delete one word in terms of uniformity and underperforms in terms of alignment at the end of 100 iterations. We were unable to reproduce the authors' results (Fig. 1). Instead, we find that both uniformity and

Dropout Probability	**STS-B** (Δ)	**SICK-R**
No dropout	66.05 ($\downarrow$ 5.05)	62.42
0.01	78.09 ($\uparrow$ 5.49)	64.13
0.05	81.18 ($\uparrow$ 0.08)	72.94
***0.1**	**81.61** ($\downarrow$ 0.89)	**74.15**
0.15	80.85 ($\downarrow$ 0.55)	72.22
***0.2**	**81.93** ($\uparrow$ 1.43)	**75.15**
0.5	69.71 ($\downarrow$ 1.29)	68.00

Table 5: Effects of different dropout probabilities on the STS-B development set (Spearman's correlation, BERT$_{base}$). These results correspond to Table 3 from the original paper. The Δ indicates difference in raw values from the original study. The original study conducted this analysis only on the STS-B benchmark. We add an additional SICK-R dataset.

Model	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
*SimCSE-BERT$_{\text{BASE}}$	65.84	78.96	71.87	80.28	77.34	75.50	71.37	74.45
*SimCSE-BERT$_{\text{LARGE}}$	46.00	65.41	53.54	67.32	69.84	55.50	65.21	60.40
*SimCSE-RoBERTa$_{\text{BASE}}$	67.05	79.54	72.13	79.81	77.94	80.41	70.89	75.40
*SimCSE-RoBERTa$_{\text{LARGE}}$	65.10	79.76	71.97	81.92	77.60	78.54	69.11	74.86

Table 6: Sentence embedding performance on STS tasks (Spearman's correlation over all topics. These results corresponds to Table 5 from the original paper. The authors report results for SimCSE-BERT$_{\text{BASE}}$, SimCSE-RoBERTa$_{\text{BASE}}$, and SimCSE-RoBERTa$_{\text{LARGE}}$. We extend this analysis to SimCSE-BERT$_{\text{LARGE}}$.

alignment dynamics of SimCSE degrades over the first 100 training iterations. Surprisingly, delete one word maintains a steady alignment while improving on uniformity.

4.5 Evaluation on Semantic Textual Similarity (STS) tasks

The authors evaluate several SimCSE models on several benchmark STS tasks like STS12–17, STS-B, and SICK-R. We show the raw values of performances in Table 6 and the differences from the original study in Table 7. For SimCSE-BERT$_{\text{BASE}}$ and SimCSE-RoBERTa$_{\text{BASE}}$, the performances for all benchmarks were within 5% of the reported values. For SimCSE-RoBERTa$_{\text{LARGE}}$, we saw worse performance on the STS-12 dataset by almost 10% (72.86 vs 65.10). For the rest of the datasets, SimCSE-RoBERTa$_{\text{LARGE}}$ performances were within 5% of the original values. All SimCSE models performed generally worse on every dataset during our evaluation. The exceptions were SimCSE-RoBERTa$_{\text{BASE}}$ on STS-B and SICK-R; there was a slight increase in performance from the original study by around 2%. Even when taking these differences into account, SimCSE outperforms competing methods by a significant margin. Our findings corroborate the authors' conclusion that SimCSE produces superior embeddings for semantic textual similarity tasks compared to existing sentence embedding methods.

Note: Our conclusion is based on the performance of competing methods as reported by authors of SimCSE; we do not reproduce results for competing methods.

4.6 Additional Results not Present in the Original Paper

Additional dataset for data augmentation: The authors compare SimCSE against common data augmentations only on STS-B dataset. We extend their analysis to SICK-R dataset Table 4. We find that SimCSE outperforms existing augmentation results in SICK-R dataset as well. Interestingly, word deletion at 10% shows competitive performance against SimCSE (73.21 vs 74.15).

Model	ΔSTS12	ΔSTS13	ΔSTS14	ΔSTS15	ΔSTS16	ΔSTS-B	ΔSICK-R	ΔAvg.
*SimCSE-BERT$_{\text{BASE}}$	($\downarrow$ 2.56)	($\downarrow$ 3.45)	($\downarrow$ 2.51)	($\downarrow$ 0.63)	($\downarrow$ 1.22)	($\downarrow$ 1.35)	($\downarrow$ 0.86)	($\downarrow$ 1.80)
*SimCSE-RoBERTa$_{\text{BASE}}$	($\downarrow$ 3.11)	($\downarrow$ 2.23)	($\downarrow$ 1.12)	($\downarrow$ 1.55)	($\downarrow$ 2.71)	($\uparrow$ 0.19)	($\uparrow$ 2.33)	($\downarrow$ 1.17)
*SimCSE-RoBERTa$_{\text{LARGE}}$	($\downarrow$ 7.76)	($\downarrow$ 4.23)	($\downarrow$ 3.65)	($\downarrow$ 2.85)	($\downarrow$ 4.2)	($\downarrow$ 3.44)	($\downarrow$ 2.15)	($\downarrow$ 4.04)

Table 7: Difference in sentence embedding performance on STS tasks (Spearman's correlation over all topics) from the original paper. These are the same as Table 6 except with the difference calculated over the raw values.

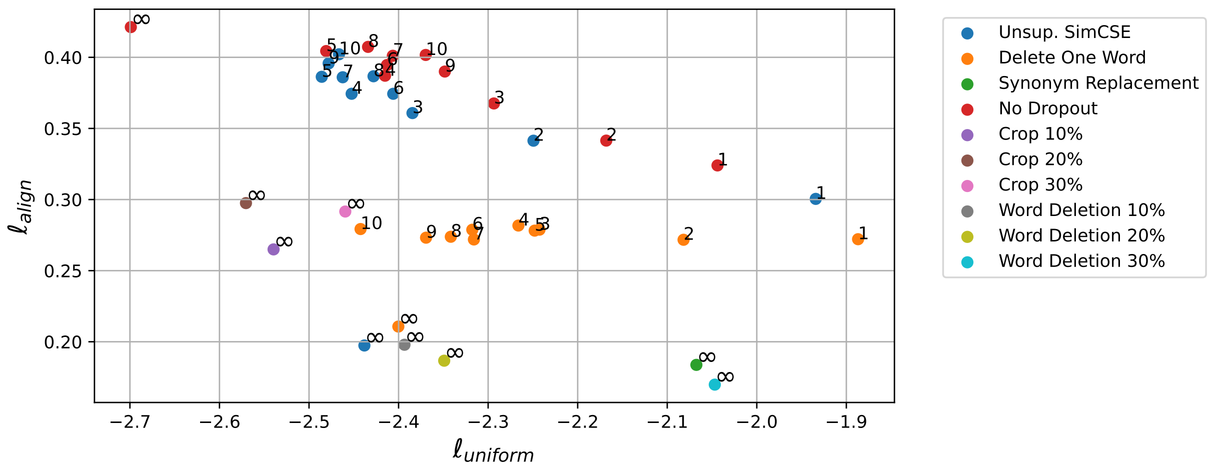

Figure 1: Alignment vs uniformity for sentence word embeddings (based on Wang and Isola (2020)) during the training process. *Lower metrics are better.* Every point with an index n represents a model checkpoint between $10 \times (n-1)^{th}$ to $10 \times n^{th}$ training epochs for the first 100 iterations. The points with an index ∞ represent the model at the end of training. The authors perform this analysis only for unsup. SimCSE, delete one word and no dropout. We extend it to synonym replacement, crop $k\%$ and word deletion $k\%$. The authors stop at the first 100 iterations while we show the metrics at the end of training.

Additional dataset for hyperparameter tuning: The authors tune dropout probability only on STS-B development dataset. We extend their analysis to SICK-R dataset Table 5. We find that dropout probabilities between 0.04 and 0.2 achieve competitive results in both STS-B and SICK-R datasets.

Different architecture for SimCSE on STS evaluation tasks: The authors omit results for SimCSE-BERT$_{\text{LARGE}}$ for STS tasks. We show results for it Table 6. Our results show that on average it performs significantly worse compared to other SimCSE models by up to 20% (60.40 vs 74.45 on compared to SimCSE-BERT$_{\text{BASE}}$.) It is also outperformed by competing baselines by up to 16% on average (60.40 vs 72.05 compared to CT-BERT$_{\text{BASE}}$; CT-BERT$_{\text{BASE}}$ performance taken from Table 5 in the original paper).

Comparison of embedding quality at the end of training and additional augmentation methods: We extend the uniformity and alignment analysis to show results at the end of training in Table 1. The original results only show results for the first 100 training iterations out of 3700 total training iterations. We also show corresponding end of training results for data augmentation methods like synonym replacement, word cropping $(10\%, 20\%, \&30\%)$, and word deletion $(10\%, 20\%, \&30\%)$. SimCSE outperforms delete one word in terms of both alignment and uniformity at the end of training. However, none of the methods outperforms competing methods for both alignment and uniformity. Word deletion at 30% achieves the best uniformity while no dropout achieves the best alignment.

5 Discussion

Our findings indicate that the results of the paper are largely reproducible. For all performance based evaluations, we were within a reasonable range of the reported values. The absolute performance values were systematically worse and the difference from competing baselines were less prominent. Nevertheless, the paper's claim that SimCSE produces higher quality embedding compared to existing methods holds true.

We strengthened results of the paper regarding comparison of data augmentation methods and dropout probabilities by extending it from just STS-B to SICK-R. The authors' conclusions held on this additional dataset as well.

We find a key weakness of the paper to be the difference in training data when compared with competing methods. For competing methods like DeCLUTR and Contrastive Tension, SimCSE authors reevaluate the checkpoints provided by the original authors on the STS benchmarks. The original authors train their methods on different datasets instead of the SimCSE training data (randomly selected Wikipedia 1M sentences). This could result in a significant difference in performance.

We could not implement all of the competing baselines and relied on the authors' reported values. If we could re-implement the existing methods, this reproducibility study would be stronger. Moreover, the supervised SimCSE model in the paper was totally out of scope for our project. We did not have enough time to implement these aspects of the paper.

The authors claim that the superior performance of SimCSE embeddings comes from their superior uniformity while maintaining competitive alignment, when compared to existing methods. To support their claim, they visualize uniformity/alignment for the first 100 iterations out of 3700 total training epochs. Firstly, their results gives a wrong impression that dynamics of these metrics converge to the reported values fairly quickly. We observed different dynamics for the first 100 iterations compared to the reported results. Moreover, when we included other methods, we saw that word deletion at 30% achieves the best uniformity while no dropout achieves the best alignment. There was no model that achieved the best uniformity and alignment. Instead, 6 out of 10 models were on the same Pareto frontier. This suggests that the uniformity and alignment metrics cannot identify the best performing model: different embedding methods simply provide different trade-offs between alignment and uniformity.

5.1 What was Easy:

The authors provided a clean code base that was easy to re-implement. The code was well-documented and the dependencies were nicely laid out. They even provided replication scripts that delivers the performance-based results for their method. They showed examples for both single GPU and multi-GPU setups, which was significant for reducing training time.

5.2 What was Difficult

Data augmentations: The authors did not provide the data or the script for data augmentation techniques. We had to reconstruct them ourselves. In particular, we implemented word cropping ourselves but upon correspondence with the author, we realized that we had implemented it incorrectly. The author did not get back to us in time for us to re-implement those results.

Benchmarking other papers' results: The authors replicated results for some of the competing baselines themselves but did not publish the code. The original sources had code of different degree of quality, mostly impossible to run. In particular, we struggled to re-implement Quickthoughts (Logeswaran and Lee (2018)) and failed to reproduce Table 2 from the paper.

5.3 Recommendations for Reproducibility

We ***STRONGLY*** recommend the original authors and their peers to publish scripts that recover not only the main results but all the other qualitative results and figures present in the paper. Moreover, if they re-implement existing methods they should publish that as well; this would help greatly with the reproducibility crisis that plagues current machine learning research.

Communication with Original Authors

In our study, we found that word cropping $k\%$ data augmentation performed significantly worse compared to the values reported in the paper. We reached out to the original authors on github (`https://github.com/princeton-nlp/SimCSE/issues/231`) about the specifics of their implementation. From their response, we found out that our implementation was different from theirs. However, they replied a day before the original project deadline and we did not have time to re-implement the method.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, et al. 2015. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 252–263.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel M Cer, Mona T Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. Semeval-2014 task 10: Multilingual semantic textual similarity. In *SemEval@ COLING*, pages 81–91.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez Agirre, Rada Mihalcea, German Rigau Claramunt, and Janyce Wiebe. 2016. 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).

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. 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.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. * 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.

Fredrik Carlsson, Amaru Cuba Gyllensten, Evangelia Gogoulou, Erik Ylipää Hellqvist, and Magnus Sahlgren. 2021. Semantic re-tuning with contrastive tension. In *International Conference on Learning Representations*.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Ting Chen, Yizhou Sun, Yue Shi, and Liangjie Hong. 2017. On sampling strategies for neural network-based collaborative filtering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 767–776.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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'06)*, volume 2, pages 1735–1742. IEEE.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Lajanugen Logeswaran and Honglak Lee. 2018. An efficient framework for learning sentence representations. *CoRR*, abs/1803.02893.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, Roberto Zamparelli, et al. 2014. A sick cure for the evaluation of compositional distributional semantic models. In *Lrec*, pages 216–223. Reykjavik.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR.

Yan Zhang, Ruidan He, Zuozhu Liu, Kwan Hui Lim, and Lidong Bing. 2020. An unsupervised sentence embedding method bymutual information maximization. *CoRR*, abs/2009.12061.

Use bibtex and check its output; manual corrections are often necessary.