Master's Thesis of Jinwoo Park

Data Augmentation, Stabilization and Negative Sample Generation Methods for Improving Sentence Embeddings of SimCSE

SimCSE 문장 임베딩 개선을 위한 데이터 증강, 안정화, 부정적 예제 생성 방법

August 2023

Graduate School of Artificial Intelligence
Seoul National University
Interdisciplinary Program in Artificial Intelligence
Major

Jinwoo Park

Data Augmentation, Stabilization and Negative Sample Generation Methods for Improving Sentence Embeddings of SimCSE

Advisor Wonjong Rhee

Submitting a master's thesis of
Science in Artificial Intelligence

Augst 2023

Graduate School of Artificial Intelligence
Seoul National University
Interdisciplinary Program in Artificial Intelligence
Major
Jinwoo Park

Confirming the master's thesis written by
Jinwoo Park

Augst 2023

Chair	Kyogu Lee	(Seal)
Vice Chair	Wonjong Rhee	(Seal)
Examiner	Bongwon Suh	(Seal)

Abstract

The contrastive sentence embeddings apply contrastive learning framework to the sentence embedding field, which summarize natural language into fixed-size vectors. The most influential study in this field is SimCSE. It achieved significant performance improvements by simply using dropout and has subsequently inspired numerous follow-up studies.

In this thesis, we summarize the key existing studies in the field of sentence embedding leading up to SimCSE and investigate several methods to enhance SimCSE from three main perspectives. We experiment with different data augmentation techniques to generate positive pairs, apply stabilization techniques to reduce variability of SimCSE, and aim to increase performance by generating additional negative samples in addition to in-batch negatives.

As the result, we found that Gaussian noise injection to the input embedding, weight perturbation alone, FGSM(fast gradient sign method) combined weight perturbation, and Gaussian noise sampling as additional negative samples, can improve the performance of SimCSE. Data augmentations on encoder output turned out to be not helpful in our experiments and the gain of augmentation on input embedding and stabilization techniques also turned out to be not so significant. Gaussian noise sampling for negative sample generation, however, demonstrated that simple noise can enhance performance comparable to more actual sentences without increasing computational complexity.

Keywords: Sentence embedding, Constrastive learning, SimCSE
Student Number : 2021-25413

요약(국문 초록)

대조적 문장 임베딩(Contrastive sentence embeddings)은 자연어를 고정된 크기의 벡터로 요약하는 문장 임베딩 분야에 자기 지도학습(Self-Supervised Learning)의 대표적 학습방법인 대조적 학습(Contrastive Learning)을 적용한 것이다. 이 분야에서 가장 영향력 있는 연구로 SimCSE를 들 수 있다. 해당 연구는 단순한 dropout 기법만을 사용하고도 상당한 성능 개선을 이루어 냈기 때문에 수많은 후속 연구들을 이끌었다.

이 연구를 통해, 우리는 이 분야에서 SimCSE에 이르기까지 주요한 기존 연구들을 개괄하고 3가지 관점에서 SimCSE의 성능을 개선하기 위한 여러 실험들을 진행하였다. positive pair를 만들기 위해 여러 데이터 증강(data augmentation) 기법들을 적용했고, 변동성을 감소시키기 위해 안정화 기법들을 도입했으며, 배치 내의 다른 문장들을 negative sample로 사용하는 것 외에도 추가적인 negative sample을 생성하여 함께 사용하였다.

결과적으로, 우리는 입력 임베딩 단에 가우시안 노이즈를 주입하는 데이터 증강 방법, FGSM(fast gradient sign method)와 Weight perturbation기법을 조합한 안정화 방법, 그리고 다른 배치 내 문장들과 유사한 스케일의 가우시안 노이즈를 추가적인 negative sample로 사용했을 때 성능 향상 효과가 있음을 발견하였다. 인코더 출력에 대한 데이터 증강은 그다지 도움이 되지 않는 것으로 나타났으며, 입력 임베딩 증강이나 안정화 기법들은 대체로 영향이 크지 않았다. 반면 추가적인 negative sample 생성에서는 연산 복잡도 증가 없이 단순한 노이즈만으로도 실제 자연어 문장을 추가로 negative sample로 사용할 때만큼 성능 개선 효과가 큰 것을 보였다.

주요어: 문장 임베딩, 대조적 학습, SimCSE

학번: 2021-25412

Table of Contents

List of Tables

List of Figures

1 Introduction

Sentence embedding is a fundamental task which organize natural language sentence as variable-length sequences composed of discrete tokens into fixed-size vectors. Various aspects of the natural language sentences, such as lexical, syntactic structure, and semantics, are summarized to representation in the embedding space that can be handled by neural networks. Since most neural networks utilize fixed-size inputs and outputs, sentence embedding is the first step of natural language processing with neural networks. Even in architectures that handle variable-length inputs, it can be divided into two parts: the encoder for sentence embedding and a simple multi-layer perceptron or linear layer part for particular task.

However, despite this significance, universal sentence embeddings remains extremely challenging study. The vocabulary varies greatly depending on the corpus and the domain. Sentence embeddings trained on a specific domain perform poorly in other domains. Moreover, annotating natural language sentences is also a difficult task. While anyone can read natural language sentences, there is no consistent and universally agreed-upon criterion for labeling. Consequently, many datasets are relatively small in size and very noisy.

As a result, there have been numerous attempts to learn universal sentence embeddings through unsupervised learning without the need for labels. BERT[29] significantly improved encoding performance solely through mask reconstruction pretraining on large corpora. However, using only next sentence prediction pretraining for sentence embedding did not outperform the performance of traditional word embeddings and mostly summarized sentences into similar vectors, which shown in Figure 1. Because entire pretraining process is too laborious task, many efforts

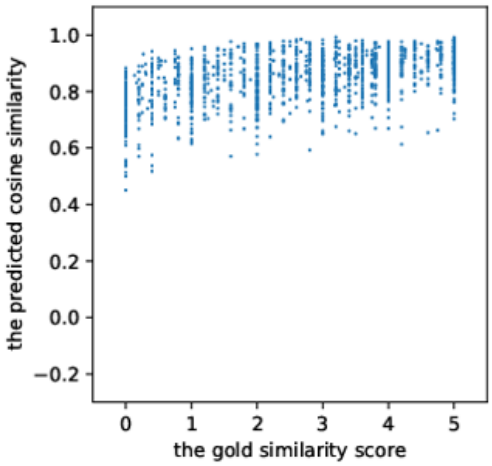

Figure 1: Cosine Similarity of BERT embeddings compared to golden score reported at [53]

have focused on extending BERT[29] with additional pretraining specifically to enhance sentence embedding performance.

Particularly, inspired by the success of the contrastive learning framework in the computer vision domain, similar approaches have been attempted to improve BERT[29]. However, unlike images where the entire content remains invariant with slight continuous changes, natural language is composed of segmented tokens. Data augmentation directly modifying tokens are less effective due to overly significant changes that can be recognized by human. In contrast, continuous data augmentation at the embedding level yielded better results. Notably, SimCSE[18] which employing the default dropout from BERT[29] as data augmentation, demonstrated a substantial performance improvement. Figure 2 from [18] show the simplicity of implementation. Also its provision of a consistent criterion and frameworks led to many subsequent studies.

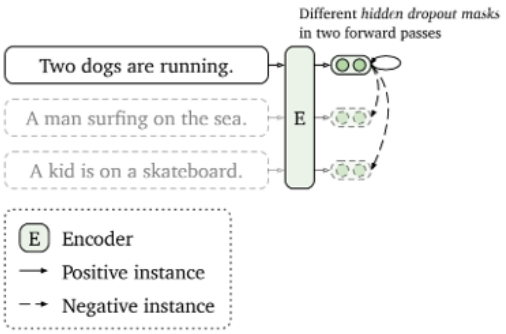

Figure 2: Unsupervised SimCSE[18]

In this study, we provided an overview of various research efforts in the field of sentence embeddings. We summarized key methodologies that have had a significant impact on the sentence embedding research, ranging from early word embedding studies to BERT[29] and SimCSE[18]. Furthermore, we presented a report on a comparison of diverse experimental ideas to explore potential improvements in SimCSE[18]. Our primary focus was divided into three aspects: data augmentation methods to create positive pairs, techniques to generate better negative samples, and approaches to reduce randomness in training.

Throughout this investigation, we discovered the following observations. Data augmentation to the output representation hindered the learning of sentence embeddings, while augmentation in the input embedding had limited improvement effects. Additionally, stability techniques had minimal impact due to SimCSE[18]'s training dynamics. However, it was found that additional negative samples using Gaussian noise with similar scales led to performance improvement and a reduction in variability.

2 Background

2.1 Sentence Embedding

Sentence embedding is fixed size vector, which summarized variable length natural language sentence. It can be used for various NLP tasks like sentence classification or information retrieval, as input of machine learning algorithm. Recently, in the view of transfer learning, sentence embedding is considered as good initialization point for fine-tuning.

For theses purpose, semantics, grammatical structure, and lexical information should be comprehensively encoded into certain size of vector. Recurrent Neural Network (RNN) and Self Attention Network (SAN) are generally used as encoder due to their ability of processing variable length input. Unsupervised learning framework, which can update model without label, are used to make universal embedding, but there are several studies for supervised sentence embedding using labels from more general downstream tasks.

Sentences are not only time-series data that can vary in length and contain order information, but also have hierarchical structure and made of discrete character. Extracting information from this complicate data for well-organized representation is very interesting work. But there are several difficulties in collecting sentence data. Annotation cost for sentence is usually higher than image. Also, it is difficult to obtain qualified data because all annotators have own criteria to meaning of sentence. A wide variety of labeling is possible according to task definition, but there is no concrete consensus for universal downstream task that can evaluate quality of learning sentence embedding.

2.1.1 Word2vec

In addition to traditional statistical methods such as TF-IDF, learning-based methods have been widely used. These methods learn embeddings, which are fixed-sized vectors, for predefined tokens in a given dictionary. Word2vec[35] presents two approaches: Continuous Bag of Words (CBOW) and skip-gram. CBOW learns the embedding of a token by averaging the embeddings of the tokens appearing within a certain window around it. On the other hand, skip-gram predicts the tokens appearing within a fixed window around a given token. These methods are based on co-occurrence, assuming that words with similar meanings will frequently appear in close proximity in sentences and should be represented in similar positions in the embedding space. Using this characteristic, the paper demonstrates the validity of analogy tasks like "king-man+woman=queen" proposed in [36]. This implies that the learned embedding space can sufficiently represent semantics.

However, this method has the drawback of indirectly modeling co-occurrences outside the window to optimize computational efficiency. Furthermore, the embeddings generated by this method only represent the meaning of the word itself, so to summarize a sentence, a method of averaging the embeddings of the tokens that make up the sentence must be used. This approach results in static embeddings that do not reflect the context well and tend to be biased towards dominant words.

2.1.2 Glove

The success of word2vec demonstrates its ability to capture semantics effectively. However, it still has the drawback that co-occurrences outside the window need to be learned through other words that appeared together. Among the methods that reflect global co-occurrence, classical Latent Semantic Analysis (LSA) creates

embeddings by reducing the dimensionality of a frequency matrix where words appear in specific documents using Singular Value Decomposition (SVD). Although LSA can reflect overall statistics, it is considered less capable of learning the vector space well for analogy tasks, which is a strength of word2vec.

GloVe[40], on the other hand, aims to combine the advantages of LSA and word2vec by training embeddings in such a way that the dot product between embeddings represents the probability of two words co-occurring in the same document. However, this method still produces word-level embeddings, so there is a need to aggregate the embeddings of all the words that make up a sentence.

2.1.3 Sent2vec

To learn embeddings at the sentence level rather than just at the word level, an extension of word2vec called sent2vec[38] has been proposed. Sent2vec goes beyond the fixed window of CBOW and learns token embeddings for the entire sentence, making the window dynamic.

Furthermore, using not only unigrams but also n-grams has also contributed to performance improvement. Although, like word2vec, the embeddings that make up the sentence are averaged, the authors argue for the importance of the dynamic window. Instead of always averaging with equal weights in a fixed local window, they average the embeddings at the sentence level. Even though fewer weights are allocated, the learning process can assign appropriate weights as it considers the entire sentence.

2.1.4 Infersent

Infersent[14] uses an SNLI(Stanford Natural Language Inference) dataset[6], which consists of pairs of sentences classified into three categories: entailment, neutral, and contradiction, to train sentence embeddings. This dataset is well-curated and has been widely used in subsequent studies that train supervised sentence embeddings.

Originally, this dataset was used for the Recognizing Textual Entailment (RTE) task, which involves classifying the three relationships. However, the authors believed that through these relationships, they could train a general sentence encoder. To accomplish this, they compared various architectures such as RNN-based networks like LSTM, GRU, and BLSTM, as well as self-attention and CNN as encoders. They also compared different combination methods such as subtracting or multiplying the two sentence embedding vectors and concatenation.

2.2 Transformer architecture

2.2.1 Self-attention mechanism

Recently, self-attention is widely used in the field of image as well as natural language. But it is initially invented for encoder-decoder structure that encode source language into fixed-size context vector and decode to target language. In this time, RNN is used to encoder and decoder, which has strong dependency at sequential order. This property led to degradation at long sentence, because model forget the front part of sequence. self-attention is introduced to alleviate this burden. It can assign more weight to more important feature for entire context.

Self-attention also has a lot of variations, but we described dot-product attention

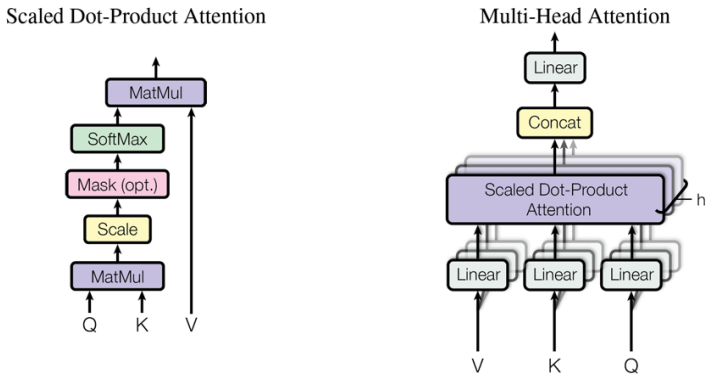

Figure 3: Attention mechanism from [46]

which most widely used. It is similar with hash table that retrieve value by query-key matching. The difference with hashing algorithm is that it return query-key similarity based weight instead of exact match. This probability can be interpreted as how much attention should be applied to certain part of sequence by neural network itself, and this is why it called self attention. Layer input is linearly transformed to yield query, key and value vectors. The similarity of query and key is calculated by dot-product. For this reason, name of this attention algorithm is dot product attention.

Softmax normalize dot-product based similarity so that the sum equals 1, and it represent probability that indicate how important each unit of sequence. Value vectors are summed up according to this attention weight.

In case of multi-head attention (MHA), several dot-product attention modules are used to deal with various feature for problem solving. Each result from modules are concatenated and transferred by linear transform to yield final output of MHA. Figure 3 show diagram for scaled dot-product attention and multi-head attention.

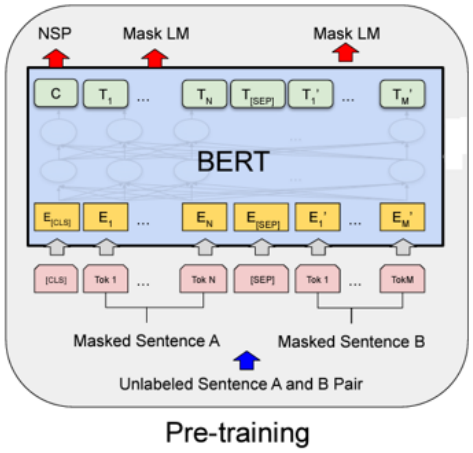

Figure 4: NSP and mask LM task for BERT[29]

2.2.2 Transformer layer

Transformer[46] layer is consist of self attention layer and position-wise feedforward network (FFN) layer. Each sub layer are wrapped by layer normalization and skip-connection. Several transformer layers are stacked to form body of transformer network. Input of entire network is embedding from loop-up table that each element represents token which means unit of split sequence. On the other hands, output of neural network is passed to mult-layer perceptron (MLP) for classification.

This structure can used as encoder that transfer input sentence to latent vector and decoder that interprets latent vector back to natural language. BERT[29] is encoder, and well-known example of decoder is GPT[41].

2.2.3 BERT

BERT[29] pretraining is comprised of masked language model (MLM) task and next sentence prediction (NSP) task, as shown in Figure 4.

In case of MLM tasks, random token of input sentence is replaced with special mask token then classifier reconstruct original token. Model should gather information from context to decide which token is appropriate for masked position, and self-attention mechanism is suitable for MLM. This pretraining task gain great success at NLP. Discrete nature of language is well fit with mask reconstruction, and unsupervised setting allow usage of a lot of unlabeled text.

On the other side, next sentence prediction is focused to relationship between sentences. Two sentences are concatenated with special separation token, and another special CLS token is placed in front of entire sequence. Model should decide whether concatenated sentences are actual adjacent or randomly paired. Binary classification is applied at position of CLS token of model output. It is assumed that model should understand semantics of each sentence, and CLS output is regarded as general sentence embedding.

2.2.4 Transformer based sentence embedding

However, it is not certain whether the NSP (Next Sentence Prediction) task is suitable for creating sentence embeddings, and in the case of models like RoBERTa[33], the task of training the CLS token during pretraining has been completely removed. For the NSP task, two sentences need to be concatenated as input, while sentence embeddings should summarize only one sentence. Additionally, there have been claims that using the CLS output of BERT as-is results in excessively high similarity compared to human evaluations, and it is common for performance to degrade

in downstream tasks when using a pretrained model while keeping it frozen. As a result, for sentence embeddings, fine-tuning is required, which again poses the problem of needing expensive labels.

Sentence-BERT[43] was developed by finetuning a pretrained BERT model as a sentence encoder. Similar to InferSent[14], it uses SNLI datasets[6] and the same architecture. To summarize the BERT output, they compared using CLS and mean pooling, and also compared different objectives such as predicting the cosine similarity of positive and negative pairs directly as a regression objective or using a triplet objective function. Notably, averaging the entire output of the last layer of BERT was more effective than using just the CLS output, and similar to InferSent, they concatenated the summarized vector along with its difference and element-wise multiplication. This indicates that the CLS output of BERT is still not sufficient as a sentence embedding.

The Universal Sentence Encoder[9] used multi-task learning with three tasks to train the Transformer architecture. In addition to the NLI training objective mentioned earlier, they added a task of predicting matching pairs of inputs and responses using a dataset collected for Gmail's Smart Reply service[23]. They also introduced the skip-thought[32] task, which involves decoding embeddings that summarize the current sentence from a corpus consisting of consecutive paragraphs and predicting the preceding and succeeding sentences.

2.3 Contrastive learning

The contrastive learning framework emerged as a crucial technique in recent unsupervised learning, and achieved significant success in the field of computer vision. The fundamental idea is that when augmentation is applied to an anchor, it

should still be perceived as the same object with different view. The agreement between transformed positive samples and anchor should be maximized. After passing through the encoder, ideally two representation should be mapped close location in the latent space. Examples of such invariant transformations can be seen in techniques like random cropping, color jittering and Gaussian blur used in Simclr[10].

$$l_i = -log \frac{exp^{sim(z_i, z_i')/\tau)}}{\sum_{j=1}^{N} exp^{sim(z_i, z_j')/\tau)}} \tag{1}$$

In Equation 1, the numerator encourages positive pairs to be closer, while the denominator pushes negative pairs further apart. In practice, it is not feasible to perform this calculation for every sample in the dataset due to computational limitations. Instead, a random negative sampling approach is used, where only a subset is sampled to be pushed further away. Also it is convenient to consider other items within the batch as randomly extracted negative samples, which are referred to as in-batch negatives.

3 Related work

3.1 Discrete contrastive sentence embedding

Like pioneers in CV, these researches mainly focused to augmentation of original sentence to produce different views. However, discrete nature of language prevents effective augmentation. Image is consists of continuous float value and have high correlation only between adjacent pixels. On the other side, simple token replacement can lead to unnatural incorrect sentences or very different meaning like 'not' of negative statements. For this reason, discrete data augmentation achieved relatively poor result.

3.1.1 Back translation

Back translation is most widely used data augmentation method in NLP. With two different translation model, original sentence is translated from source language to target language, than translated back to source language. It is assumed that semantic maintained same not only between two different languages, but also paraphrase that result of back translation. CERT[15] augment original English sentence to two paraphrase with German and Chinese translation model. Model architecture is same with MoCo[22], which use momentum encoder and negative sample queue.

The performance improvement is not that great because paraphrase should not be mapped into exact same position in embedding space. Although semantic is the most important factor for sentence embedding, there are still other information should be encoded like lexicon. Also, this method highly depends on quality

of translation model. However, quality of paraphrase from back translation is not always good, so it may be not safe for automatic data augmentation without cleansing and refinement by human annotator.

3.1.2 Span in paragraph

It can be assumed that sentences in continuous text share the same context. Based on this assumption, DeCLUTR[19] expand single sentence to span. Anchor which indicates original view and positive span are sample from same paragraph. This pair can overlap, have no intersection, or be inclusive, but length of anchor is longer than positive span. This consists positive pair for contrastive framework, and positive span of other anchor become negative pair. BERT output of each span is averaged to yield same size embedding vector.

Although all training data are unlabeled, order preserved corpus is hard to gather then data consisting of randomly collected sentences without such constraints. Also, the context of the entire paragraph is not necessarily need to encode sentence.

3.1.3 Word replacement

BERT is not robust to certain adversarial samples. For example, single adverb like 'lastly' or 'satisfying' can reverse meaning of original sentence. CLINE[47] introduce semantic negative example for contrastive learning. Rule-based algorithm replace representative word of sentence. If replaced word is synonym, entire augmented sample is for positive pair. On contrast, sentence replaced by antonym become negative pair.

This study mainly focused on transfer learning rather than sentence similarity. Also, word replacement highly depends on heuristics.

3.1.4 Combination of discrete data augmentations

Like SimClr[10], ConSERT[53] tried to find maximum performance by combining various types of discrete data augmentation techniques. In addition, these transformations are applied at input embedding instead of natural language sentence as previous studies.

4 augmentations are tested: Token shuffle is reordering vocabulary by mixing position embedding order. Cutoff is removing one dimension at token axis or feature axis, while dropout randomly delete cell of embedding without considering the axis. Finally, adversarial attack adopts fast gradient value for supervised learning scenario. Each technique is complementary. According to this research, combination of token shuffling and feature cutoff performed best.

The difference from SimCSE, which will be described later, is that all augmentations are applied at only input embedding, not intermediate representations. But it is worth that location where the augmentation is applied is starting to deviate from raw input sentence. All models can observe is limited to discrete sequence of embedding. Even a small change to the human eye can make a big difference in the model input. it can be additional burden for the model to learn which change is meaningful.

3.2 Continuous contrastive sentence embedding

3.2.1 Summarization of local feature with 1D CNN

N gram means N subsequent words in sliding window. In classic NLP studies, A set of N gram that increase in window size by 1 used to capture local pattern frequently occurring. IS-BERT[54] utilize 1D convolution neural network (CNN)

as N gram, and CNN filter size represent N. BERT output passed by 1D CNN filter as localized representation and CLS output as global representation constitute positive pair. localized features from other sentences used as negative sample.

When this study was conducted, there are still explanation of contrastive learning based on mutual information. So researchers also mentioned MI in paper, but no clear connection was suggested. However, it is an earlier approach that apply contrastive learning at latent space instead of raw input.

3.2.2 Output of the other layer

BERT is known to handle different aspects of a sentence based on its layers, where the vocabulary is processed in the lower layers, grammar in the middle layers, and semantics in the final layers[27]. As a result, depending on the task, it is sometimes common to concatenate not only the output of the last layer but also the output of the first layer of BERT and use them together.

SG-OPT[30] made use of this and assumed that the intermediate outputs of each layer are also different views. They used two BERT models, one of which used the CLS output to create sentence embeddings. The other encoder was kept frozen, and the outputs of the 12 layers were randomly sampled, pooled to create a fixed-size vector. These two vectors formed the positive pair, while the negative pair used the CLS output or pooled intermediate output from another sentence in the batch.

The drawback of this approach is that although mean pooling is commonly used, it is difficult to argue that it is appropriate to make it exactly the same as the CLS output. Furthermore, while there may be different perspectives of information in the intermediate layers, they are merely partial perspectives, and trying to make

them converge completely to the final output after passing through all layers can be problematic.

3.2.3 Different batch

To address the problem of outputs from BERT being excessively similar for any pair of sentences, it is necessary for the embeddings of different sentences to be sufficiently distant from each other. However, simply pushing them apart can cause the embeddings to lose their semantic coherence and scatter in the embedding space. This is why an anchor point is needed, which serves as a point where the embeddings can become closer while still moving away from each other.

CT-BERT[26] initializes two separate encoders with the same weights initially, but they gradually diverge due to the different batches they receive. One encoder only encodes the sentences designated as positive pairs, while the other encoder takes in all the in-batch negative samples and the positive pair sentences together. For these pairs, the loss is structured to increase the dot product for positive pairs and decrease it for negative pairs. Since the embeddings of positive pairs should be identical, tension is created to not only push them apart but also maintain proximity, enabling them to function as soft anchors for each other.

The major challenge is the need for twice the memory resources due to using two encoders. Furthermore, concentrating the negative samples in only one encoder to forcefully create separation in the model is not efficient.

3.2.4 Dropout based augmentation

Among various data augmentation attempts, SimCSE[18] utilized dropout[44], which was commonly used during pretraining BERT, to create positive pairs.

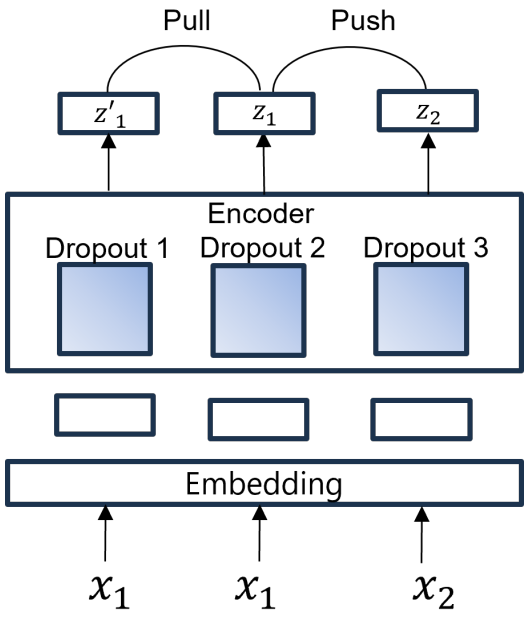

Figure 5: Redrawn concept of SimCSE

Dropout in BERT is applied to input embeddings, the output layer of self-attention, and skip connections. When different dropout masks with a default ratio of 0.1 are applied, the final outputs show cosine similarities of approximately 0.9 to 0.95. In the supervised setting, SimCSE also used NLI task, commonly used in other sentence embedding studies, considering entailment as positive pairs and neutral and contradiction as negative pairs.

With this simple data augmentation, significant performance improvement was observed, and SimCSE became easy to apply, leading to numerous subsequent studies aiming to enhance it. Furthermore, prior to this, the training procedures and evaluation methods varied across studies, but SimCSE provided a framework for both training and evaluation, enabling researchers to compare their work under the same protocol.

SimCSE attributed its success to maintaining alignment with dropout noise while improving uniformity in in-batch negative pairs[49]. However, subsequent research has shown inconsistent results in this regard, and some argue that the performance drop observed when directly using alignment and uniformity as loss cannot be solely explained by this perspective[37].

We redraw concept of the SimCSE in figure 5 again for easy comparison with our proposition.

3.3 Follow-up studies of SimCSE

3.3.1 Wrapping with prompt

In the pretraining phase, the mask token serves a distinct role from other input tokens as it guides the model to focus on reconstructing the original values at its corresponding positions. This concentration of information from the masked positions enables the model, through a simple MLP head, to classify the original words. From this perspective, it can be inferred that sufficient information for summarizing the sentence accumulates at the masked locations.

At PromptBERT[28], the prompt is introduced in the sentence embedding task as a replacement for the mask language model task. When denoting the original sentence as X, prompts such as "[X] means [mask]" are used. To find the optimal prompt, various methods were compared. This included comparing hand-crafted templates in a greedy manner, selecting the sentence with the highest probability when using T5 model[42] to replace the mask[17], and even exploring the approach of learning continuous vectors instead of actual tokens[55], concatenating them with the original sentence's embeddings for prompt usage. Among these

methods, combining hand-crafted prompts and continuous prompts yielded the best performance.

The focus on the mask token instead of the CLS token, considering the criticism of NSP's inadequacy for sentence embedding, is intriguing. However, it should be noted that prompt search requires additional time and separate model training, which can be considered as drawbacks.

3.3.2 Negation as soft negative sample

The anchor sentences were transformed into negated sentences using rule-based discrete transformations. By using the spacy tool[24] to separate and analyze dependency syntax trees, it was possible to reverse the meaning of the sentences without grammatical errors. Additionally, the technique of PromptBERT[28] was adopted, using the representation of the mask token's position instead of the CLS token.

Furthermore, the negated sentences created in this way were used with bidirectional margin loss (BML), unlike regular negative samples. In SNCSE[48], these soft negative samples should be kept at a certain margin of closeness compared to the cosine similarity between the anchor and positive samples. This is done to express sentences that are almost similar to the original text but have significantly different meanings.

Similar to PromptBERT[28], there is a heavy dependence on prompt engineering. Additionally, a drawback of the proposed BML is that tuning is required for the margin between the anchor and soft negative samples.

3.3.3 Replaced token detection

Among various pretraining methods, the most widely known approach is the mask language model (MLM) task, where a portion of the original sentence is replaced with mask tokens and fed into the encoder. The task involves reconstructing the masked values. However, besides this method, other pretraining approaches have been proposed, and this paper draws inspiration from one of them, namely ELECTRA[12].

ELECTRA employs a replacement token detection task instead of mask reconstruction. It involves using a separate BERT model to restore the masked sentence and then performing binary classification to determine whether the tokens at the respective positions are unchanged or replaced. However, there was a problem with ELECTRA when the BERT model used for sentence restoration performed too well. In such cases, it became challenging to detect the fact that the tokens were replaced since the restoration appeared highly convincing.

In DiffCSE[11], when performing replacement detection, the authors introduced the use of SimCSE to provide summarized sentence embeddings as references. Conditioned ELECTRA then detects whether the original sentence has been modified based on the provided sentence embeddings.

The formulation of finding differences between the original and edited sentences is a good idea. However, the addition of a generator and discriminator makes the architecture more complex.

4 Methods

While studying related works, there are three major areas for improvement in SimCSE. We conducted studies to improve performance at each of these perspectives: Data augmentation for positive pair, stability, and additional negative sample generation.

4.1 Data augmentation

Compared to computer vision tasks, SimCSE lacks diverse data augmentation methods for generating positive samples. While various data augmentation techniques are commonly used in image-related tasks, SimCSE relies on the simplest form of dropout.

It was believed that token-level discrete augmentation, which is generally understandable to humans, would bring about too drastic changes and may not be suitable for generating positive samples. Instead, continuous data augmentation was considered to be effective.

4.1.1 Data augmentation at encoder output

Before SimCSE, there have been reports of the effectiveness of adversarial attacks as a form of data augmentation. Adversarial training typically involves gradient descent up to the input level. However, while exploring relevant studies, we drew inspiration from intermediate level attacks[25] that apply adversarial perturbations only up to intermediate layers, as well as feature space attacks[52] that target the latent representations of auto-encoders.

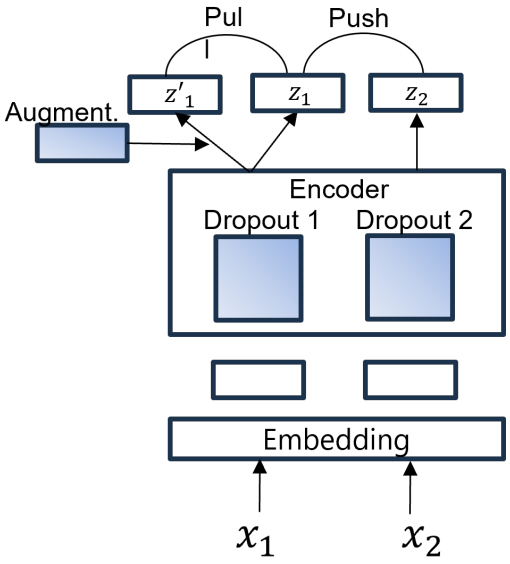

Figure 6: Data augmentation at encoder output

The positive pairs of SimCSE are created by passing the same sentence through the encoder twice, with different dropout masks applied to each pass. However, in this study, the goal was to manipulate the representation obtained by passing a single sentence through the encoder to create embeddings that can be used as positive pairs as illustrated in Figure 6. The goal was performance improvement without increasing the computational load on the encoder.

GNI (Gaussian Noise Injection) For each dimension of the feature representation obtained by passing the sentences within a batch through the encoder, we calculated the standard deviation. We sampled Gaussian noise using a mean of 0 and the standard deviation based on the batch-level standard deviation. This noise was then added to the output of the anchor.

Bernoulli dropout Similar to dropout[44], we also generate a binary mask by

sampling from a Bernoulli distribution. We apply this mask by multiplying it with the output of the anchor when calculating the cosine similarity. This has a similar effect to applying dropout. To obtain cosine similarities in the range of 0.9 to 0.95 for positive pairs, we used a dropout ratio of 0.1.

Mask on trivial features Magnitude-based pruning[21] is widely used technique for neural network compression. Weights with small absolute values are considered to have little impact on overall computations and are therefore excluded by setting them to 0. Inspired by this, instead of using a Bernoulli mask, we set to 0 the dimensions with small absolute values that contribute trivially to computing cosine similarity of vectors. Similarly, we performed this operation only on the bottom 10% of dimensions.

Mask on significant features In contrast to the above approach, top 10% of dimensions to 0 based on absolute values.

FGSM (Fast Gradient Sign Method) [20] is a very simple adversarial attack that computes the sign of the gradient in the direction that maximizes the loss, and then multiplies it by a very small value and adds it to the input. We added FGSM noise on encoder output instead of random noise in GNI.

4.1.2 Data augmentation at encoder input embedding

The proposed methods are shown in Figure 7. Some of these methods have been introduced in various combinations of data augmentation in ConSERT[53]. But it was paper before SimCSE, there has been no comparison of their effectiveness when used in conjunction with dropout.

GNI Adding noise to the input as a data augmentation technique is a widely used

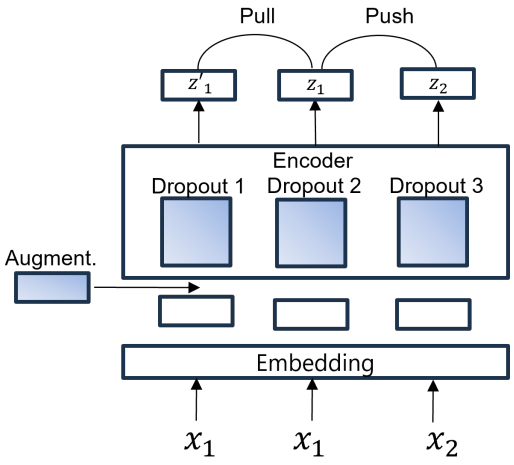

Figure 7: Data augmentation at encoder input embedding

practice in various domains. When obtaining input embeddings from tokenized sentence, Gaussian noise with a mean of 0 and a standard deviation of 0.001 is sampled and add to the embeddings.

Gaussian dropout dropout[44] also proposed Gaussian dropout, which multiplies the input embedding with Gaussian noise. We also introduce Gaussian dropout in addition to Bernoulli dropout. Gaussian noise with a mean of 1 and a standard deviation of 0.001 was sampled and multiplied with the input embedding.

Bernoulli dropout 0.1 ratio Bernoulli mask is applied to input embedding similarly to conventional dropout. This method was proposed in ConSERT[53], but it was only applied at the input level and dropout was not used across all layers as SimCSE.

Bernoulli dropout to feature dimension This approach is similar to cut-off method in ConSERT[53], which drop certain feature dimension. A similar technique can be found in research on speech recognition. SpecAugment[39]) zeroed out random frequency dimensions as a form of data augmentation.

FGSM In ConSERT[53], FGSM method was compared only in the supervised setting, as it requires labels. However, in the unsupervised setting, it can still be applied by computing the gradient in the direction that deteriorates the contrastive loss.

The methods that apply transformations to the input embeddings in addition to SimCSE, show some improvement in performance compared to manipulating the encoder outputs. However, considering the significant variability of SimCSE, it is difficult to claim a significant improvement.

4.2 Stability

During reproducibility experiments, it was discovered that SimCSE itself exhibited significant instability. The performance varied considerably depending on the random seed, to the extent that a substantial portion of the performance improvement observed in subsequent studies could be attributed to randomness. To address this issue, techniques to enhance the model's stability were applied.

During various tests, it was discovered that SimCSE itself can exhibit significantly different performance depending on the random seed. Many seeds yielded lower performance than the reported performance of SimCSE (76.25). When using different machines or libraries, which inherently use different random seeds, it becomes difficult to compare subsequent studies.

To address this issue and stabilize the results, commonly used regularization techniques were introduced to SimCSE. In situations with high variability, these techniques were expected to stabilize the training dynamics and improve the average performance.

Label smoothing One-hot encoding which the target is set to 1 and the rest to 0 can lead to over confident predictions with cross entropy loss. Label smoothing[45] replace hard target with soft target and transform label distribution uniform. It prevents the largest logit from becoming excessively large. In this experiment, a label smoothing parameter is 0.1.

Sharpness-Aware Minimization (SAM) If the loss landscape is sharp, a slight change in the input can result in a significant change in the output. During the training of model parameters, Sharpness-Aware Minimization [16] find parameters that points around the parameters have uniformly low loss. This helps to flatten the entire loss landscape and can contribute to generalization. 0.05 was used for the parameter related to the detection range of sharpness.

Adversarial Weight Perturbation (AWP) From the perspective of adversarial training, a flatter loss landscape for weights also contributes to robust generalization. Adverserial Weight Perturbation[51] flatten the loss landscape by applying worst-case perturbations to the weights.

AWP + GNI AWP is combined with adding Gaussian noise injection at input embedding to transform optimization problem into input-weight perturbation.

AWP + FGSM Instead of simple Gaussian noise, FGSM[20] make worst-case scenarios for both the input and weight aspects. The direction for optimization at the worst case is used to update the original weights.

4.3 Additional negative sample generation

While data augmentation was applied to the anchor to generate positive samples, negative samples were simply obtained from other batch items without any

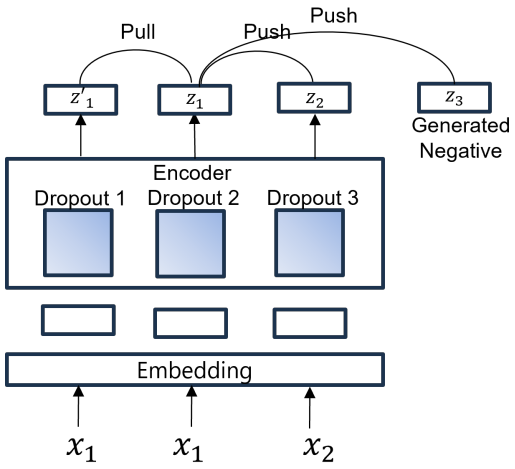

Figure 8: Additional negative sample generation

modification. We attempt to introduce better negative samples.

We mainly explored method to model and sample from the distribution of encoder outputs for the anchor rather than using actual sentences to avoid increasing the computation of the encoder. These generated negative samples are used in conjunction with the in-batch negative pairs, following the same approach as the original SimCSE as visual representation of Figure 8.

Larger batch The batch size used in SimCSE was 64, which is not very large. Generally there is a tendency to use larger batch sizes in contrastive learning. To examine the impact of batch size, the original size 64 is doubled to 128.

Memory bank Large batch size have limitation of computation and memory. The memory bank is commonly used approach to overcome constraint of resources. In sentence embedding task, MoCoSe[7] used memory bank technique with momentum encoder like MoCo[22], but we evaluated memory bank only to show the impact of representation of the past. Encoder outputs from the previous steps are

stored in a queue and used as additional negative pairs in the next step. The number of additional samples from memory bank same as batch size.

GNI to anchor encoder output Similarly to the positive augmentation at the encoder output experiment, we generated additional negative samples by sampling Gaussian noise with a mean of 0 and the standard deviation of feature dimensions across the entire batch. This noise was added to the encoder output of the anchor.

Gaussian noise sampling We calculated the mean and standard deviation from the anchors within the batch and sampled Gaussian noise accordingly. These generated negative samples were then used alongside the anchor samples.

Gaussian noise sampling filtered with covariance matrix The aforementioned method models each feature dimension distribution independently, neglecting the relationships between features. To incorporate this information, covariance matrix is calculated from from the anchors in the batch and perform inner product with sampled noise.

Variational auto-encoder sampling A variational auto-encoder (VAE) [31] approximates the function that samples the latent space by passing the input through an encoder. Sampled latents are passed through a decoder to generate data points similar to the original input distribution. In this approach, we trained a VAE from the stored encoder outputs from previous 8 steps. This model was used to yield additional negative sample similar to the anchor's encoder output.

Gaussian Mixture Model sampling Gaussian Mixture Model (GMM) is a method of modeling distributions by linearly combining Gaussian distributions. Similar to the above method, GMM was updated at each step using the encoder outputs from the previous steps. Then samples from the this GMM are used as additional negative samples.

5 Experiments

5.1 Training procedure

Like other studies, we utilized the framework provided by SimCSE for our experiments.

Model training employed the transformers library distributed by HuggingFace [50], and we utilized 1 million wikipedia data shared by subsequent SimCSE research for training. The model utilized the bert-base-uncased model available in the transformers library. To ensure a fair comparison, we maintained a learning rate of 3e-5 and a batch size of 64, without tuning other hyperparameters. During training, the entire dataset was trained for 1 epoch, and the best model was selected based on the STS-B[8] development set.

Similar to SimCSE, during training, we employed an additional MLP head on top of the CLS output, but during testing, this head was removed, and the CLS output was used directly.

5.2 Evaluation method

	Sentence 1	Sentence 2	Score
1	A girl is styling her hair	A girl is brushing her hair	2.5
2	A group of men play soccer on the beach	A group of boys are playing soccer on the beach	3.6
3	One woman is measuring another woman ’s ankle	A woman measures another woman ’s ankle	5.0
4	A man is cutting up a cucumber	A man is slicing a cucumber	4.2
5	A man is playing a harp	A man is playing a keyboard	1.5
6	A woman is cutting onions	A woman is cutting tofu	1.8
7	A man is riding an electric bicycle	A man is riding a bicycle	3.5

Table 1: STS evaluation example

For evaluation, we utilized the SentEval toolkit[13]. Within this toolkit, it is possible to download data for seven sentence textual similarity (STS) tasks, including STS 2012-2017[4, 5, 2, 1, 3], STS Benchmark[8], and SICK-Relatedness[34]. Examples for STS task were shown in Table 1.

$$cos(\theta) = \frac{A \cdot B}{\|A\| \, \|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}} \tag{2}$$

$$r = \frac{cov(rank(X), rank(Y))}{\sigma_{rank(X)} \sigma_{rank(Y)}} \tag{3}$$

These datasets consist of pairs of sentences that have been human-rated for their similarity on a scale of 0 to 5. Since multiple evaluators provided scores, the ratings are represented as real-valued scores. For these labels, we used cosine similarity of Equation 2 and Spearman's correlation of Equation 3 between the sentence embeddings generated by the model as evaluation metrics. In Equation 3, *cov* denotes for covariance, and *rank* for the rank of score list.

The final performance evaluation criterion is the average performance across the 7 tasks. All results are reported based on the average and standard deviation across five random seeds (19984, 5838, 16822, 19294, 17173). These seeds are also randomly generated to avoid human biases.

6 Result

6.1 Data augmentation

6.1.1 Data augmentation at encoder output

The results with augmentation on output embeddings were not good, as shown in Table 2. The performance of pretrained BERT was very low, at 31.40, so there is some improvement compared to that. However, the baseline SimCSE performs around 75, while most of the results fall within the range of 40 to 50.

Pretrained BERT's sentence embeddings were trained for the next sentence prediction task. Therefore, two sentences should be given to output the scores range from 0 to 1 for meaningful result. Unlike SimCSE's evaluation method, which measures cosine similarity between individual sentences separately, pretrained BERT's encoding ability was not specifically trained for this purpose. Nevertheless cosine similarity quickly increases with just a little training. This indicates that the encoding ability was sufficiently learned during the pretraining process, albeit with a different representation.

This method is expected to make the output embeddings more noisy. Rather than the common Gaussian Noise injection, which maps noisy inputs to the same positions, this approach maps the same inputs to noisy positions. Considering cosine similarity as the decoder, we believed that these noisy representations would directly affect the results, making it difficult to achieve effects similar to intermediate level attack or feature space attack.

As a result, we went back to building upon SimCSE and conducted further research by incorporating additional data augmentation techniques.

method	STS 12	STS 13	STS 14	STS 15	STS 16	STS Benchmark	SICKRelatedness	STS Avg.
Pretrained BERT	21.54	32.11	21.28	37.89	44.24	20.29	42.42	31.40
SimCSE(baseline)	67.26±1.57	80.19±1.17	72.52±1.76	79.65±1.34	77.71±1.06	76.02±1.31	70.26±1.23	74.8±1.12
GNI	40.07±2.77	52.11±2.65	41.38±1.78	56.26±1.60	61.73±3.83	47.64±3.48	52.32±2.27	50.21±2.01
Bernoulli dropout	27.44±4.36	49.14±1.63	35.55±2.65	49.04±2.05	57.26±2.67	35.81±3.55	50.51±1.83	43.54±1.75
Mask trivial feat.	30.7±5.33	51.11±2.91	39.45±3.15	52.37±1.88	61.42±2.01	40.94±4.49	52.19±1.34	46.88±2.65
Mask significant feat.	24.71±3.72	50.20±0.83	36.63±1.59	49.76±0.53	59.52±1.39	35.39±2.46	50.52±1.07	43.82±1.20
FGSM	37.93±11.64	36.12±19.66	30.78±16.45	42.47±16.07	48.81±11.09	42.86±13.7	49.59±9.32	41.22±13.53

Table 2: Data augmentation on encoder output

Method	STS 12	STS 13	STS 14	STS 15	STS 16	STS Benchmark	SICKRelatedness	STS Avg.
SimCSE(baseline)	67.26±1.57	80.19±1.17	72.52±1.76	79.65±1.34	77.71±1.06	76.02±1.31	70.26±1.23	74.80±1.12
GNI	68.14±1.16	81.92±1.10	73.51±1.34	80.26±0.89	77.72±0.37	76.25±0.81	71.22±0.65	75.57±0.70
Gaussian dropout	67.84±1.84	80.25±1.77	72.45±1.43	80.06±1.15	77.93±1.01	76.70±1.35	71.01±1.04	75.17±1.18
Bernoulli dropout	67.24±1.46	79.75±1.77	72.16±1.2	79.09±0.65	76.7±0.95	75.15±1.24	70.26±0.49	74.34±0.81
Bernoulli dropout to feat.	65.66±2.43	78.55±1.25	71.88±1.81	79.44±1.08	76.52±1.39	75.92±0.76	71.92±0.53	74.27±1.17
FGSM	67.32±2.40	79.36±2.25	72.32±1.44	80.13±1.01	77.85±0.43	76.14±0.66	70.13±1.28	74.75±1.21

Table 3: Data augmentation on input embedding

6.1.2 Data augmentation at encoder input embedding

According to Table 3, there are some experiments that finally show improved performance compared to the baseline. When GNI and Gaussian dropout were applied to input embeddings, the performance increased to over 75. However, Bernoulli dropout, on the other hand, slightly decreased the performance, and applying dropout along the feature dimension for more meaningful augmentation did not lead to any significant improvement. Additionally, contrary to the expectation that continuous data augmentation would have a beneficial effect, performance of FGSM is similar to the baseline.

Nevertheless, it is challenging to claim that both cases showed significant performance improvements. This is because SimCSE itself has high variability, making improvement fall within one standard deviation range. Moreover, these methods themselves did not reduce the variation compared to the baseline.

Considering these results, it appears that the input embedding augmentation techniques with SimCSE do not show significant performance improvements.

6.2 Stabilization

Method	STS 12	STS 13	STS 14	STS 15	STS 16	STS Benchmark	SICKRelatedness	STS Avg.
SimCSE(baseline)	67.26±1.57	80.19±1.17	72.52±1.76	79.65±1.34	77.71±1.06	76.02±1.31	70.26±1.23	74.80±1.12
Label smoothing	56.52±1.46	72.18±1.03	62.45±1.22	73.09±1.06	73.72±0.57	67.44±1.53	67.05±0.95	67.49±1.03
SAM	65.45±0.77	79.52±1.65	70.53±1.79	78.51±1.23	76.85±0.93	74.68±1.19	70.43±0.66	73.71±1.02
Weight perturb.	67.87±0.64	80.08±2.09	72.59±0.84	80.53±0.82	78.06±0.83	76.74±0.62	70.90±1.04	75.25±0.74
+ input GNI	67.20±1.42	79.96±2.11	71.84±2.14	79.70±2.14	77.91±1.01	76.20±1.65	70.83±1.17	74.81±1.43
+ input FGSM	68.19±0.73	81.10±0.47	73.83±0.70	80.61±0.74	77.99±0.92	76.81±0.74	71.11±1.19	75.66±0.54

Table 4: Stability

Method	STS 12	STS 13	STS 14	STS 15	STS 16	STS Benchmark	SICKRelatedness	STS Avg.
SimCSE(baseline)	67.26±1.57	80.19±1.17	72.52±1.76	79.65±1.34	77.71±1.06	76.02±1.31	70.26±1.23	74.80±1.12
Larger batch	67.17±1.19	80.66±1.14	73.07±1.13	80.27±0.4	78.8±0.35	76.55±0.86	70.80±0.48	75.33±0.68
Memory bank	69.61±1.91	82.11±1.32	74.58±1.28	81.06±0.74	78.78±0.51	77.88±0.61	71.60±0.65	76.52±0.87
GNI to encoder output	61.00±1.57	72.93±1.00	63.53±0.80	72.58±0.66	72.54±0.48	68.37±0.71	70.58±0.76	68.79±0.47
Gaussian noise sampling	68.43±1.65	82.09±0.83	74.17±0.51	81.76±0.5	79.06±0.38	77.92±0.26	71.21±1.12	76.38±0.52
+ cov.mat. filtering	66.29±0.98	80.41±1.04	72.24±1.27	79.52±1.00	78.04±0.30	76.41±0.84	71.47±0.39	74.91±0.67
VAE sampling	67.01±0.74	79.75±0.96	72.44±0.89	79.57±0.81	78.11±1.18	76.22±0.78	70.41±0.74	74.79±0.69
GMM sampling	68.20±1.49	80.86±0.87	72.48±1.13	80.07±0.88	78.90±0.60	77.88±0.43	71.27±0.92	75.67±0.73

Table 5: Additional negative sample generation

These experiments aimed not only to improve the average performance but also to reduce the standard deviation which indicating variability. Furthermore, most of the methods used in these experiments involved evaluating the update direction by performing a forward pass before real back propagation step. As a result, they exhibit much slower learning speed compared to conventional training methods.

In Table 4, the widely used technique SAM showed a significant performance decline with a score of 73.71 and even did not improve variability. On the other hand, weight perturbation alone and with FGSM achieved a performance of over 75 and reduced the standard deviation to less than 1.

However, the variability is still too high to consider these techniques significant. Especially when considering the losses due to the slow learning speed, the benefits obtained from these methods may be further diminished.

6.3 Additional negative sample generation

The results of additional negative samples are shown in Table 5. The most effective approach was Gaussian noise generated based on the statistics of anchors within the batch, combined with in-batch negatives.

Comparing the representations of the previous and current steps, it can be observed that the cosine similarity between the anchor and each representations are almost same scale. This suggests that the encoder changes very small during training, and the past encoder outputs stored in the memory bank are similar to the current in-batch negatives.

It is noteworthy that simple noise yields a greater improvement in performance compared to larger batches or past encoder outputs, which is the actual sentence embeddings summarizing meaningful sentences.

DCLR[56] proposed a method of applying FGSM[20] to Gaussian noise and using it as an additional negative sample. However, since they did not report the average value, it is difficult to determine if it resulted in a significant improvement. Based on the results of this study, it is presumed that the scale could be a much more critical factor.

7 Discussion

7.1 Data augmentation

Attempts to perform data augmentation on the encoder output generally did not yield good performance. These methods make the representation noisy. When perturbations are applied at intermediate layers, such as in intermediate-level attacks, the subsequent layers can handle noisy representation like this. Different view of the same instances can be mapped to similar positions in the final embedding space. However, there are no layers after the encoder output in SimCSE. Noise directly affects the scores, so cosine similarity as a decoder is not invariant. Therefore, experiments on the encoder output showed lower performance.

On the other hand, methods that applied data augmentation to the input embedding showed slight performance improvements. Gaussian noise injection or Gaussian dropout demonstrated increased performance compared to the baseline. However the magnitude of improvement was relatively small and it is difficult to claim significant performance enhancement considering the high variability of SimCSE. Augmenting only the input embedding has a limited effect because there are many layers following it in the network. However, the dropout used in SimCSE is distributed throughout the entire network, not just at the beginning of layers. Therefore, we can conclude that the input embedding augmentation is a relatively small change and may have limited effectiveness.

Furthermore, the original BERT's dropout is applied after the summation of input embeddings, position embeddings, and token type embeddings. This duplicated dropout may have reduced effectiveness. The case of FGSM could

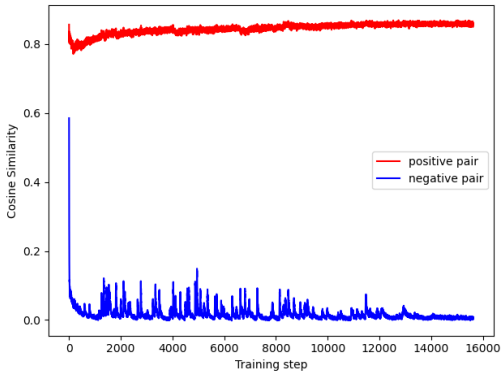

Figure 9: Cosine similarity during SimCSE training

also be attributed to the trade-off between adversarial robustness and original performance in adversarial training.

7.2 Stability

The weight perturbation alone, and the weight perturbation combinated with FGSM show a slight performance improvement. However, this improvement is also small in magnitude, and the variability did not improve significantly as expected. This issue can be attributed to the charicteristic training dynamics of SimCSE.

Figure 9 depicts the cosine similarity of positive pairs and negative pairs during the training of SimCSE. The positive pairs already exceed 0.9 in the early stages of training, while the negative pairs approach 0. In other words, the loss composed of cosine similarity already approaches 0 from the beginning of training. The gradients hardly descend and weight parameters are barely updated. This phenomenon is referred to as gradient dissipation in [37].

The stabilization techniques tested in this study include an additional term in optimization that evaluates the direction of update with preliminary forward path.

Since learning occurs only in very few initial steps, stabilization techniques that continuously influence throughout the training process may not function properly.

Instead, it is possible that data order has a greater impact considering the fluctuations in the similarity of negative pairs observed in Figure 9. The similarity increases if there are coincidentally similar sentences within a batch as in-batch negatives. Since a shared framework is used, it is difficult to separate the effects of data order and augmentation, as the data loader and dropout mask utilize the same random number generator. In future research, it will be possible to fix one side and compare the influence of the other side separately.

7.3 Additional negative sample generation

GMM sampling and Gaussian noise sampling showed performance improvement in additional negative sample generation. GMM sampling had a slower learning speed, but the improvement in performance was not substantial. On the other hand, Gaussian noise sampling was more noteworthy because it exhibited a performance improvement of more than 1 standard deviation of SimCSE. It reached a performance level close to when doubling the batch size to 128 or using a memory bank. Gaussian noise sampling achieved comparable improvements with simple noise injection whereas memory bank utilize encoder outputs of actual meaningful sentences.

When negative pairs are pushed away from the anchor, the direction can become erratic if the batch size is small. Random noise mitigates this scattering effect to yield improved performance. We conducted additional experiments by increasing the number of additional negative samples more than batch size. It can be observed that the performance improves linearly as the sample count increases in Figure 10.

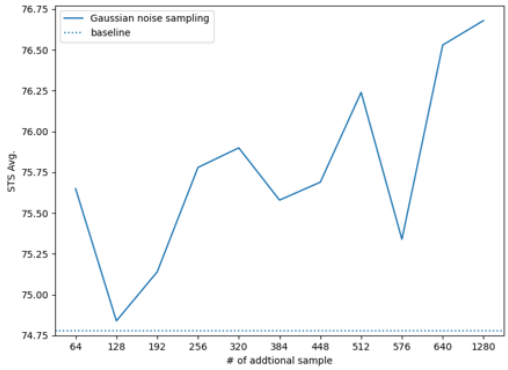

Figure 10: The impact of number of generated negatives

	SimCSE(baseline)			Gaussian noise sampling		
Seed	Alignment	Uniformity	STSB test	Alignment	Uniformity	STSB test
19984	0.48	-2.47	77.04	0.46	-2.35	77.75
5838	0.47	-2.42	76.47	0.49	-2.53	77.74
16822	0.48	-2.44	75.27	0.45	-2.42	77.88
19294	0.45	-2.38	73.85	0.41	-2.11	77.78
17173	0.51	-2.56	77.46	0.47	-2.40	78.43
Mean	0.48	-2.45	76.02	0.46	-2.36	77.92
Std.	0.02	0.07	1.47	0.03	0.16	0.29

Table 6: Alignment and uniformity

Furthermore, SimCSE and subsequent studies explain the performance improvement within the contrastive learning framework by enhanced alignment and uniformity. Alignment indicates how closely positive pairs are positioned to each other, while uniformity reflects how uniformly they are distributed on the unit hypersphere, in relation to negative pairs. We compared the results of five random seeds for the STSB test, and as shown in Table 6, we can observe that in most cases, alignment and uniformity both have improved.

Recently, DCLR [56] proposed converting random noise into additional negative samples through FGSM. However, since reported results is only from a single random seed, it is difficult to make direct comparisons. Also, the performance improvement in [56] is around 1% point, similar to our research. The noise sampled from a mean of 0 and standard deviation of 1 and FGSM enhanced negative sample are compared, and latter achieved better performance improvement in [56].

Method	STS 12	STS 13	STS 14	STS 15	STS 16	STS Benchmark	SICKRelatedness	STS Avg.
SimCSE(baseline)	67.26±1.57	80.19±1.17	72.52±1.76	79.65±1.34	77.71±1.06	76.02±1.31	70.26±1.23	74.80±1.12
N(0,1)	68.85±1.42	80.84±1.03	73.48±0.91	80.51±0.92	78.16±0.65	76.64±0.69	70.47±0.81	75.56±0.74
N(μ_z,1)	67.34±0.87	80.63±1.25	73.12±0.88	79.53±1.35	78.14±0.46	76.30±0.43	70.76±0.87	75.11±0.64
N(0,σ_z)	67.00±0.74	81.08±1.20	72.82±0.71	79.93±0.86	78.01±0.63	76.64±1.28	70.44±0.32	75.13±0.62
N(μ_z,σ_z)	68.43±1.65	82.09±0.83	74.17±0.51	81.76±0.50	79.06±0.38	77.92±0.26	71.21±1.12	76.38±0.52

Table 7: The impact of scale on Gaussian noise sampling

Model	Method	STS Avg.
BERT base	SimCSE(baseline)	74.80±1.12
	Gaussian noise sampling	76.38±0.52
BERT large	SimCSE(baseline)	76.39±0.82
	Gaussian noise sampling	77.02±1.06
RoBERTa base	SimCSE(baseline)	75.82±1.12
	Gaussian noise sampling	75.4±0.34
RoBERTa large	SimCSE(baseline)	76.53±0.78
	Gaussian noise sampling	61.94±28.76

Table 8: Result of the other pretrained model

However, it appears that scale is more crucial. In Table 7, we compared significantly different values, such as 0 and 1, instead of the mean or standard deviation of the encoder outputs within a batch. The performance improvement becomes minimal when scale is differed.

As a follow-up study, hard negative mining can be introduced by incorporating a hardness measure. Hard negative samples, which are similar to the anchor but noticeably different, provides more informative training signal, whereas an easy negative pairs have little impact. Hardness measure can explain the performance improvement effect of generated negative samples. Also, selective hard negative samples from large generated pool are expected to further enhance performance.

7.4 Performance comparison with other pretrained model

Additionally, we verified whether the Sampling from Normal Distribution, which had the most significant performance improvement, had similar effects on other pretrained models with SimCSE. The results are summarized in Table 8.

For BERT large, a similar trend to BERT base was observed. But in case of RoBERTa, proposed method actually hindered performance. In particular, with RoBERTa large, we even observed cases where performance completely deteriorated depending on the seed. However, when comparing SimCSE alone, there was not a significant performance gap between BERT and RoBERTa. This suggests that SimCSE may be a more specialized algorithm for BERT, and it could be challenging to apply the techniques that improve SimCSE's performance to RoBERTa without any modification. Unlike BERT, RoBERTa does not have a specific task focused on the CLS token during the pretraining stage, and this could be a factor that lead to difference in result.

8 Conclusion

Sentence embedding summarizes a sentence of variable length sequence consisting of discrete tokens into fixed size vectors. Pretrained BERT has to be found to struggle in generating effective sentence embeddings, despite the significant impact in the field of natural language processing. Various methods have been proposed to enhance performance of BERT sentence embedding, including studies utilizing unsupervised learning with the contrastive learning framework. Especially, SimCSE shows significant performance improvements with simple dropout augmentation and led many follow up studies. We first provided an overview of the key researches in the field of sentence embeddings in this study.

Next, we conducted several experiments that could be categorized into three areas to enhance the performance of SimCSE. First of all, various data augmentation methods were compared to yield positive samples in addition to dropout. Secondly, stabilization techniques were introduced to reduce excessive variability caused by randomness during training. At last, we generated additional negative samples to aid training instead of solely using other sentences within the batch as negative samples.

The main findings can be summarized as follows: Data augmentation at the output representation was counterproductive, because the cosine similarity used in SimCSE is not invariant to the transformation. Applying data augmentation to the input embedding also had limited improvement due to minor changes compared to dropout in SimCSE. Stability techniques did not significantly improve variability, because their update are mainly constrained in the early stages of SimCSE training. Finally, the additional negative samples showed significant improvement effects.

Among the specific experimental ideas, we confirmed that Gaussian noise injection in input embeddings, a combination of FGSM and weight perturbation, and additional negative samples generated by Gaussian noise sampling with the mean and standard deviation of anchor samples within the batch, can improve the performance of SimCSE.

Particularly, the last method which sampling Gaussian noise showed comparable performance to the widely used memory bank technique. It is more important that simple noise has a performance improvement effect similar to meaningful sentences embeddings without increasing computational complexity. Consistent enhancements in alignment and uniformity were also observed across various random seeds. This can be attributed to the small batch size which tends to scatter the negative samples while modifying pretrained BERT's sentence embeddings with contrastive learning. The noise added as negative sample can alleviate degradation. The fact that generated negatives should be similar in scale with the actual sentence embeddings supports our hypothesis and explains the performance improvement effect.

The limitation of proposed method is that this is specifically designed for application when pretraining BERT into SimCSE. Therefore it may not generalize to improving sentence embeddings of other pretrained models such as RoBERTa. As an future research topic, hardness measure in hard negative sampling could explain how generated samples can be beneficial during training. Also, there can be further performance improvement by generating pool with a large number of additional negative samples and selecting only informative ones.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, 2015.

[7] Rui Cao, Yihao Wang, Yuxin Liang, Ling Gao, Jie Zheng, Jie Ren, and Zheng Wang. Exploring the impact of negative samples of contrastive learning: A case study of sentence embedding. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3138–3152, 2022.

[8] 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.

[9] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

[10] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In

International conference on machine learning, pages 1597–1607. PMLR, 2020.

[11] Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljačić, Shang-Wen Li, Scott Yih, Yoon Kim, and James Glass. Diffcse: Difference-based contrastive learning for sentence embeddings. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4207–4218, 2022.

[12] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*, 2020.

[13] Alexis Conneau and Douwe Kiela. Senteval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

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

[15] Hongchao Fang, Sicheng Wang, Meng Zhou, Jiayuan Ding, and Pengtao Xie. Cert: Contrastive self-supervised learning for language understanding. *arXiv preprint arXiv:2005.12766*, 2020.

[16] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

[17] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, 2021.

[18] Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, 2021.

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

[20] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[21] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015.

[22] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9726–9735. IEEE, 2020.

[23] Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. Efficient natural language response suggestion for smart reply. *arXiv preprint arXiv:1705.00652*, 2017.

[24] Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear, 2017.

[25] Qian Huang, Isay Katsman, Zeqi Gu, Horace He, Serge Belongie, and Ser-Nam Lim. Enhancing adversarial example transferability with an intermediate level attack. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4732–4741. IEEE.

[26] Sverker Janson, Evangelina Gogoulou, Erik Ylipää, Amaru Cuba Gyllensten, and Magnus Sahlgren. Semantic re-tuning with contrastive tension. In *International Conference on Learning Representations, 2021*, 2021.

[27] Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*, 2019.

[28] Ting Jiang, Jian Jiao, Shaohan Huang, Zihan Zhang, Deqing Wang, Fuzhen Zhuang, Furu Wei, Haizhen Huang, Denvy Deng, and Qi Zhang. Prompt-

bert: Improving bert sentence embeddings with prompts. *arXiv preprint arXiv:2201.04337*, 2022.

[29] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.

[30] Taeuk Kim, Kang Min Yoo, and Sang-goo Lee. Self-guided contrastive learning for bert sentence representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2528–2540, 2021.

[31] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[32] Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. *Advances in neural information processing systems*, 28, 2015.

[33] 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.

[34] Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. A sick cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth Interna-*

tional Conference on Language Resources and Evaluation (LREC'14), pages 216–223, 2014.

[35] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[36] Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 746–751, 2013.

[37] Zhijie Nie, Richong Zhang, and Yongyi Mao. On the inadequacy of optimizing alignment and uniformity in contrastive learning of sentence representations. In *The Eleventh International Conference on Learning Representations*, 2023.

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

[39] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*, 2019.

[40] 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.

[41] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training.

[42] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[43] 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 and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, 2019.

[44] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[45] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[47] Dong Wang, Ning Ding, Piji Li, and Haitao Zheng. Cline: Contrastive learning with semantic negative examples for natural language understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2332–2342, 2021.

[48] Hao Wang, Yangguang Li, Zhen Huang, Yong Dou, Lingpeng Kong, and Jing Shao. Sncse: Contrastive learning for unsupervised sentence embedding with soft negative samples. *arXiv preprint arXiv:2201.05979*, 2022.

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

[50] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[51] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. *Advances in Neural Information Processing Systems*, 33:2958–2969, 2020.

[52] Qiuling Xu, Guanhong Tao, Siyuan Cheng, and Xiangyu Zhang. Towards feature space adversarial attack. *arXiv preprint arXiv:2004.12385*, 2020.

[53] Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. Consert: A contrastive framework for self-supervised sentence representation transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5065–5075, 2021.

[54] Yan Zhang, Ruidan He, Zuozhu Liu, Kwan Hui Lim, and Lidong Bing. An unsupervised sentence embedding method by mutual information maximization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1601–1610, 2020.

[55] Zexuan Zhong, Dan Friedman, and Danqi Chen. Factual probing is [mask]: Learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5017–5033, 2021.

[56] Kun Zhou, Beichen Zhang, Wayne Xin Zhao, and Ji-Rong Wen. Debiased contrastive learning of unsupervised sentence representations. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6120–6130, 2022.