Efficient Contrastive Learning via Novel Data Augmentation and Curriculum Learning

Preprint · September 2021

DOI: 10.48550/arXiv.2109.05941

Kim Jiseon
Dong-Eui University
46 PUBLICATIONS **1,768** CITATIONS

Efficient Contrastive Learning via Novel Data Augmentation and Curriculum Learning

Seonghyeon Ye, Jiseon Kim, Alice Oh
School of Computing, KAIST
{vano1205, jiseon_kim}@kaist.ac.kr
alice.oh@kaist.edu

Abstract

We introduce EfficientCL, a memory-efficient continual pretraining method that applies contrastive learning with novel data augmentation and curriculum learning. For data augmentation, we stack two types of operation sequentially: cutoff and PCA jittering. While pretraining steps proceed, we apply curriculum learning by incrementing the augmentation degree for each difficulty step. After data augmentation, we apply contrastive learning on projected embeddings of original and augmented examples. When fine-tuned on GLUE benchmark, our model outperforms baseline models, especially for sentence-level tasks. Additionally, this improvement is achieved with only 70% of computational memory compared to the baseline model. [1]

1 Introduction

Many state-of-the-art language models involve the paradigm of unsupervised pretraining followed by fine-tuning on downstream tasks (Devlin et al., 2019; Liu et al., 2019; Brown et al., 2020). However, pretraining a language model from scratch with a huge corpus has high computational costs. One way to cut down the cost is to use continual training of a pretrained model which could improve the language model with less computation (Giorgi et al., 2021).

Contrastive learning is effective for self-supervised learning for image classification (Chen et al., 2020b,c), and it works by allowing the model to put similar examples close and different examples far from one another. Often in contrastive learning, data augmentation is used to make the positive pairs. Recent papers describe how to apply contrastive learning to the language domain (Meng et al., 2021; Gunel et al., 2020; Qu et al., 2020; Wu et al., 2020), and even a combination of contrastive learning with continual pretraining (Giorgi et al., 2021). However, because of the sequential nature of language, it is difficult to apply data augmentation methods used in images directly to language modeling.

Additionally, curriculum learning is a powerful training technique for deep networks (Hacohen and Weinshall, 2019; Cai et al., 2018). By training easy to hard examples in order, it facilitates faster convergence, leading to better performance. Other studies show that it is also effective for language modeling (Xu et al., 2020; Wei et al., 2021; Press et al., 2021; Li et al., 2021).

We propose an efficient yet powerful continual pretraining method using contrastive learning and curriculum learning. The contribution of this paper is as follows:

- We suggest a novel data augmentation method for contrastive learning: first cutoff, then PCA jittering. This leads to robustness to sentence-level noise, resulting in a better sentence-level representation.
- We apply curriculum learning by increasing the noise degree of augmentation for each level of difficulty. This leads to faster convergence at the pretraining stage.
- In addition to outperforming baseline models on the GLUE benchmark, our model is memory efficient and applicable to a wider range of corpora.

2 Method

The overall learning process of EfficientCL is illustrated in Figure 1.

2.1 Sampling Anchor from Text

We modify the method from Giorgi et al. (2021) which samples anchors and positive instances. For each document from the pretraining corpus, a sequence of 512 tokens is randomly sampled, referred to as an anchor. The anchor goes through

[1] Our code is publicly available at `https://github.com/vano1205/EfficientCL`

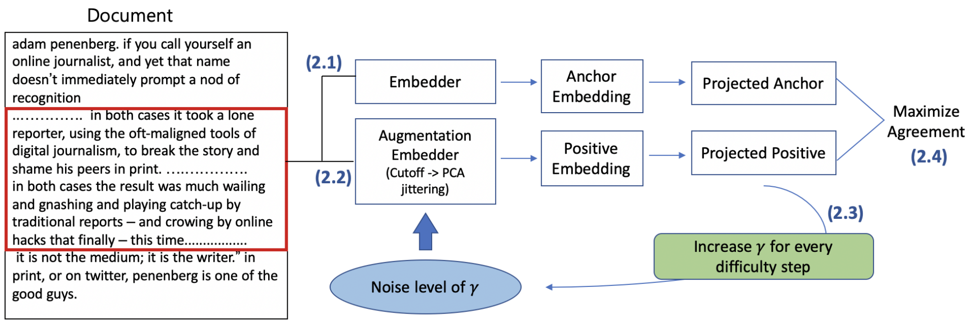
Document
adam penenberg. if you call yourself an online journalist, and yet that name doesn't immediately prompt a nod of recognition
.... in both cases it took a lone reporter, using the oft-maligned tools of digital journalism, to break the story and shame his peers in print. .... in both cases the result was much wailing and gnashing and playing catch-up by traditional reports – and crowing by online hacks that finally – this time................
it is not the medium; it is the writer." in print, or on twitter, penenberg is one of the good guys.
(2.1)
(2.2)
Embedder
Augmentation Embedder (Cutoff -> PCA jittering)
Anchor Embedding
Positive Embedding
Projected Anchor
Projected Positive
Maximize Agreement
(2.4)
(2.3)
Increase γ for every difficulty step
Noise level of γ

Figure 1: For continual pretraining, we sample a fixed-length sequence from each document and obtain the anchor embedding. We obtain the positive embedding from the augmentation embedder with cutoff and PCA jittering. While training, we increase the augmentation degree for every difficulty step. From projected anchor and positive embedding, we apply contrastive learning to maximize agreement.

the RoBERTa encoder to get the embedding of the anchor sequence.

2.2 Data Augmentation

We introduce two data augmentation methods, *cutoff* and *PCA jittering*. These methods were inspired by SimCLR (Chen et al., 2020b), which showed that random cropping followed by color jittering is effective for contrastive learning among various augmentation combinations. Likewise, our method applies cutoff first, then PCA jittering. We apply on one of the inner layers of RoBERTa model randomly sampled from {7,9,12} layers. These layers contain the most of syntactic and semantic information (Chen et al., 2020a).

Cutoff Augmentation Cropping-based data augmentation is simple but effective for natural language understanding (Meng et al., 2021; Shen et al., 2020). Among various methods introduced in Shen et al. (2020), we use the most robust *span cutoff* method. Previous span cutoff method makes a continuous portion of sequences with a specific ratio to zero on the embedding layer. On the other hand, we apply this operation to the hidden states of an inner layer of RoBERTa.

PCA jittering Augmentation Our method is similar to the widely-used color jittering method in Krizhevsky et al. (2012) for computer vision domain, but we apply the operation on the hidden states. If the original hidden state is $h = [h_0, h_1, ..., h_d]$, hidden state after PCA jittering would be $h = [h_0 + \delta, h_1 + \delta, ..., h_d + \delta]$ where $\delta = [p_1, p_2, ..., p_d][\alpha\lambda_1, \alpha\lambda_2, ..., \alpha\lambda_d]^T$, $\alpha \sim N(0, \sigma^2)$, d is the dimension of hidden states, p_i and λ_i are the ith eigenvector and eigenvalue respectively.

2.3 Simple Curriculum Learning Method

We apply curriculum learning during the data augmentation process by increasing the noise level for each difficulty step, which is cropping ratio for cutoff and standard deviation of the noise hyperparameter for PCA jittering method. As the noise level gets larger, the augmented positive would be more dissimilar from the anchor, resulting in a harder example for contrastive learning. For both augmentation methods, the noise level is initially set as 0.01 and incremented until 0.1. [2]

We introduce and compare two curriculum learning methods: *Discrete* and *Continuous*. First, the discrete curriculum learning divides the pretraining step into ten steps and increases the augmentation level for each step. Second, the continuous curriculum learning increases the augmentation level continuously for every iteration of training, starting from 0.01 until 0.1.

2.4 Contrastive Learning Framework

For each positive and anchor sequence embedding denoted as e_i, the projected embedding is obtained by $z_i = g(e_i) = W^{(2)}\psi(W^{(1)}e_i)$ where $W^{(1)}, W^{(2)}$ are trainable weights and ψ is a ReLU non-linearity (Chen et al., 2020b). From the projected anchor and the projected positive, we apply contrastive learning. For a minibatch of size N, there is an anchor and a positive instance for each document, resulting in $2N$ instances in total. The contrastive loss for a positive pair $\{z_{2i-1}, z_{2i}\}$ is

$$L_{contrastive} = \sum_{i=1}^{N} l(2i-1, 2i) + l(2i, 2i-1)$$

$$l(i,j) = -log \frac{\exp(\mathrm{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} 1_{[i \neq k]} \exp(\mathrm{sim}(z_i, z_k)/\tau)}$$

[2]The rationale is explained in Appendix A.2.

Model	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST	STSB	Avg.
Roberta-base	59.38	87.76	90.07	92.44	89.44	76.90	94.15	90.55	85.09
DeCLUTR	**59.60**	87.75	89.63	92.70	**89.58**	75.63	94.04	90.48	84.93
EfficientCL	59.05	**87.80**	**90.68**	**92.81**	89.51	**77.62**	**94.61**	**90.61**	**85.34**

Table 1: Evaluation result on GLUE development set. All performance is median of five runs with random seeds except RTE, which reports median of ten runs. The evaluation metric for each task is as follows: Matthews correlation for CoLA, average of Pearson and Spearman correlation for STS, average of accuracy and F1 for MRPC and QQP, and accuracy for other tasks.

$$\mathrm{sim}(z_i, z_j) = z_i^T z_j / (\|z_i\| \, \|z_j\|)$$

where τ refers to the temperature hyperparameter.

Our training process is applied continuously on the pretrained RoBERTa model. To prevent catastrophic forgetting of token-level MLM objective (Chronopoulou et al., 2019), we add the loss from the MLM objective to the contrastive objective,

$$L_{total} = L_{MLM} + L_{contrastive}.$$

3 Experiments

3.1 Pretraining and Finetuning

The dataset used for pretraining is OpenWebText corpus containing 495,243 documents with a minimum token length of 2048, which is the same setting as the DeCLUTR model (Giorgi et al., 2021). We use a NVIDIA Tesla V100 GPU for pretraining, which takes 19.7 hours for training. For finetuning evaluation, we use the development set in GLUE benchmark. For small datasets (CoLA, STSB, MRPC, RTE), the model is finetuned for 10 epochs, and for the rest (MNLI, QQP, SST, QNLI), it is trained for 3 epochs. We report median values over 5 random initializations for all tasks except RTE. For RTE, we report median values of 10 runs due to the high variance of the performance.

3.2 Baselines

We compare our model with the pretrained RoBERTa-base to check the effectiveness of our continual pretraining method. We observe whether our contrastive objective complements the MLM objective by improving the performance of sentence-level tasks. We also compare with the DeCLUTR model (Giorgi et al., 2021) which is continually pretrained from RoBERTa-base with contrastive learning. It samples adjacent positive sequences given an anchor instead of applying data augmentation or curriculum learning. [3]

[3]Additional variants of baseline models are shown in Appendix A.1.

4 Results

4.1 Overall Results

We compare the performance of our model with two baseline models, and the results are shown in Table 1. Overall, our model performs better than the baseline models, especially for small datasets such as MRPC and RTE. For sentence-level tasks (all except CoLA), our model performs better than RoBERTa-base and better than DeCLUTR except QQP. Since our pretraining method makes the model robust to sentence-level noise, it captures better representations of sentences. For the CoLA dataset, our model performs poorly because it is trained to be robust from small to big noises sequentially. The linguistic acceptability task is sensitive to noise, meaning that small changes in a sentence could lead to a different label. Our method hardly differentiates when the change is small, leading to wrong predictions.

4.2 Ablation Study

We conduct ablation studies on the GLUE development set, and the results are shown in Table 2.

Data Augmentation Method To observe the impact of each augmentation method on the performance, we conduct experiments with only one of the two methods. For all tasks except QQP, each method underperforms the combination of both. This is consistent with Chen et al. (2020b) which suggests that a composition of multiple augmentations is effective. Because each of the augmentations is relatively simple, applying only one method leads to a small degree of augmentation. This disturbs effective learning because the benefit of *hard positive* in contrastive learning is neglected.

We highlight the novelty of our augmentation method by comparing with Wei et al. (2021) which uses the popular EDA augmentation method. Our method is different from Wei et al. (2021) in that curriculum learning and data augmentation are applied at the continual pretraining stage, not at the finetuning stage. The augmentation level is from

Curriculum	Cutoff	PCA	EDA	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST	STSB	Avg.
Discrete	✓			58.04	87.69	90.07	92.60	89.54	76.90	94.27	90.42	84.94
Discrete		✓		59.07	87.56	89.89	92.62	89.44	76.17	94.15	90.48	84.92
Discrete			✓	58.04	87.72	90.27	92.66	89.49	76.71	93.92	90.49	84.91
No Curr	✓	✓		**60.33**	87.70	90.27	92.51	89.51	76.17	94.04	90.51	85.13
Continuous	✓	✓		58.82	87.71	90.27	92.60	**89.57**	76.35	94.27	90.45	85.01
Discrete	✓	✓		59.05	**87.80**	**90.68**	**92.81**	89.51	**77.62**	**94.61**	**90.61**	**85.34**

Table 2: Ablation studies on GLUE dev set. The first three rows show the results of different data augmentation. The fourth and fifth test the impact of curriculum learning. The last setting is the best, using both cutoff and PCA jittering and discrete curriculum learning.

0 to 0.5 with 6 discrete steps, the same as the paper. The results show that EDA underperforms our method for all tasks, and although EDA is a simple data augmentation approach, it does not perform well in the continual pretraining setting.

Curriculum Learning Method We set three different experiment settings to see how curriculum learning influences performance. The first setting is *No Curr*, which randomly selects one ratio out of ten ranging from 0.01 to 0.1 with 0.01 interval for every iteration. The second and third settings are *Continuous* and *Discrete* explained in 2.3. For all tasks except for QQP, the discrete setting performs best. Continuous method goes over simple examples too fast, leading to confusion rather than fast convergence (Hacohen and Weinshall, 2019).

Looking at the effect of curriculum learning, it is effective for most of the tasks, especially for QNLI and SST. It facilitates faster convergence to the pretraining objective as shown in Figure 2, leading to better performance on downstream tasks. Surprisingly, the method without curriculum learning results in the highest performance on CoLA. Due to catastrophic forgetting suggested in (Xu et al., 2020), EffectiveCL is likely to be robust to larger noise at the end of training. Therefore, because random shuffling can better differentiate small noise, it is suitable for noise-sensitive tasks.

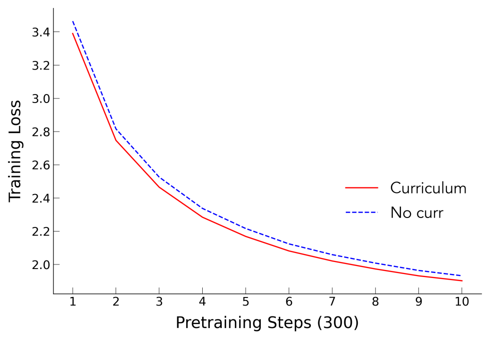

Figure 2: Average batch training loss of first 3,000 steps with and without curriculum learning.

4.3 Efficiency and Applicability

Another advantage of EfficientCL is its memory efficiency. We compare with DeCLUTR model since both train with continual pretraining. Figure 3 shows that with 70% of memory, our model performs better on GLUE. By applying data augmentation on the anchor for contrastive learning instead of sampling neighboring positives, our model needs only one anchor to sample, leading to reduced computational costs but better performance.

Additionally, for our model, pretraining is possible with documents having more than 512 tokens. This is significant for applicability since DeCLUTR needs at least 2048 tokens for each document to sample the anchors and the positive spans. From the OpenWebText corpus, documents with more than 512 tokens result in 4,126,936 documents, more than 8 times compared to the current setting. Although we used the same pretraining data as DeCLUTR for fair comparison, using more documents would lead to better performance.

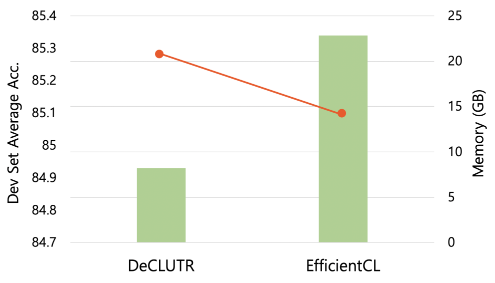

Figure 3: Efficiency of our model compared with DeCLUTR.

5 Conclusion

In this paper, we propose EfficientCL, a pretrained model with efficient contrastive learning method utilizing data augmentation and curriculum learning. Our data augmentation process is divided into *cutoff* and *PCA jittering*. Rather than using one, combining two augmentation methods significantly boosts the performance. Additionally, by incrementing the augmentation level for each difficulty step, the model achieves faster convergence, resulting in better performance on sentence-level tasks. Our method is also memory efficient and applicable for a wide range of corpora at pretraining stage.

Acknowledgements

The authors want to thank Kyubeom Han for helpful discussions and support. This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2017-0-01779, A machine learning and statistical inference framework for explainable artificial intelligence).

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33*, pages 1877–1901.

Qi-Zhi Cai, Chang Liu, and Dawn Song. 2018. Curriculm adversarial training. *arXiv preprint arXiv 1805.04807*.

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020a. MixText: Linguistically-informed interpolation of hidden space for semi-supervised text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2147–2157, Online. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020b. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. 2020c. Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029*.

A. Chronopoulou, C. Baziotis, and A. Potamianos. 2019. An embarrassingly simple approach for transfer learning from pretrained language models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, page 2089–2095.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanov. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, page 4171–4186.

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

Beliz Gunel, Jingfei Du, Alexis Conneau, and Ves Stoyanov. 2020. Supervised contrastive learning for pretrained language model fine-tuning. *arXiv preprint arXiv:2011.01403*.

Guy Hacohen and Daphna Weinshall. 2019. On the power of curriculum learning in training deep networks. In *ICML*, page 2535–2544.

Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. 2020. Hard negative mixing for contrastive learning. In *Advances in Neural Information Processing Systems 33*, pages 21798–21809.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *In Advances in Neural Information Processing Systems*, pages 1097–1105.

Conglong Li, Minjia Zhang, and Yuxiong He. 2021. Curriculum learning: A regularization method for efficient and stable billion-scale gpt model pretraining. *arXiv preprint arXiv 2108.06084*.

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

Yu Meng, Chenyan Xiong, Payal Bajaj, Saurabh Tiwary, Paul Bennett, Jiawei Han, and Xia Song. 2021. Coco-lm: Correcting and contrasting text sequences for language model pretraining. *arXiv preprint arXiv:2102.08473*.

Ofir Press, Noah A. Smith, and Mike Lewis. 2021. Shortformer: Better language modeling using shorter inputs. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 5493–5505.

Yanru Qu, Dinghan Shen, Yelong Shen, Sandra Sajeev, Jiawei Han, and Weizhu Chen. 2020. Coda: Contrast-enhanced and diversity-promoting data augmentation for natural language understanding. *arXiv preprint arXiv:2010.08670*.

Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. 2020. Contrastive learning with hard negative samples. *arXiv preprint arXiv:2010.04592*.

Dinghan Shen, Mingzhi Zheng, Yelong Shen, Yanru Qu, and Weizhu Chen. 2020. A simple but tough-to-beat data augmentation approach for natural language understanding and generation. *arXiv preprint arXiv:2009.13818*.

Jason Wei, Chengyu Huang, Soroush Vosoughi, Yu Cheng, and Shiqi Xu. 2021. Few-shot text classification with triplet networks, data augmentation, and curriculum learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, page 5493–5500.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Madian Khabsa, Fei Sun, and Hao Ma. 2020. Clear: Contrastive learning for sentence representation. *arXiv preprint arXiv:2012.15466*.

Benfeng Xu, Licheng Zhang, Zhendong Mao, Quan Wang, Hongtao Xie, and Yongdong Zhang. 2020. Curriculum learning for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6095–6104, Online. Association for Computational Linguistics.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2021. Revisiting fewsample bert fine-tuning. In *In International Conference on Learning Representations*.

Model	Roberta	Continual Roberta	EfficentCL
CoLA	**59.38**	59.07	59.05
MNLI	87.76	87.65	**87.80**
MRPC	90.07	90.14	**90.68**
QNLI	92.44	92.60	**92.81**
QQP	89.44	89.47	**89.51**
RTE	76.90	76.90	**77.62**
SST	94.15	93.92	**94.61**
STSB	90.55	90.45	**90.61**
Avg.	85.09	85.03	**85.34**

Table 3: Evaluation result on GLUE development set. All performance is median of five runs with random seeds.

A Appendix

A.1 Variants of Baseline Models

Continually Pretrained RoBERTa-Base

For a fair comparison of pretrained RoBERTa-base, we also continually pretrain this baseline on OpenWebText. All the settings such as sampling procedure and batch size are the same as EfficientCL except that none of data augmentation, curriculum learning, or contrastive learning is used. Only a naive MLM objective is used for continual pretraining. Table 3 shows that EfficientCL outperforms continually pretrained RoBERTa for all tasks except for CoLA, which shows comparable results. For many tasks (CoLA, MNLI, SST, STSB), continually pretrained RoBERTa even underperforms pretrained RoBERTa. This shows that straightforward continual pretraining is ineffective because training hyperparameters such as learning rate scheduling are completely changed from the pretraining stage. In contrast, our EfficientCL shows robust performance, although the training objective has changed for continual pretraining setting.

Naive Curriculum Learning Method

Many traditional ways of applying curriculum learning on natural language use sentence length, word rarity, or additional teacher model to evaluate the difficulty for each sentence (Xu et al., 2020). However, these naive methods are infeasible at the continual learning stage. First of all, using multiple teacher models is memory inefficient since many language models are needed. Also, sorting sentences in respect to sentence length or word rarity is a huge overhead when the corpus is large. Compared to these naive methods, our curriculum learning, which augments the noise level of data augmentation for contrastive learning, is efficient.

A.2 Hyperparameter Settings

Number of Anchor

Different from DeCLUTR (Giorgi et al., 2021), we sample one anchor per document instead of two. Multiple anchors improve the performance by *hard negative mining* because hard negative mining is effective for contrastive learning (Kalantidis et al., 2020; Robinson et al., 2020). Sampling multiple anchors from the same document would lead to mining negatives that are similar, resulting in a harder task for the model to learn. However, we empirically found out that the number of epoch for the model to converge at pretraining stage is insufficient when training with 2 anchors with data augmentation. This is a tradeoff between training efficiency and model performance. We expect that more training epochs with multiple anchors per document would lead to better performance.

Data Augmentation Ratio

For cutoff method, Shen et al. (2020) shows that there exists a sweet point for good performance of span cutoff, which is cropping ratio from 0 to 0.1. For PCA jittering method, the original paper set the standard deviation as 0.1. We found out that this value is also appropriate for large dimensions of hidden states.

Curriculum Learning Step

Xu et al. (2020) suggests that the difficulty step of 10 is appropriate for curriculum learning in natural language domain.

Experiments

For continual pretraining setting, we apply contrastive learning with a minibatch size of 4 and a temperature of $\tau = 0.05$. The model is trained for 1 epoch using AdamW optimizer with a learning rate 5e-05 and 0.1 weight decay. Slanted triangular LR scheduler is used for the scheduler, and gradients are scaled to 1.0 norm.

For finetuning evaluation setting, it is done with a minibatch size of 16 and optimized using Adam optimizer with learning rate of 1e-05. The number of epochs is set as 3 epochs for small datasets (CoLA, STSB, MRPC, RTE) and 10 epochs for the rest (MNLI, QQP, SST, QNLI). Zhang et al. (2021) showed that conventional 3 epoch finetuning is suboptimal for small datasets due to instability. We also empirically found out that training for longer epochs significantly improves the performance for small datasets.