Max-MixMatch: Approach to Maximize MixMatch for Semi-Supervised Learning

TEAM 13 : PIRL

20170489
Jaewon Lee
jwlee3746

20183290
Jinyoung Sung
jysung710

20130482
Jaejin Lee
Singed-jj

Abstract

Semi-supervised learning (SSL) is a powerful method to train a model with small amount of labels. In this project, we extend the baseline model by applying state-of-the-art techniques such as pseudo-labeling. Through empirical study, we confirmed the recent studies of SSL can be applied to Naver 200K fashion dataset. We show our results for predicting the classes of Naver Fashion 200K dataset, and analyze our results to help further studies of semi-supervised learning.

1. Introduction

Semi-supervised learning(SSL) is an effective when using data which have large amount of unlabeled data. Common SSL methods try to leverage the labeled data for predicting the label of unlabeled data, without using high-cost human labor.

In this work, we extend the baseline MixMatch model by applying pseudo-labeling, augmentation, and by reinforcing the existing mix up method. Thanks to the support of Naver corporation, we were able to apply our semi-supervised learning method to Naver Fashion 200K dataset, running the model over 370 times during the project period.

We will share our most effective method we have found, which achieved 84.5% in Fashion dataset, by testing numerous variations of baseline model.

In the following section, we introduce our semi-supervised model. In section 4, we show our experimental results and . Finally in section 5 we point out directions of improvement and conclude with summary.

2. Related Works

Recent studies of semi-supervised learning use both supervised loss and unsupervised loss to make a better artificial label and generalization result. These loss includes entropy minimization, consistency regularization and traditional regularization. MixMatch[1] benefits from all of these approaches to make a single loss term.

FixMatch[2] this work by applying strong augmentation and pseudo-labeling.

Highly motivated by these methods, we reinforce mix-up technique MixMatch[1] and study the variations of the baseline model.

3. Methods

We maximize the performance of base model MixMatch[1] by apply following approaches.

3.1. Baseline: MixMatch

The baseline code is applied to labeled and unlabeled samples for each step used in the mix-match. At this time, in order to use it as the target value of the labeled sample, the ratio of the labeled sample is mixed with a ratio greater than 0.5. The target of the unlabeled sample is created through entropy-minimization, and the outputs for the unlabeled sample of the model are MSE with the target value to calculate consistency regularization. At this time, the cross-entropy method is used to compare the output and target of the labeled sample, and the MSE method is used for the unlabeled sample.

3.2. RandAugment to Unlabeled Data

Previous augmentation techniques use learned augmentation policies which has big parameter space. RandAugment selects transformation for each sample from the transformation pool. It uses uniform probability between techniques and magnitude parameters. However the training parameters follow similar schedule during training and postulate that a single global distortion M may suffice for parameterizing all transformations.

Instead of sampling a random magnitude from a range, shown in FixMatch[2], we globally fixed the magnitude to the strongest. Our strong augmentation setting is fixed to 2 random transformations and magnitude of 10, which is the strongest.

3.3. Reinforced MixUp

MixUp is a generic regularization method which encourages the model to have strictly linear behavior between examples.

To increase the speed of convergence, we reinforce MixUp by applying strong augmentation not only for unlabeled data, but also for the labeled data. It was also expected that strong augmentation on labeled data would bring more variations of the dataset, resulting in better score. The MixUp process is also an augmentation process, so the greater the difference between data and label, the greater the learning performance.

3.4. Pseudo-labeling

During the project, we found that the recent entropy minimization technique has less effect on the Naver Fashion 200K dataset. Instead of using sharpening, which lowers the entropy of unlabeled from weak augmentation, we decided to include pseudo-labeling as a part of our model.

Our motivation behind this approach is that we thought that the early part of the learning would be interfered by the incorrect labels. Masking the pseudo label of weakly augmented image allows you to use only the data of high confidence a was expected to filter the poorly generated labels.

The threshold of masking was set to 0.95, and an unlabeled sample output with 95% confidence was used. Masked sample only survived during training epochs.

epoch	masked samples
50	0.32
100	0.38
150	0.46
200	0.54

Table 1. Masked sample survived during training epochs. It is percentage of maksed unlabeled data used to calculate loss beyond threshold (0.95) at Lambda_u 75.

3.5. Consistency Regularization

Main idea of consistency regularization is that the model would output same label when fed unlabeled data and the perturbed one.

We apply consistency regularization to maintain consistency among weakly augmented and strong augmented data. We set the threshold to 0.95 to decide whether the data would be included or not.

$$Loss = L_x + lambda_u * L_u$$
$$L_x = crossEntropy(pred_x, label_x)$$
$$L_u = crossEntropy(pred_u, label_u) * mask$$

Figure 1. Our loss term. Lx for labeled data, Lu for unlabeled data.

Our loss term uses cross-entropy loss for both labeled data and unlabeled data, and mask is applied for consistency regularization. Confidence threshold is set to 0.95 in our setting.

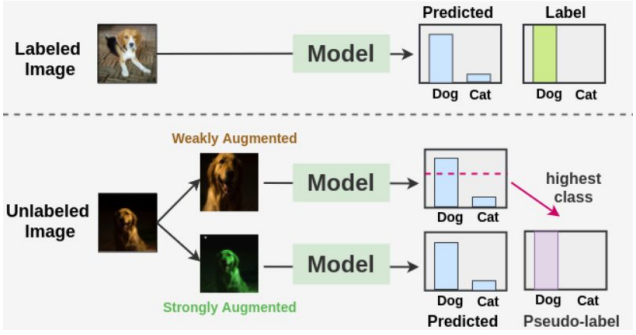

Figure 2. Our data augmentation pipeline

Figure 2 shows our data augmentation pipeline. Pseudo-labeling is applied on unlabeled image, so that it can generate more augmented images.

4. Experiments

Here, we present the results of using two networks for Naver fashion datas. Naver fashion data consists of the following. labeled data: 16k, validation data: 4k, unlabeled data: 40k, fashion images with 265 classes. Commonly used were 48 batch size, 200 epoch for each batch, and exponential decay of learning-rate.

4.1. Implementation details

In all experiments we use the "ResNet-18" and "EfficientNet" model.

Hyper parameter	Value
epochs	200
batchsize	48
learning-rate	1e-3
momentum	0.9
lambda-u	75
T	0.5
threshold	0.95

Table 2. Hyper parameters and values

Our implementation of the model and training procedure closely matches that of MixMatch, except for the following differences: First, we use batch size of 48; relatively small because of NSML memory circumstances. Second, we apply Efficient Net model pre-trained by efficientnet-b2.

4.2. Results

Here, we present the results of using two networks: ResNet-18 Net and EfficientNet for Naver fashion datas.

Res-18 Net

	Best Top1 Acc (Val.)	Best Top5 Acc (Val.)
Baseline	63.55	83.01
+ RandAugment	64.56	83.32
+ Reinforced mix-up	63.44	83.69
+ Pseudo-label&Mask	64.69	83.52

Table 3. Table of top-1, top-5 accuracy that appears for each application of each method, starting with Baseline, in a Resnet-18 environment.

Each method was gradually applied from the baseline. Table 3 shows that the improvement of top-1 and top-5 accuracy could be confirmed with each method applied from the top one from above.

Efficient Net

	Best Top1 Acc (Val.)	Best Top5 Acc (Val.)
Only with RandAugment (efficient-b0)	68.19	85.2
Latest (efficient-b0)	70.12	86.76
Latest (efficient-b2)	72.26	87.25

Table 4. Table of top-1, top-5 accuracy that appears for each application of each method, starting with Baseline, in a Efficient Net environment.

Table 4 shows that the improvement of top-1 and top-5 accuracy could be confirmed with each method applied from the top one from above. First one is only RandAugment methods applied version with pre-trained efficient-b0 model. Second and Last one is the version of all above-mentioned methods applied. Second is latest one with pre-trained efficient-b0, and the Last one is with pre-trained efficient-b2.

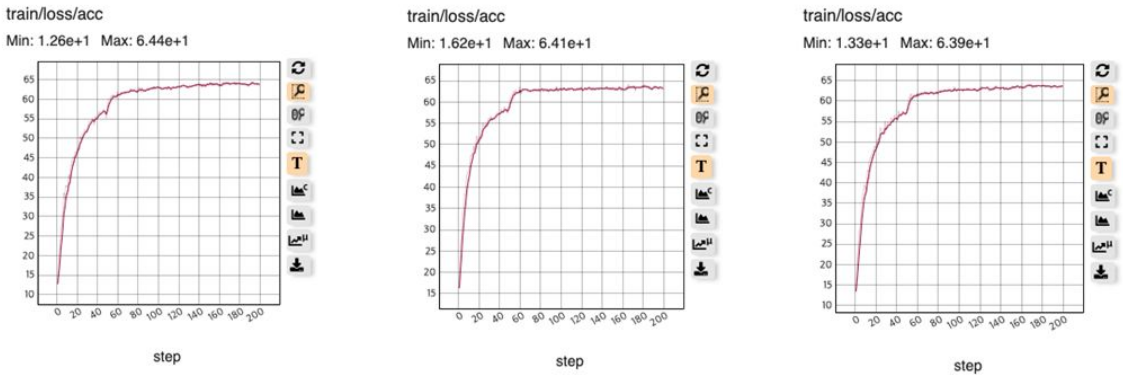

Figure 3. Graph or Top-1 accuracy. Each graph is for (a): Baseline + RandAugment, (b): (a) + Reinforced MixUp, (c): (b) + Pseudo Labeling and Mask

5. Conclusion

We have applied several methods: RandAugment, Reinforced MixUp, and Pseudo-Labeling to maximize the performance of MixMatch. Significant performance improvements were derived through the application of the methods, various parameters and changes in the model environment. Further studies expect that fine changes in parameters, environmental settings will further boost performance.

References

[1] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In Advances in Neural Information Processing Systems 32. 2019. 1, 2, 3, 5, 6, 7

[2] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Han Zhang, Colin Raffel. FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence.

[3] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In ICML Workshop on Challenges in Representation Learning, 2013. 1, 3, 4, 5, 6

[4] Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space. arXiv preprint arXiv:1909.13719, 2019. 1, 3, 6, 7, 12, 13, 14