Focal Loss for Contrastive Learning

Advanced topics in Deep Learning

Natan Kaminsky

Noam Rotstein

October 2021

Abstract

The application of contrastive learning to self-supervised representation learning has seen an upsurge in recent years, leading to state of the art performance in unsupervised training of deep image models. Focal loss is a loss function that reduces the relative loss for well-classified examples putting more focus on difficult, misclassified examples. In this project, we examine the impact of focal loss on the self-supervised contrastive approach.

1 Contrastive Learning

While supervised learning has enabled great advances in many applications, unsupervised learning has not become as widespread. Contrastive learning seeks to extract useful representations from high-dimensional data. Contrastive losses measure the similarities of pairs of samples in a representation space. Rather than matching an input to a fixed target, the target in contrastive loss formulations can vary during training and be defined in terms of the data representation computed by a network. Contrastive learning can be thought of as training an encoder for a dictionary look-up task. Consider an encoded query q and a set of encoded samples $k_0, k_1, k_2, ...$ that are the keys of a dictionary. Assume that there is a single key in the dictionary that q matches k_+. A contrastive loss is a function whose value is low if q is similar to its positive key k_+ and dissimilar to all other keys (which are negative keys for q). The similarity here is measured by the dot product. The most popular loss for contrastive learning is InfoNCE [5], which is the loss we use in our project:

$$loss(q) = -\log \frac{e^{q \cdot k_+}}{\sum_{k=0}^{K} e^{q \cdot k_i}} \tag{1}$$

Here $q \cdot k_j$ can be divided by a temperature hyperparameter τ. The contrastive loss serves as an unsupervised objective function for training the encoder networks representing the queries and the key.

1.1 Momentum Contrast

In our project, we chose to use the MoCo v2 model [2, 1] (see Fig 1) (with its suggested hyper-parameters). This model considers a query and a key as a positive pair if they are from the same image, and as a negative sample pair otherwise. It takes two random "views" of the same image under random data augmentation to form a positive pair. The queries and keys are respectively encoded by their encoders f_q and f_k. The encoder used is resnet50 [3]. The core of this model is maintaining the dictionary as a queue of data samples. By introducing a queue, the size of the dictionary is decoupled from the size of the mini-batches. The current mini-batch is enqueued to the dictionary and the oldest mini-batch in the queue is removed. A naive solution is to copy the key encoder f_k from the query encoder f_q, ignoring this gradient. But this solution leads to poor results. To solve this problem, a momentum is used. The fc head in resnet is replaced by a 2-layer MLP head. Note that this only influences the unsupervised training stage. The linear classification or transferring stage does not use this MLP head.

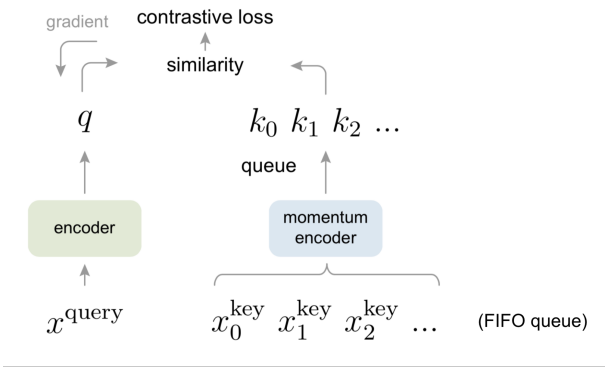

Figure 1: MoCo v2: Encodes the new keys on-the-fly by a momentum-updated encoder, and maintains a queue of keys.

2 Focal Loss

Focal Loss is a modified version of Cross-Entropy Loss that attempts to handle the classification problem by assigning more weight to hard or easily misclassified examples [4]. Focal Loss thus reduces the loss contribution of easy examples and increases the importance of correcting misclassified examples. Easy positive/negative examples are examples that are correctly classified as positive/negative examples. Hard positives/negatives are samples that are incorrectly classified as negative/positive examples. If there are many easy samples, their proportion of loss would be large, even if the model already easily classifies them. This is contrary to what we would want, which is for the model to focus on the more difficult samples that it cannot easily classify. Focal loss puts less emphasis on easy examples and focuses training on the difficult negative examples (see fig 2. The cross entropy loss - CE) takes the estimated probability it wants to maximize and calculates the following loss:

$$L(p) = CE(p) = -\log p \tag{2}$$

The focal loss (FL) takes this estimated probability and calculates the following:

$$L(p) = FL(p) = -(1-p)^{\gamma} \log p \tag{3}$$

There are other modifications of this loss to deal with other class imbalance problems that are not relevant to our case.

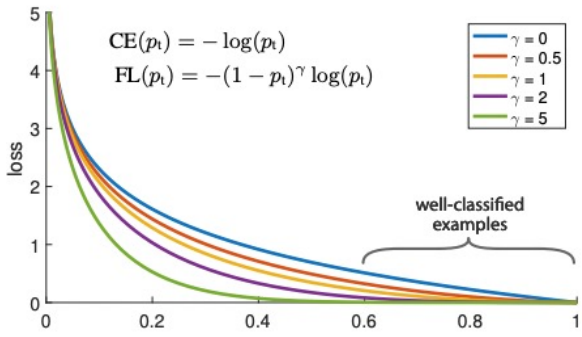

Figure 2: The focal loss down weights easy examples with a factor of $(1-p)^{\gamma}$

3 Experiment Setup

Due to computational limitations, we used the Imagenette dataset. For each experiment (for each γ value), we trained the Resnet model for 1250 epochs in an unsupervised form. Then, we trained the

linear layer using the common protocol of linear classification for 50 epochs. The gamma values tested in each of the experiments were $\gamma = [0, 0.5, 1, 2]$. The loss term in our experiments was converted from the cross-entropy loss used in equation (1) to a focal loss term:

$$loss(q) = -\left(1 - \frac{e^{q \cdot k_+}}{\sum_{k=0}^{K} e^{q \cdot k_i}}\right)^{\gamma} \log \frac{e^{q \cdot k_+}}{\sum_{k=0}^{K} e^{q \cdot k_i}} \tag{4}$$

4 Experiment A- Naive

The first experiment we performed was a naive experiment in which we trained the model from scratch with focal loss using the different γ terms. As can be seen in Table 1, the results were similar for all γ values. The focal loss does not lead to a significant improvement in this case.

γ	Accuracy
0	**81.7**%
0.5	81.4%
1	80.7%
2	80.7%

Table 1: Experiment A- Naive model accuracy for each γ value.

5 Experiment B- Post Training

The second experiment we conducted was designed to show whether a pre-trained model post trained with a focal loss yields different results. We used the model trained with $\gamma = 0$ in Experiment A as our pre-trained model and tried to train it with the different γ values for another 1250 epochs. As can be seen in Table 2, these results also showed no advantage to using focal loss, as the results were similar for all values of γ.

γ	Accuracy
0	84.1%
0.5	**84.3**%
1	84.2%
2	83.9%

Table 2: Experiment B- Post Training model accuracy for each γ value.

6 Loss Distribution

The focal loss efficiency is based on biased loss distribution of many easy examples, and hence a substantial part of the loss values will be concentrated at low values. To test whether this is the case in our task, we used the pre-trained model with $\gamma = 0$ and calculated the loss of the model over each of the samples. This was done exactly as a training epoch, but without backpropagation and with the saved loss for each sample.

6.1 Naive Loss Distribution

As can be seen from the histogram (see Fig. 3), the distribution of losses does not seem to be biased with a strong concentration of low values and therefore the results of experiment A seem logical.

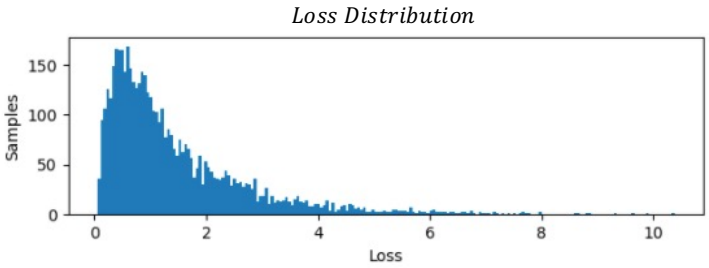

Figure 3: Loss distribution of a $\gamma = 0$ pre-trained model.

6.2 Clustered Loss Distribution

Contrastive learning is based on increasing the similarity in embedded space between the current sample and its augmented version, and decreasing the similarity in embedded space between the sample and all previous samples that are in the queue (in MoCo). However, we speculate this is not the best choice for pre-training a model before training a classification layer. Even if other samples from the queue belong to the same class as the current sample, the contrastive approach still tries to reduce the similarity in the embedded space. Therefore, we thought it would be beneficial to take our pre-trained model ($\gamma = 0$) and use an unsupervised approach (K-means) to estimate which samples from the queue belong to the same class. In this way, we can partially increase the similarity between the current sample and the samples in the queue that belong to the same class, instead of decreasing it. Since the downstream classification task of the Imagenette dataset has ten classes, we used $K = 10$. We performed the loss distribution test in the same configuration as the naive loss distribution test with a modified loss function. We can write all queue keys belonging to the same class as the current sample as $[k_m1, k_m2, ..., k_mn]$. For this approach, we set all these keys as positive and construct the next loss function (for $\gamma = 0$):

$$loss(q) = -\log \frac{e^{q \cdot k_+} + e^{q \cdot k_{m1}} + e^{q \cdot k_{m2}} + ... + e^{q \cdot k_{mn}}}{\sum_{k=0}^{K} e^{q \cdot k_i}} \tag{5}$$

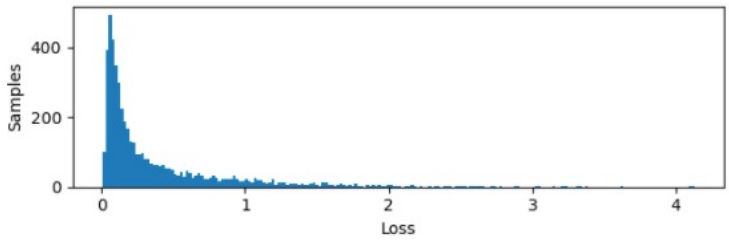

Figure 4: Loss distribution of a $\gamma = 0$ Clustered pre-trained model.

Unlike the naive loss distribution, this time the loss distribution focuses mainly on lower values (see Fig. 4). This is expected since the current positive samples is significantly different from the negative samples.

6.3 Filtered Loss Distribution

In the clustered version, we used all samples in the queue that belong to the same class as positive examples. One can argue that this approach depends too much on unsupervised clustering. Such approach focuses on increasing rather than decreasing the similarity between the current sample and the samples from the same cluster. In the filtered approach, we remove all samples belonging to the same cluster from the queue, leaving only samples from other clusters. This way we can partially avoid decreasing the similarity between them without bringing closer samples that were not clustered correctly. Thus, the modified loss function used for this setup is:

$$loss(q) = -\log \frac{e^{q \cdot k_+}}{\sum_{k=0}^{K} e^{q \cdot k_i} \cdot 1[k \notin k_{mi}]} \tag{6}$$

In this experiment the loss is concentrated more densely in small values (see Fig. 5) than in the clustered loss distribution test. Therefore, additional training with focal losses could be effective.

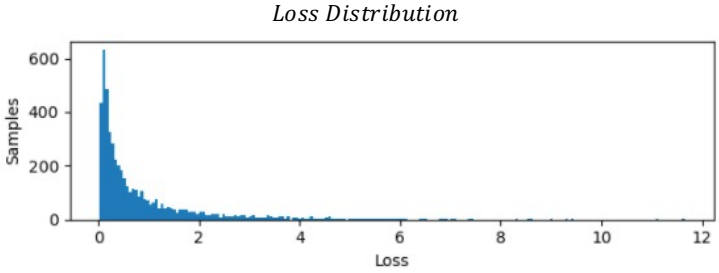

Figure 5: Loss distribution of a filtered $\gamma = 0$ pre-trained model.

7 Experiment C- Clustered

In this experiment we used the same setup as experiment B but changed the post-training loss to be the clustered loss described in equation 5. Unfortunately, the results of this experiment (Table 3) were no better than those of the regular post-training experiments, nor did they show any positive improvement with the use of Focal Loss. We speculate this is due to the strong dependence on the clustering as explained in 6.3.

γ	Accuracy
0	84.0%
0.5	**84.1**%
1	**84.1**%
2	83.8%

Table 3: Experiment C- Clustered model accuracy for each γ value.

8 Experiment D- Filtered

In this experiment we used the same setup as experiment B but changed the post-training loss to be the filtered loss described in equation 6. As we can see in Table 4, the results of this experiment are the best

γ	Accuracy
0	84.2%
0.5	84.9%
1	**85.1%**
2	85.0%

Table 4: Experiment D- Filtered model accuracy for each γ value.

results obtained in all experiments. Post-training with a filtered loss not only increases model accuracy, but also benefits from training with focal losses with increased γ values.

9 Conclusions

In this project we focused on applying focal loss for unsupervised learning. In our experiments we show that naively using focal loss instead of normal classification loss does not improve model accuracy. We

managed to successfully incorporate the focal loss function into the training process by filtering samples in the queue that had the same class as the query. The filtering was done with an unsupervised clustering algorithm on the representations produced by the encoder. The incorporation of the focal loss was done in later stages of training to ensure that the representations of the samples in the queue have good semantic meaning. We can conclude from our experiments that applying focal loss during training improved the performance of our model on downstream tasks, surpassing previous known methods.

10 Future Work

The K values used in the K-means algorithm are based on prior knowledge of the image classes. Further testing could investigate whether this number can be approximated without prior knowledge, or whether other clustering algorithms can be used that do not require a predetermined number of clusters. In this project we focused on utilizing focal loss for contrastive learning by filtering samples from the queue that were of the same class as the query. This could lead to a filtering approach that is more robust and independent of hyper-parameters. However, we noticed that often the augmented query pair do not hold much semantic meaning due to the randomness of applied augmentations. This can cause the model to learn incorrect semantic representations. This problem is further amplified when used with focal loss. We speculate that applying filtering for query pairs with meaningless semantic information would further improve the performance of the model. Another possible venture of research we did not delve into is using different clustering algorithms for filtering the queue.

References

[1] X. Chen, H. Fan, R. Girshick, and K. He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[2] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[4] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[5] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.