Proposed Changes in Semi-Supervised Learning by Augmented Distribution Alignment

Tejas Ahuja
GitHub Repository

November 6, 2024

1 Introduction

To enhance model performance, semi-supervised learning (SSL) makes use of a sizable pool of unlabelled data and a small amount of labelled data. Adversarial distribution alignment aids in minimising the discrepancy between labelled and pseudo-labeled data in Augmented Distribution Alignment (ADA-Net). However, sample selection bias and noisy pseudo-labels may reduce the efficacy of this method. In order to improve model resilience, I am proposing **adaptive sample selection** approach that dynamically concentrates on examples that are most advantageous for alignment. In my second enhancement, a CNN model is optimised to increase performance without sacrificing accuracy.

2 Adaptive Sample Selection Approach

My approach prioritizes pseudo-labeled samples based on their alignment utility, focusing on low-confidence samples that are likely near decision boundaries or show distribution gaps.

2.1 Adaptive Selection Strategy

Let $f(x_u)$ be the model's output for sample x_u. We are given a batch of them and We compute the confidence of each pseudo-label using the entropy of $f(x_u)$:

$$\text{Entropy}(f(x_u)) = -\sum_c f(x_u)^{(c)} \log f(x_u)^{(c)},$$

where $f(x_u)^{(c)}$ is the model's predicted probability for class c. Samples with high entropy are prioritized for adaptive alignment.

For each low-confidence sample, we apply stronger augmentation and assign a higher weight in the loss function. Let λ_i be the weight assigned to sample $x_u^{(i)}$, defined as:

$$\lambda_i = \alpha \cdot \exp(-\text{Entropy}(f(x_u^{(i)}))),$$

where α is a scaling parameter.

2.2 Implementation

1. For each pseudo-labeled sample x_u, compute the confidence score based on the entropy of $f(x_u)$.
2. Apply stronger augmentation to low-confidence samples
3. During adversarial training, apply higher gradient reverse weights to low-confidence samples, emphasizing alignment for samples near decision boundaries.

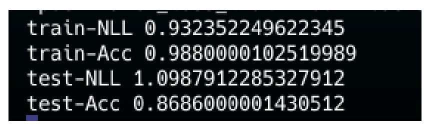
2.3 Results

The updated model weights have been provided in the Google Drive

Dataset	**Previous Accuracy**	**Current Accuracy**
CIFAR-10 (Test Seed 1)	87.24%	87.99%
CIFAR-10 (Test Seed 2)	87.23%	87.93%
CIFAR-10 (Test Seed 3)	87.18%	87.96%

Training Negative Log Likelihood reduced from 1.0817 to 0.9323


train-NLL 1.0817737913131713
train-Acc 0.9348000037670136
test-NLL 1.0830386173725128
test-Acc 0.8912000072002411


train-NLL 0.932352249622345
train-Acc 0.9880000102519989
test-NLL 1.0987912285327912
test-Acc 0.8686000001430512

3 Modifications in CNN Network

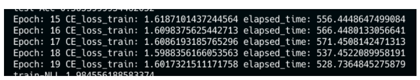
- **Depthwise Separable Convolutions:**
  Depthwise separable convolutions, which divided the convolution process into depthwise and pointwise phases, were used in place of the conventional convolutional layers (layers $c3$, $c4$, and $c5$). By applying a filter to each channel and then combining the channel outputs using a 1x1 convolution, this minimises computation.
- **Reduced Channel Sizes:**
  Initial channels are reduced, starting with $f_{out} = 64$ filters instead of 128, progressively increasing in later layers. This optimization minimizes computation early in the network while retaining necessary depth in later layers for complex feature extraction.
- **Removed Dropout in Convolutional Layers:**
  Convolutional layer dropout was eliminated since batch normalisation

(BN) is adequate for regularisation. Fully connected layers preserve model generalisation while lowering computing cost by retaining dropout.

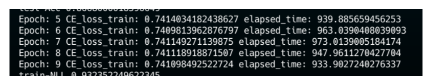
- **Global Average Pooling (GAP):**
  Replaced flattening in the final convolutional layer with GAP using $h = \text{tf.reduce\_mean}(h, \text{axis} = [1, 2])$. This reduces the number of parameters and enforces spatial invariance.

3.1 Results

Significant reduction in training time was observed. Training time reduced from an initial 953 seconds per epoch on the CC server to only 550 seconds per epoch on average resulting in a **42.28 % reduction in training time per epoch**.



```
Epoch: 15 CE_loss_train: 1.6187101437244564 elapsed_time: 556.4448647499084
Epoch: 16 CE_loss_train: 1.6098375625442713 elapsed_time: 566.4480133056641
Epoch: 17 CE_loss_train: 1.6086193185765296 elapsed_time: 571.4508142471313
Epoch: 18 CE_loss_train: 1.5988356166053563 elapsed_time: 537.4522089958191
Epoch: 19 CE_loss_train: 1.6017321511171758 elapsed_time: 528.7364845275879
```



```
Epoch: 5 CE_loss_train: 0.7414034182438627 elapsed_time: 939.885659456253
Epoch: 6 CE_loss_train: 0.7409813962876797 elapsed_time: 963.0390408039093
Epoch: 7 CE_loss_train: 0.741149271139875 elapsed_time: 973.0139005184174
Epoch: 8 CE_loss_train: 0.741118918871507 elapsed_time: 947.9611270427704
Epoch: 9 CE_loss_train: 0.741098492522724 elapsed_time: 933.9027240276337
```

The model was trained for 40 Epochs and was under-performing from the original model for the number of epochs it was trained for. The model weights have been provided in the Google Drive. The training and inference was much faster for this model.

Number of Epochs Trained	Previous Model	Updated Model
20	55.54%	51.69%
40	66.31%	63.46%

4 References

Wang, Qin, Li, Wen, and Van Gool, Luc. "Semi-Supervised Learning by Augmented Distribution Alignment." *arXiv preprint arXiv:1905.08171* (2019).