ENSTA

Rapport

Computer Vision Semi Supervised Learning Project Report

Élèves :
Nathan Gaubil

Encadrants :
Gianni Franchi (ENSTA)

1 Introduction

The aim of the project is to train a CNN on CIFAR 10 using 250 annotated images. The other images can also be used without annotations, which is known as semi-supervised learning. A Wide ResNet-28-2 must be trained from scratch. The goal is to ensure that the CNN performs well on the CIFAR 10 test set and does not overfit too much.

The lack of labeled data is a major obstacle in the application of deep learning to practical fields such as medical imaging, which can result in poor model performance. To overcome this challenge, it is possible to leverage unlabeled images using a semi-supervised learning approach called FixMatch. FixMatch is a simpler method than previous approaches such as UDA and ReMixMatch. The article also explains how FixMatch has improved the state-of-the-art in semi-supervised learning with a median accuracy of 78 percent and a maximum accuracy of 84 percent on CIFAR-10 using only 10 labeled images.

Consistency regularization is an important element of advanced semi-supervised algorithms. It uses unlabeled data by assuming that the model should produce similar predictions when fed with perturbed versions of the same image. The idea has been popularized by several research works and is implemented through a loss function that combines a supervised classification loss and a loss on unlabeled data. Pseudo-labeling involves using the model itself to obtain artificial labels for unlabeled data. It uses hard labels and only keeps those whose largest class probability is greater than a predefined threshold. These two methods are often used together to improve the performance of semi-supervised models.

2 Pipeline

We will study data augmentation.

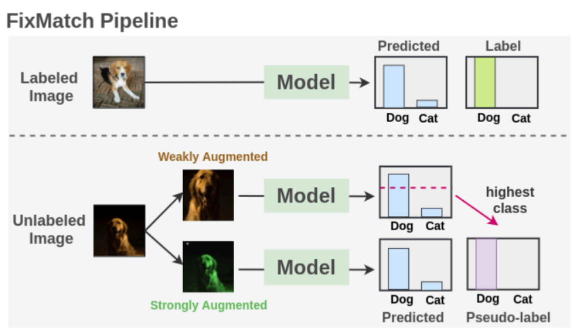

FIGURE 1 – Overview of FixMatch

2.1 Data Augmentation

FixMatch uses two types of augmentations : "weak" and "strong". Weak augmentations consist of a standard strategy of flipping and random shifting of images. For strong augmentations, two methods based on RandAugment are experimented, followed by Cutout. RandAugment is a regularization technique for machine learning that increases training data by applying random transformations to images. This method involves selecting a set of random image transformations from a library of predefined transformations, such as rotation, zoom, or distortion, and applying these transformations to the original image randomly with varying intensity. More concretely in the code, n augmentations are chosen among the 14. The magnitude value of M is a scalar between 0 and a predefined maximum value, and determines to what extent the transformation is applied to the original image. A smaller value of M gives a smoother transformation, while a larger value of M gives a more significant transformation.

2.2 Model

WideResNet 28 2 is a widely used convolutional neural network (CNN) model for image classification. It is an improved version of the ResNet model, which uses wider layers to enhance the network's representation capacity. WideResNet 28 2 has 28 layers of neurons and a widening factor of 2, which means that each layer has twice as many filters as in a standard ResNet. This increase in network width significantly improves its classification accuracy, especially for complex datasets and high-resolution images.

2.3 Training the Model

In FixMatch, mini-batches are composed of a mixture of labeled and unlabeled data. Labeled data are the images for which labels are known, while unlabeled data are those for which labels are unknown. Unlabeled data are multiplied by a factor of MU (e.g., MU=7 for CIFAR-10) to increase the number of unlabeled samples in each mini-batch. This is an important hyperparameter.

The supervised part of FixMatch uses a cross-entropy loss H() for the classification task on labeled images. The total loss for a mini-batch is defined as Ls and is computed by taking the average cross-entropy loss for each image in the mini-batch. The formula for the supervised loss is :

$$l_s = \frac{1}{B}\sum_{b=1}^{B} H(p_b, p_m(y|\alpha(x_b))) \tag{1}$$

where B is the mini-batch size, p_b is the class prediction for image b, $p_m(y|\alpha(x_b))$ is the model prediction for the perturbed image, and α is the perturbation function for unlabeled images.

For unlabeled images, "weak" augmentations are first applied to obtain a perturbed version of the image, and then the most probable class is determined by applying the argmax function. This predicted class is considered the associated artificial label (or pseudo-label) for that image, and is compared to the model predictions for a heavily perturbed version of the unlabeled image. The formula for the model prediction on a perturbed image is :

$$q_b = p_m(y|\alpha(u_b)) \tag{2}$$

and the formula for the pseudo-label is :

$$\hat{q}_b = argmax(q_b) \tag{3}$$

We finally combine these two losses to obtain a total loss that we optimize to improve our model. λ_u is a fixed scalar hyperparameter that decides the relative contribution of losses from unlabeled images compared to losses from labeled images.

The formula for the total loss is :

$$loss = l_s + \lambda_u l_u \tag{4}$$

An interesting result arises from λ_u. Previous work has shown that increasing the weight during training is beneficial. But in FixMatch, this is built into the algorithm itself. At the beginning of training, the model does not trust the labeled data, so its output predictions on unlabeled data will be below the threshold. Thus, the model will only be trained on the labeled data. But as training progresses, the model becomes more confident in the labeled data and the predictions on the unlabeled data also begin to exceed the threshold. Therefore, the loss will soon start to integrate predictions on unlabeled images.

2.4 Generalization

To allow for model generalization and prevent overfitting, certain techniques are implemented. First, regularization is particularly important. In the official implementation, they use a simple weight decay regularization. Using standard SGD with momentum is more performant. They found no substantial difference between standard momentum and Nesterov momentum. For a learning rate scheduler, I use a cosine learning rate decay that sets the learning rate to $\eta cos(7\pi k/16K)$, where η is the initial learning rate, k is the current training step, and K is the total number of training steps. Finally, we report final performance using an exponential moving average (EMA) of the model parameters.

EMA is used to smooth out model parameters by reducing noise and preventing overfitting. It also allows for adjusting the speed at which old data is forgotten and new data is integrated.

The calculation of EMA involves giving an exponentially decreasing weight to previous data and giving a higher weight to recent data. Specifically, EMA for a time instant t is computed from the current value x_t and the previous EMA value for a previous time instant $t-1$. The parameter alpha is used to control the speed at which EMA reacts to new data and is typically set between 0 and 1.

The hyperparameters are in the appendices.

3 Conclusion

In its presentation, FixMatch was a simpler SSL algorithm that had achieved state-of-the-art results on many datasets. For training, only standard cross-entropy losses were used on labeled and unlabeled data, and the objective could be written in just a few lines of code.

It was important to take into account weight decay and optimizer choice to get the best possible results with FixMatch. Indeed, these factors play a crucial role in the algorithm's effectiveness, even when the model architecture is controlled.

The results are in the appendices.

4 Annexe

4.1 Train

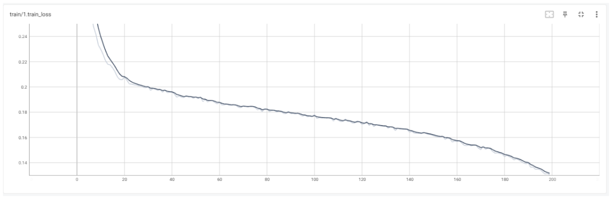

FIGURE 2 – Training loss

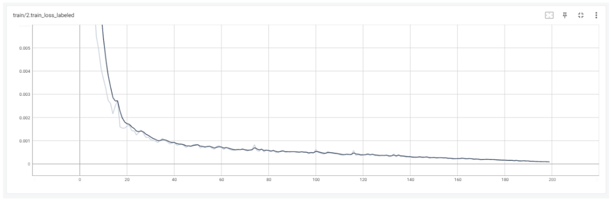

FIGURE 3 – Training loss label images

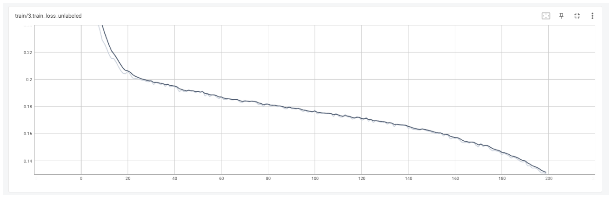

FIGURE 4 – Training loss unlabeled images

On observe que l'entrainement est plutot stable, les pertes décroit de manière uniforme. J'ai décié de n'utiliser que 200 epochs par rapport à 1024 du papier car l'entrainement est trop long et j'obtiens déja un score largement suffisant pour montrer que ça fonctionne.

4.2 Test

4.3 Model Parameters

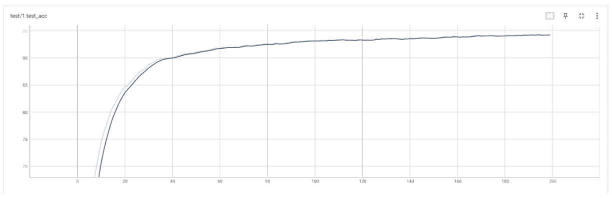

FIGURE 5 – Testing accuracy

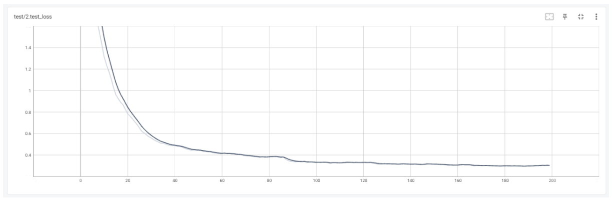

FIGURE 6 – Testing loss

Variable	Valeur
BATCH_SIZE	64
EVAL_STEPS	1024
LEARNING_RATE	0.03
NUM_EPOCHS	200
LAMBDA_U	1
WEIGHT_DECAY	5e-4
MU	7
TEMPERATURE	1
THRESHOLD	0.95

TABLE 1 – Values of the variables used in the model training