SupCL-Seq: Supervised Contrastive Learning for Downstream Optimized Sequence Representations

Hooman Sedghamiz1 Shivam Raval1 Enrico Santus1 Tuka Alhanai2 Mohammad Ghassemi3
1 DSIG - Bayer Pharmaceuticals, New Jersey, USA
2 New York University, Abu Dhabi, UAE
3 Michigan State University, Michigan, USA
hooman.sedghamiz@bayer.com, sbraval@asu.edu, enrico.santus@bayer.com, tuka.alhanai@nyu.edu, ghassem3@msu.edu

Abstract

While contrastive learning is proven to be an effective training strategy in computer vision, Natural Language Processing (NLP) is only recently adopting it as a self-supervised alternative to Masked Language Modeling (MLM) for improving sequence representations. This paper introduces SupCL-Seq, which extends the supervised contrastive learning from computer vision to the optimization of sequence representations in NLP. By altering the dropout mask probability in standard Transformer architectures (e.g., BERTbase), for every representation (anchor), we generate augmented altered views. A supervised contrastive loss is then utilized to maximize the system’s capability of pulling together similar samples (e.g., anchors and their altered views) and pushing apart the samples belonging to the other classes. Despite its simplicity, SupCL-Seq leads to large gains in many sequence classification tasks on the GLUE benchmark compared to a standard BERTbase, including 6% absolute improvement on CoLA, 5.4% on MRPC, 4.7% on RTE and 2.6% on STS-B. We also show consistent gains over self-supervised contrastively learned representations, especially in non-semantic tasks. Finally we show that these gains are not solely due to augmentation, but rather to a downstream optimized sequence representation. Code: https://github.com/hooman650/SupCL-Seq

1 Introduction

Sequence classification is a fundamental problem in natural language processing (NLP), as it has a wide range of applications, including but not limited to the tasks such as sentiment analysis, inference and question answering (Minaee et al., 2020). Cross-entropy loss is generally the default loss function in training neural networks for NLP downstream tasks (Zhang and Sabuncu, 2018; Sukhbaatar et al., 2015). Recently, thanks to the simplicity of augmentation methods in computer vision (e.g., zooming, cropping, rotation, etc.), self-supervised and supervised variants of contrastive learning proved to be effective training approaches in image classification tasks (Wu et al., 2018; Hénaff et al., 2019; Khosla et al., 2020). These methods aim at optimizing the representations by minimizing the distance between similar samples and maximizing it between diverse samples (Chen et al., 2020). Gao et al. (2021) proposed to leverage the built-in dropout masks in attention and fully-connected layers of Transformers (Vaswani et al., 2017) to introduce noise in the embedding representations. This is obtained by simply passing twice the same input and using different dropout masks. In this way, for every representation (anchor), altered views are generated. Gao et al. (2021) applied this augmentation approach to improve the semantic representation of a sequence in a self-supervised fashion, by taking an input sentence and contrasting its similarity against its augmented version and the remaining samples in a batch. The authors further extended this approach by employing positive (i.e., entailment) and negative (i.e., contradiction) examples from natural language inference (NLI) datasets. The resulting sentence embeddings achieved large gains in semantic textual similarity (STS) tasks.

To the best of our knowledge, however, contrastive learning has not yet been applied in a supervised fashion to optimize sequence representations towards downstream tasks.1 Inspired by the recently proposed supervised contrastive learning in computer vision (Khosla et al., 2020), in this paper we introduce SupCL-Seq, which extends the self-supervised contrastive method by Gao et al. (2021) to a supervised contrastive learning approach, in which anchors and altered views, along with their classification labels, are used to learn downstream representations.

1During the review process, we were made aware of a contemporaneous work by Gunel et al. (2020) on supervised contrastive learning for natural language processing. A major difference between their work and ours lays in the adopted augmentation methodology.
The book was written by John.

Books were sent to each other by the students.

She voted for herself.

Positive Class

Negative Class

Different views in two forward passes with different dropout masks

dropout p = 0.0
dropout p = 0.1

Positive Class

Embedding

Negative Class

Embedding

Anchor Positives Negatives

Figure 1: SupCL-Seq applied to COLA (Warstadt et al., 2018). SupCL-Seq first forward propagates the input N times (in this example N = 2) through the same encoder (e.g., BERT\textsubscript{base}) with N different dropout masks (e.g., \( p = 0 \) and \( p = 0.1 \) respectively) and obtains their corresponding noisy embedding views. The noisy embeddings that belong to the same class are then employed as the positive pairs for the original input (anchor with dropout mask of \( p = 0 \)). In this way, the samples belonging to the negative class effectively are used as negatives.

The augmentation step is obtained by forward propagating the input batch \( N \) times in the same encoder with \( N \) distinct dropout masks (i.e., different dropout probabilities). The generated altered views, along with their anchor’s label, are then used to optimize the sequence representations through a supervised contrastive loss function (Khosla et al., 2020). Figure 1 details our training approach.

Formally, our pipeline consists of a single Encoder Transformer, \( Enc(\cdot) \) (i.e., BERT\textsubscript{base} with \( \approx 110M \) parameters (Devlin et al., 2018)). This encoder generates \( N \) altered embeddings, \( \tilde{x}_n = Enc(x, p_n) \), for each input \( x \) and dropout probability \( p_n \). A contrastive loss function is then applied in a supervised fashion to maximize the encoder’s capability of building downstream optimized sequence representations (see Section 2.1). After this contrastive training, the encoder parameters are frozen and a linear classification layer is then trained with cross-entropy. In the remainder of this section, we review the self-supervised contrastive function (Gao et al., 2021) and its extended supervised counterpart inspired by Khosla et al. (2020).

2 Method

SupCL-Seq extends the self-supervised contrastive learning (Gao et al., 2021) for improving semantic representations to a supervised setting, in which representations are optimized towards the downstream task, independently on the number of labels.

2.1 Contrastive Learning

Let \( i \in I \equiv \{ 1 \cdots MN \} \) be the index of all the encoded sequence embeddings \( \tilde{X} \equiv \{ \tilde{x}_1 \cdots \tilde{x}_{MN} \} \) in an input batch. Each sample \( i \) is forward propagated \( N \) time using distinct drop-out masks, gener-

\footnote{We employ the BERT\textsubscript{base}’s last layer’s hidden-state of the first token of the sequence (i.e., pooled CLS embeddings) as \( \tilde{x}_n \), which is then \( L_2 \) normalized.}
We performed a set of experiments to i) evaluate the effect of number and level of dropout passes on two challenging datasets (see 3.1); ii) compare the performance of a standard BERTbase (Devlin et al., 2018) architecture with a SupCL-Seq-empowered BERTbase model on several benchmarks in GLUE (Wang et al., 2018) (see 3.2); iii) compare the performance of SupCL-Seq with the self-supervised contrastive approach introduced by Gao et al. (2021) in a subset of tasks (see 3.3); and, finally, iv) assess whether the improvements achieved with our approach are solely due to augmentation (i.e., dropout masks) and to which extend contrastive loss helped (see 3.2.1).

### 3.1 Dropout Levels

In order to study the effect of the number and the level of dropout passes, we assessed the performance of several configurations of BERTbase on CoLA (Warstadt et al., 2018) and RTE (Dagan et al., 2006) datasets. Gao et al. (2021) empirically showed that using two distinct dropout masks with the same probability of $p = 0.1$ lead to the highest performance in their settings. In our supervised experiments, instead, we can generate views with different levels of noise, as the system can always rely on their labels. Therefore we choose different parameters, using intervals of 0.1 for the dropout probabilities. Table 1 reports the results for both datasets. While clear improvements are visible on CoLA when more masks are applied, experiments on RTE show that this is not always the case. In the latter dataset, in fact, performance fluctuates largely across the settings, achieving the highest score when three passes are used. This suggests that the number and level of dropout passes is a task-dependent hyper-parameter.

### 3.2 GLUE Tasks

In order to assess the benefit of SupCL-Seq, we compared the performance of a standard base BERT on several GLUE benchmarks. Table 1 reports the results for both CoLA and RTE tasks. Score denotes Matthews Correlation Coefficient.

<table>
<thead>
<tr>
<th>Task</th>
<th>Drop-out</th>
<th>Batch size</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoLA</td>
<td>$[0.0,0.1,0.2,0.3,0.4]$</td>
<td>800</td>
<td>61.2</td>
</tr>
<tr>
<td>CoLA</td>
<td>$[0.0,0.1,0.2,0.3]$</td>
<td>640</td>
<td>57.9</td>
</tr>
<tr>
<td>CoLA</td>
<td>$[0.0,0.1]$</td>
<td>480</td>
<td>58.9</td>
</tr>
<tr>
<td>CoLA</td>
<td>$[0.0,0.1,0.2,0.3,0.4]$</td>
<td>640</td>
<td>62.4</td>
</tr>
<tr>
<td>RTE</td>
<td>$[0.0,0.1,0.2,0.3,0.4]$</td>
<td>800</td>
<td>63.5</td>
</tr>
<tr>
<td>RTE</td>
<td>$[0.0,0.1]$</td>
<td>320</td>
<td>69.3</td>
</tr>
<tr>
<td>RTE</td>
<td>$[0.0,0.1,0.2,0.3]$</td>
<td>640</td>
<td>63.8</td>
</tr>
<tr>
<td>RTE</td>
<td>$[0.0,0.1]$</td>
<td>256</td>
<td>65.3</td>
</tr>
</tbody>
</table>

Table 1: Effects of different dropout masks and number of views on CoLA and RTE tasks. Score denotes Matthews Correlation Coefficient.
Table 2: GLUE Test results. BERT\textsubscript{base} - Standard is our implementation using the reported hyper-parameters in Devlin et al. (2018) for each task. BERT\textsubscript{base} - Dropout Augmented is the standard version trained also on augmented samples. Matthews Correlation Coefficient is reported for CoLA, Pearson/Spearman correlations for STS-B, F1/Accuracy for MRPC, F1 score for QQP, and accuracy scores are reported for the other tasks.

BERT\textsubscript{base} architecture with the one of a SupCL-Seq-empowered BERT\textsubscript{base} model on numerous tasks from the GLUE benchmark (for a detailed description of each task see Wang et al. (2018)). GLUE also includes a regression task (i.e., STS-B), which requires no architecture modifications. For the classification experiments, we deploy the hyper-parameters reported in Devlin et al. (2018). Appendix A details our grid-search-based training details for SupCL-Seq. Results are described in Table 2, rows one and two. As it can be noticed, in all cases the SupCL-Seq-empowered BERT\textsubscript{base} model obtains equal or higher performance compared to the standard implementation.

3.2.1 Is it the dropout augmentation or the loss-function?

In order to study whether the performance gain observed in the previous experiments is solely due to the dropout augmentation, we ran a new set of experiments on the smaller datasets (i.e., MRPC, RTE, STS-B and SST-2) in which the standard BERT\textsubscript{base} is trained also on the augmented samples (for the training parameters, see Appendix A). Table 2, third row, shows the results. While we notice performance gains compared to the BERT\textsubscript{base} - Standard in a few tasks, augmentation does not always help. For example, the score for the augmented row is lower in the RTE dataset. Interestingly, dropout augmentation significantly hurts the performance (≈ 8 points) in STS-B dataset, where MSE loss is employed. We also observe that for all CoLA, MRPC and STS-B, SupCL-Seq outperforms the augmented variant, suggesting that its gains are due to the combination of augmentation and contrastive learning, rather than from only the former.

Table 3: Comparison of unsupervised and supervised contrastive loss.

3.3 Supervised Versus Unsupervised contrastive Learning

Since, to the best of our knowledge, the only previous attempt of using contrastive learning for improving sequence representation in NLP was performed by Gao et al. (2021) – they used a self-supervised approach to improve the semantic representation, adopting a loss similar to Equation 1 –, in Table 3.3 we compare the performance of a linear layer trained on top of their representations with the one of a linear layer trained on top of our representations, which are instead optimized in a supervised fashion while the parameters of the base model are kept frozen. SupCL-Seq significantly outperforms the re-implementation of Gao et al. (2021), with larger gains in non-semantic tasks (e.g. CoLA), suggesting that our representations are optimized for the given downstream tasks.

4 Discussion and Conclusion

In this paper, we introduced SupCL-Seq a supervised contrastive learning framework for optimizing sequence representations for downstream tasks. In a series of experiments, we showed that SupCL-Seq leads to large performance gains in almost all GLUE tasks when compared to both a standard BERT\textsubscript{base} architecture and an augmented BERT\textsubscript{base} (i.e., improvements are not only due to augmentation). We also investigated the effect of number and level of dropout passes, finding that this has to be treated as a task-dependent hyper-parameter, to be fine tuned in a validation set. Finally, we compared our supervised approach to the self-supervised method by Gao et al. (2021), showing...
consistent performance improvements, especially in non-semantic tasks, where the self-supervised approach is weaker. These encouraging results open the door to multi-task learning applications of SupCL-Seq, where the optimization needs to be constrained towards multiple objectives.

Acknowledgments

We would like to thank the reviewers and the chairs for their insightful reviews and suggestions.

References


### A Training Details

<table>
<thead>
<tr>
<th>Task</th>
<th>Learning Rate</th>
<th>Batch Size</th>
<th>dropout</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoLA</td>
<td>5e − 05</td>
<td>128</td>
<td>[0.0, 0.1, 0.2]</td>
</tr>
<tr>
<td>MRPC</td>
<td>1e − 4</td>
<td>128</td>
<td>[0.0, 0.05, 0.1, 0.2]</td>
</tr>
<tr>
<td>RTE</td>
<td>1e − 4</td>
<td>48</td>
<td>[0.0, 0.1, 0.2]</td>
</tr>
<tr>
<td>STS-B</td>
<td>1e − 4</td>
<td>64</td>
<td>[0.0, 0.05, 0.1, 0.2]</td>
</tr>
<tr>
<td>SST-2</td>
<td>5e − 05</td>
<td>320</td>
<td>[0.0, 0.1, 0.2]</td>
</tr>
<tr>
<td>WNLI</td>
<td>1e − 04</td>
<td>320</td>
<td>[0.0, 0.1, 0.2]</td>
</tr>
<tr>
<td>QNLI</td>
<td>5e − 05</td>
<td>48</td>
<td>[0.0, 0.2]</td>
</tr>
<tr>
<td>QQP</td>
<td>5e − 05</td>
<td>16</td>
<td>[0.0, 0.2, 0.3, 0.4, 0.5]</td>
</tr>
<tr>
<td>MNLI</td>
<td>5e − 05</td>
<td>8</td>
<td>[0.1, 0.1]</td>
</tr>
</tbody>
</table>

Table 4: Contrastive learning training details per GLUE task. All of the tasks were trained for 5 epochs (except QNLI, QQP and MNLI that were trained for 2, 1 and 3 epochs respectively) and $\tau = 0.05$.

SupCL-Seq is implemented on top of the Huggingface’s trainer python package (Wolf et al., 2019)⁴. For the sim(.) (similarity) function, we employed inner dot product. For supervised contrastive learning, we employed the hyperparameters detailed in Table 4. We used a grid search strategy for our hyperparameter optimization, where the number of dropouts and their corresponding probability were set to two (i.e. [0.1, 0.1]) and five respectively ([0.0, 0.1, 0.2, 0.3, 0.4]). For the learning rate we employed a range of [5e − 05, 1e − 4].

⁴https://github.com/huggingface/transformers