Contrastive Learning for Sentiment Classification

Studer Justin, Ghodki Hrishikesh, Neuner-Jehle Joel, Group: Taskforce

Department of Computer Science, ETH Zurich

Abstract

Contrastive Learning provides a framework for comparing many data types with each other. We propose to use it for Sentiment Classification of informal text snippets and find that it performs on-par with methods that rely on the same base model, with the potential to outperform them when using larger datasets. We also explore the viability of hard sample mining in this context and find that it does not offer any benefits.

1. Introduction

With the vast amount of data produced by users of social media websites and the potential value of said data, there is an ever increasing desire to automate data analysis. Behind the content created and published on social media platforms is usually a human user who associates certain feelings with this content. Therefore, one particular analysis type of interest is sentiment classification for informal texts, such as can be found on websites like Twitter and Facebook. With modern technology, these associated sentiments can be classified and estimated by trained algorithms with the help of deep neural networks. This task has been explored in numerous works, such as (Yang et al., 2019; Jiang et al., 2019). Many of the more recent methods for this task rely on transformer type models, which are able to process contextual information well by using a learnable latent space embedding. Typically, those models are trained using binary cross-entropy as a training objective.

A different objective for binary classification which has also proven useful for other tasks - such as automated code search (Anonymous, 2021) - leverages the so-called contrastive learning objective (Gutmann & Hyvärinen, 2010), which compares the prediction's embedding to the embeddings of other samples instead of producing an immediate class estimate. In this work, we explore the application of the contrastive loss objective to improve sentiment classification and compare our results to state-of-the-art baselines, with and without further training augmentations. Future sentiment classification algorithms can potentially benefit from the usage of the contrastive loss and could be taken further to more complex environments. Our contrastive model, the benchmarked baseline classification models including experiment ablations and a list of required Python libraries are provided on `https://github.com/jstuder3/cil_taskforce`.

2. Related Work

Sentiment analysis has been an active area of research for many decades and describes the systematic extraction and processing of subjective information, such as emotion or personal opinion. In the industry, it is typically applied to online product reviews and, especially in recent years, social media texts on Twitter and other competing social media platforms. With the most recent progress in the field of natural language processing, sentiment analysis has taken a huge leap and advanced to one of the fastest growing research areas (Mäntylä et al., 2016).

Types of sentiment analysis can range from a graded approach, e.g. one to five stars corresponding to the level of customer satisfaction given a text based review, to a binary representation (positive/negative), as it is used in our work. Classic emotion detection models use a static lexicon to correlate words or phrases with sentiments (Nielsen, 2011; Kiritchenko et al., 2014), whilst more recent approaches use efficient and well-performing deep neural networks trained on a large dataset of informal texts (Yang et al., 2019; Jiang et al., 2019). Many of the latter models use the transformer-type BERT architecture, consisting of several encoder, decoder and attention layers. (Vaswani et al., 2017; Devlin et al., 2018) In particular, these models are typically fine-tuned using a cross-entropy objective.

Sentiment analysis has a wide range of applications and is considered a very useful tool in today's ever more technological world, such as business, government and biomedicine: Companies can better analyse customers' feedback and improve their business strategy. Recommender services are getting stronger and medical processes that involve natural language - such as in the field of mental health or patient consultation - can be sped-up and assisted by computer programs. While the advances in the mentioned fields are beneficial, sentiment analysis still suffers from the challenges related to natural language processing, such as inconsistent language and labelling noise (Dang et al., 2020).

Contrastive learning is an objective function that has its roots in self-supervised representation learning (Gutmann & Hyvärinen, 2010). It is particularly useful when we want to find correlations between inputs of different data types, such as text and images (Chen et al., 2020). Instead of producing immediate class predictions, this objective relies on similarity between embeddings of different samples for its predictions. The goal is hence that the encoder should map corresponding inputs to a similiar part of the latent space. This is achieved by putting sample embeddings into ”contrast” with each other in the objective function. This effectively pulls corresponding embeddings closer to each other, whilst pushing apart embeddings that do not correspond to each other. Further applications of contrastive learning can be found in image classification, image-to-image translation (Park et al., 2020) or sentence embedding learning (Gao et al., 2021).

A particular framework that uses contrastive learning is called Momentum Contrast (MoCo), which uses a queue holding the embeddings from previous iterations to stabilize training (He et al., 2020). While MoCo was proposed for unsupervised representation learning, it is also possible to use this framework for labelled data, as further described in Section 3.

One way to improve training with contrastive learning is to mine hard samples, i.e. to specifically feed the encoder samples which it struggles to map close to samples of the same class. It has been shown that this has the potential to improve generalization capability of the model (Anonymous, 2021).

3. Method

For our analysis we use the contrastive learning loss, which we can define as

$$L_{Contrast}(q) = \frac{1}{\|B_q\|} \sum_{i \in B_q} -log \frac{exp(q \cdot k_i/\tau)}{\sum_{j \in B} exp(q \cdot k_j/\tau)} \quad (1)$$

where q is the normalized embedding of the current input text, B is the current batch with k_t being the normalized embedding of t-th input in the batch, B_q is the subset of B that contains the samples that are in the same sentiment class as q and τ is a temperature hyperparameter. This loss function will have a small value when q and k_p with $p \in B_q$ (positive samples) tend to have a large dot product (i.e. they are similar) and q and k_n with $n \in B \backslash B_q$ (negative samples) tend to have a small dot product (i.e. they are dissimilar).

As is immediately visible from the definition, this loss function requires embeddings of several samples to produce good results. In fact, the quality of this objective in this form is most of all restricted by the batch size. Since we often cannot arbitrarily increase the batch size due to memory restrictions, we need other ways to obtain more embeddings to use in this objective.

One way to obtain more embeddings is by encoding the entire database prior to training. However, this comes with the issue of staleness: As training progresses, the newly produced embeddings could differ drastically from the initial outputs. Updating all of them is prohibitively expensive, so we can instead keep just a subset of all embeddings in a queue and replace stale ones continuously. This was proposed in (He et al., 2020), which also recommended to use a ”slow” encoder to overcome fluctuations in the produced mappings. This slow encoder is essentially a weighted average over iterations of the main/fast encoder's parameters. The queue and this ”momentum update” are the key components of the Momentum Contrast (MoCo) framework. A diagram of it is shown in Figure 1.

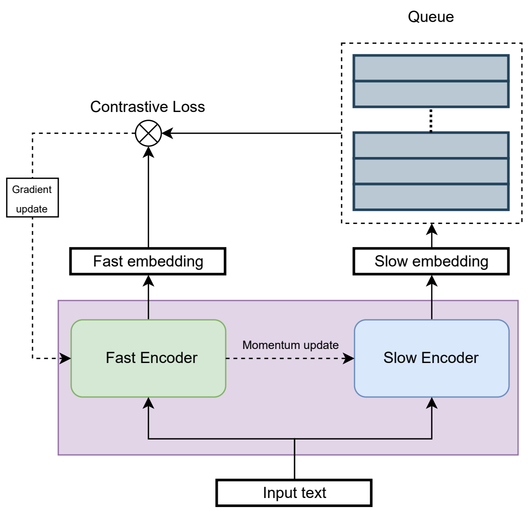

Figure 1. Schematics of the MoCo framework

In addition to using embeddings from prior iterations in the objective, we can also employ hard sample mining, as proposed in (Anonymous, 2021). Concretely, we want to feed samples to the encoder which are particularly hard for it to map to the ”correct” neighbourhood. By doing this, we hope to make the model more robust to these ”difficult cases”. We can find hard samples by keeping either embeddings of all samples or only a subset like above. We then compute the similarity on this data to extract the indices of samples which have undesirable similarity. In particular, we are looking for samples for each element in our batch which have the top k smallest dot products but are of the same sentiment class (hard positives). The same applies to the ones which have the top k largest dot products but are of the respective other sentiment class (hard negatives). Afterwards, we simply produce updated embeddings based on these indices, which can then be included in the contrastive loss. Note that staleness is less of an issue here, since we only look up indices of potentially troublesome samples, not their embeddings.

Prior to inference, we encode the entire training dataset. For every new query, we produce an embedding using the fast encoder and compare it to all of the training embeddings. Based on the mean similarity per sentiment class, the input is classified to be either of positive or negative sentiment.

As the baseline for our encoder, we used BERT base (Devlin et al., 2018), a pretrained model on English natural language. BERT has been trained in a self-supervised fashion on raw texts and achieves state-of-the-art masked language modeling. We apply this pretrained model to obtain embeddings and use them downstream for sentiment classification. For this, we fine-tune it using the methods outlined above.

4. Experiments

We perform experiments with our contrastive model on a single text-to-sentiment classification task using a dataset of 2.5M tweets. Each tweet is assigned a corresponding positive or negative sentiment label. The training and evaluation process was then done using only 200k of those 2.5M tweets due to excessive memory consumption during the similarity calculation of the contrastive learning algorithm. Preprocessing of the data was done following the approach of Horne et al. to preserve the raw text of the tweets and leave out sentiment-independent information, such as the author's name, URLs or excessive whitespaces (Horne et al., 2020). We ran all experiments multiple times to obtain empirical estimates of the standard deviation (shown in brackets).

Table 1 shows a comparison between the best configuration of different models we found.

Model	**Accuracy**
Logistic Regression	82.32
BCE Bert	87.96 (0.07)
GRUBERT	88.05 (0.12)
Contrastive BERT (Ours)	**88.16 (0.17)**

Table 1. Accuracy of the baseline models and our contrastive learning model.

In the following, we show the experiments we ran to arrive at these values. Unless otherwise noted, we used the hyperparameters listed in Table 2. The training and validation computations were done on the ETH internal Euler cluster, which uses a Nvidia RTX 2080 Ti GPU and several Intel CPUs. To counter the problem of overfitting to the data distribution during training, the learning rate decays exponentially after each epoch. As proposed by (Horne et al., 2020), we applied early stopping with a patience of 3 epochs, where the indicator for stopping is the validation loss instead of the usually used accuracy metric. We allowed a maximum computation time of twelve hours, which was hardly ever reached due to an early termination after a few sufficient epochs.

Hyperparameter	**Value**
Learning rate	1e-5
Training batch size	64
Validation batch size	256
Temperature τ	0.07
Sample queue size	16384
Momentum update weight	0.99
Number of hard samples	0

Table 2. Base hyperparameters used for the experiments

In the experiments, the performance of our contrastive learning model is compared to a variety of other sentiment classification models: A pretrained BERT (Devlin et al., 2018) model which we fine-tune using binary cross-entropy instead of contrastive learning , GRUBERT (Horne et al., 2020), which uses GRUs to fuse multiple layers of BERT with each other, and a set of classifiers acting as a commonly used baseline. Ablations on our contrastive learning model regarding the learning rate, the sample queue length and the usage of hard samples during training have also been evaluated separately and show quite differing results. The following tables show our ablation experiments for the contrastive learning approach. Ablations for the other methods can be found in the Appendix document in our GitHub repository (see Section 1).

Learning Rate	1e-4	5e-5	1e-5	5e-6	1e-6
Ours w/o queue	86.56 (0.37)	87.72 (0.10)	**87.95 (0.23)**	87.81 (0.18)	87.65 (0.06)

Table 3. Learning Rate Experiments for our method that trains BERT using the the constrastive learning objective

Using the best learning rate from above, we conducted further experiments wherever applicable and reasonable:

		Queue size			
		1024	4096	16384	65536
U. w.	0.99	87.85 (0.11)	87.84 (0.01)	**88.16 (0.17)**	88.06 (0.07)
	0.999	87.87 (0.24)	88.16 (0.24)	87.81 (0.40)	87.61 (0.00)

Table 4. Update weight and queue size experiments for BERT trained with the contrastive learning objective

In addition to quantitative measurements, we also produced a qualitative visualization of the latent space by downprojecting a subset of the training set's embeddings to two dimensions using a combination of Principal Component Analysis and t-SNE (Van der Maaten & Hinton, 2008). As

		Collection size		
		8192	32768	131072
H. n.	1	52.10 (3.05)	58.56 (3.37)	56.47 (8.47)
	2	51.40 (0.85)	59.87 (6.34)	55.08 (1.37)

Table 5. Number of hard negatives per sample and "collection" size experiments. The collection is essentially just a second, bigger queue which is used to fetch hard samples from. Notice that every experiment produced bad results. A discussion of this can be found in Section 5.

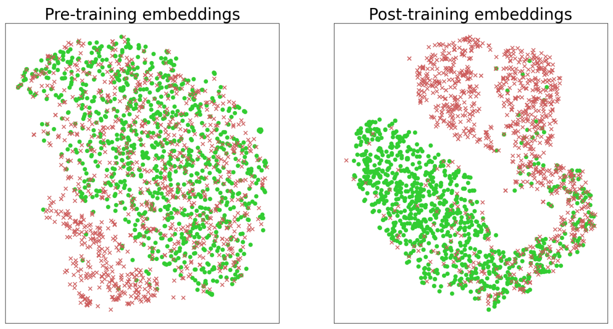

Figure 2. Qualitative visualization of the latent space before and after training. Green dots represent positive-sentiment embeddings and red crosses represent negative-sentiment embeddings

can be seen in Figure 2, the mapping to the latent space differs drastically before and after training. In particular, the different sentiment classes are much more separated, which is precisely what the contrastive learning objective aims to do.

5. Discussion and Conclusion

Our experiments show that the contrastive learning approach achieves a slightly higher accuracy at predicting the sentiment of Twitter data than the two baselines - simple classifiers and BCE BERT. However, by just including the BERT transformer, the accuracy improves significantly and comes up just a little short to our contrastive approach. This shows that the embeddings created by BERT contribute a lot to the improvement of accuracy metric, unlike the manually crafted embeddings used in the classifier baseline. GRUBERT - with its rather simple approach of combining hidden layer representations from BERT with weight sharing GRUs and in doing so attempts to extract more information from the transformer than just its final embeddings - has a similar accuracy score compared to our contrastive approach. There is no reason to believe that GRUBERT is a better or worse model, since it inherently differs on how it tackles the sentiment classification problem. Moreover, with the limited computational resources and time restrictions we decided to conduct our experiments on the smaller 200k dataset, which is also the reason why the numbers ought to be taken with a grain of salt.

That being said, the result of the contrastive learning approach is easier to interpret, since it has the intuitive goal of bringing similar data points closer to each other in latent space. Enhancing it with hard sampling did not yield in a better model, but on the contrary led to a model which performs only slightly better than random guessing. We believe that hard sampling had a poor effect on the inference because of labelling noise in the dataset, as wrongly labeled samples are more likely to be considered hard samples and thus their negative impact on training is amplified.

While contrastive learning does not offer a drastic improvement over GRUBERT, we were nonetheless able to apply it to a problem class which, to the best of our knowledge, it has not been applied to before. For future work, it would be interesting to see how it compares to even more methods, and whether using the full dataset - which we were unable to use due to computational restrictions - changes these results. Moreover, we believe that due to the straightforward adaptability of MoCo and contrastive learning in general, many more problem classes can be approached this way, and we hope to see more works that use it in the future.

References

Anonymous. Contrastive learning of natural language and code representations for semantic code search. 2021. URL https:openreview.net/forum?id=eiAkrltBTh4.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1597–1607. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/chen20j.html.

Dang, N. C., García, M. N. M., and de la Prieta, F. Sentiment analysis based on deep learning: A comparative study. *CoRR*, abs/2006.03541, 2020. URL https://arxiv.org/abs/2006.03541.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

Gao, T., Yao, X., and Chen, D. Simcse: Simple contrastive learning of sentence embeddings. *CoRR*, abs/2104.08821, 2021. URL https://arxiv.org/abs/2104.08821.

Gutmann, M. and Hyvärinen, A. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Teh, Y. W. and Titterington, M. (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 297–304, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/gutmann10a.html.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Horne, L., Matti, M., Pourjafar, P., and Wang, Z. GRUBERT: A GRU-based method to fuse BERT hidden layers for Twitter sentiment analysis. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: Student Research Workshop*, pp. 130–138, Suzhou, China, December 2020. Association for Computational Linguistics. URL https://aclanthology.org/2020.aacl-srw.19.

Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Zhao, T. SMART: robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. *CoRR*, abs/1911.03437, 2019. URL http://arxiv.org/abs/1911.03437.

Kiritchenko, S., Zhu, X., and Mohammad, S. Sentiment analysis of short informal text. *The Journal of Artificial Intelligence Research (JAIR)*, 50, 08 2014. doi: 10.1613/jair.4272.

Mäntylä, M. V., Graziotin, D., and Kuutila, M. The evolution of sentiment analysis - A review of research topics, venues, and top cited papers. *CoRR*, abs/1612.01556, 2016. URL http://arxiv.org/abs/1612.01556.

Nielsen, F. Å. A new ANEW: evaluation of a word list for sentiment analysis in microblogs. *CoRR*, abs/1103.2903, 2011. URL http://arxiv.org/abs/1103.2903.

Park, T., Efros, A. A., Zhang, R., and Zhu, J. Contrastive learning for unpaired image-to-image translation. *CoRR*, abs/2007.15651, 2020. URL https://arxiv.org/abs/2007.15651.

Van der Maaten, L. and Hinton, G. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J. G., Salakhutdinov, R., and Le, Q. V. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237, 2019. URL http://arxiv.org/abs/1906.08237.