Teacher-ensemble distillation based learned Label Interpolation for SSL

A BS thesis end-semester report submitted in fulfillment of the requirements for the degree of

Bachelor of Science
in
Data Science and Engineering
by

Manas Dubey
Roll No.: 19184

Under the Supervision of
Dr. Akshay Agarwal

Department of Data Science and Engineering
Indian Institute of Science Education and Research Bhopal
Bhopal - 462066, India

22nd April, 2023

Abstract

Semi-supervised learning is important because it allows us to leverage the vast amounts of unlabeled data that are available in many real-world applications. In many cases, acquiring labeled data can be time-consuming and expensive, but there may be large amounts of unlabeled data available that could be used to improve the accuracy of machine learning models.Overall, semi-supervised learning is an important area of research because it allows us to make more effective use of the data that is available to us, which can lead to more accurate and robust machine learning models in many real-world applications. In this project I propose a novel strategy for the training of models with a dearth of labeled data and an excess of unlabeled data using the Pseudo-Labeling method for semi-supervised learning. Two teacher models are trained on the limited labeled data available and their predicted pseudo-labels on the unlabeled data guide the learning of a student model which is trained with both unlabeled and labeled data by employing a Mixup based augmentation routine.By increasing the diversity of the input data, it is possible to reduce overfitting and improve the generalization of the student model, leading to better performance on both the labeled and unlabeled data. The performance is evaluated on benchmark datasets such as CIFAR-10, CIFAR-100 and SVHN with an assortment of lightweight and heavy models with sufficient ablation studies.

Contents

Chapter 1

Introduction

Pseudo-labeling has been an efficient and relatively successful way of training deep learning learning models with less supervised data and more unsupervised data owing to its simplicity of application. First proposed by Lee[1] in 2013, the technique itself follows four simple steps (i) train a model on a batch of labeled data (ii) Use the trained model to predict labels on a batch of unlabeled data which now act as the pseudo-labels(iii) take the supervised loss on the unlabeled data with the pseudo-labels and lastly, combine the supervised loss and unsupervised loss on the basis of a training procedure which combines both the labeled and pseudo-labeled data. At its core is the aim of Semi-Supervised learning which is to use both labeled and unlabeled data to learn from. In order to be useful, these methods should improve upon the predictive accuracy attained by supervised methods, and have been shown to do so under certain conditions. This potential has been recognized and a lot of research has been done in this area.

The main problem with the method is that the model which eventually learns from these pseudo-labels might end up learning wrong information. As a result the performance of the model might deteriorate on the test set. This problem is called confirmation bias[2].It thereby increases the models confidence in inaccurate predictions over time.In order to tackle that I use a consensus based mechanism where instead of using assigning the pseudo label as that predicted by one single model, there should be agreement between multiple models on the label (in my case 2 CNN's). The data instances where the theres no consensus ,mixup augmentation is used. Mixup [3] is a simple and data-agnostic data augmentation routine that trains a DNN on convex combinations of pairs of examples and their labels.By adding the prior knowledge that linear interpolations of feature vectors should result in linear interpolations of the related targets, mixup broadens the training distribution. This regularises the DNN (while it is being trained) to favour linear behaviour in between training samples.Minimal

amount of compute overhead is introduced during the simple implementation of Mixup training.

$$\tilde{x} = \lambda x_i + (1 - \lambda) x_j,$$
$$\tilde{y} = \lambda y_i + (1 - \lambda) y_j,$$

Augmented data that mismatch the statistics of the training set are not likely to match the statistics of the test set, leading to less effective augmentation. From the perspective of training, using augmented data that deviate too much from the training set statistics will also cause high training error on the original training set, and in turn hurt generalization.The augmented data should be as diverse as feasible to completely fill the training set's coverage area. It is sufficient to increase the training set by interpolating the nearest neighbours in the scenario where the data distribution is a low dimensional manifold . On the other hand, in the case in typical vision and speech classification tasks, nearest neighbours do not provide enough information to reconstruct the geometry of the data distribution, and other training samples can provide a significant amount of additional geometric information. Therefore, compared with interpolating nearest neighbors, interpolating random pairs of training data provides a better coverage of the data distribution.

Chapter 2

Background

Labeling data is particularly laborious, time-consuming, and may call for domain knowledge in many NLP tasks, such as webpage categorization, audio analysis, or named-entity recognition; or in less common machine learning applications, such as protein sequence classification. Therefore ,a SSL technique like pseudo-labeling can come into picture whenever there's easy access to data collection and it is shown that blending unlabeled and labeled datasets can boost model performance.SSL algorithms use unlabeled data to learn additional structure about the input distribution. The existence of cluster structures in the input distribution could hint the separation of samples into different labels. This is often called the cluster assumption: if two samples belong to the same cluster in the input distribution, then they are likely to belong to the same class.[4] The cluster assumption is equivalent to the low density separation assumption: the decision boundary should lie in the low-density regions. The equivalence is easy to infer: A decision boundary which lies in a high-density region, will cut a cluster into two different classes, requiring that samples from different classes lie in the same cluster; which is the violation of the cluster assumption. This assumption has inspired many consistency-regularization based SSL techniques, including the -model (Laine Aila, 2017; Sajjadi et al., 2016), temporal ensembling (Laine Aila, 2017), VAT (Miyato et al., 2018), and the Mean-Teacher (Tarvainen Valpola, 2017).

Prior studies on deep SSL vary in terms of whether they learn from the unlabeled set via consistency regularisation or pseudo-labeling, but they all utilise a cross-entropy loss (or a comparable technique) on the labelled data.Consistency regularisation assumes that the same sample must yield the same results under various perturbations. This concept was applied before where they forced softmax predictions to be comparable while applying randomised data augmentation, dropout, and random max-pooling. Similar reasoning is used in other research avenues, which also extends the perturbation to other epochs; specifically, the

current prediction for a sample must be comparable to an ensemble of historical predictions for the same sample. Co-training [6] uses two (or more) networks trained simultaneously to agree on their predictions (consistency regularization) and disagree on their errors. Errors are defined as different predictions when exposed to adversarial attacks, thus forcing different networks to learn complementary representations for the same samples.

Chapter 3

Research Gaps

The Mix-up training method has been empirically shown to have better generalization and robustness against attacks with adversarial examples than the traditional training method.One major enigma that has always surrounded this field is the lack interpretatbility. It can be difficult to understand how the models are making their predictions, which can be a problem in applications where interpretability is important. With Grad-CAM visualizations[7], which i reproduce , it is possible to see which regions of an input image are most important for the model's classification decision. This can help to provide insight into why the model is making certain decisions and can help to identify potential weaknesses or biases in the model. Another potential research gap is the robustness of these SSL model to adversarial attacks. It is unclear how well models trained using these techniques will perform in the presence of adversarial examples or other forms of attacks.

I employ soft and strong augmentations to the training data which can lead to more robust models because they test the strength of the classifier under various forms of perturbations. More research is needed to understand the potential for bias in these techniques and to develop strategies for mitigating it. Addressing more pragmatic concerns such as that of generalization, it is often unclear how well these techniques generalize to different datasets and problem domains. More research is needed to understand the extent to which they can be applied to different types of data and problems. Regularization strategies have been widely employed in regards to this. Augmentation strategies such as Mixup and masked autoencoders[8] can be effective techniques for regularizing machine learning models.

One could feed the output of the teacher models into a separate neural network that learns the hyperparameter alpha for Mix-up. This approach is known as a meta-learning approach, where a separate model learns to adapt to the specific characteristics of the data or task.To implement this approach, you could use a small neural network as a meta-learner that takes in the softmax probabilities produced by the teacher models and outputs the hyperparameter alpha. This network could consist of one or more fully connected layers followed by a sigmoid activation function to ensure that the output is between 0 and 1.During training, one would first train the teacher models on the labeled data and use them to generate the pseudo-labels for the unlabeled data using Mix-up. You would then feed these pseudo-labeled data along with the original labeled data to the student model and train it using the Mix-up loss that includes the hyperparameter alpha predicted by the meta-learner.By incorporating the meta-learner into your semi-supervised learning framework, you can potentially learn the optimal value of alpha that is specific to your data and task, which can lead to better performance.

Chapter 4

Objectives

The first part of my thesis focused on a novel yet simple strategy that used teacher-student pseudo labeling training strategy in order to tackle semi-supervised learning. The other half tried to address a more important problem ,one that has been studied through various angles.

(1)To propose a novel SSL framework that utilises the consensus of two teacher CNN models for pseudo label generation and utilises Mix-up where there is dissent between the teacher models.

(2)Report and analyse the performance of the student model which has been trained according to a training routine which amalgamates the labeled and pseudo labeled data. The experiments performed will be on benchmark datasets such as CIFAR-10, CIFAR-100 [9], SVHN [10] and MNIST datasets [11].

(3) Implement a "learn your own teacher" strategy where a multi-headed network manages to not only proceed to minimize the expected loss on psuedo-labels and labeled data but also learns the Mix-up parameter "lambda" ,which is done by the second head of the same model, where the network meta-learns which teacher should it tend towards.

(4) Provide an analysis of model by comparing with various baselines using different combinations of neural networks and render explanatory plots such as t-SNE[12], Grad-CAM's etc.

Chapter 5

Methodology

5.0.1 Teacher Ensemble distillation through pseudo-labeling and Mixup

The proposed framework is as follows:-
(1) Initially, two Teacher models are initialised which in my case were VGG 13 and VGG-11 [13] which are trained on the labeled data which is less in quantity.

(2) The trained weights are stored in path files and accessed during the training of student. The student model chosen was ResNet-9[14] and MobileNet v2 [15].

(3) A batch of data is sampled from the un-labeled dataset and the output logits are taken from both the teacher models.

(4) The hard pseudo labels are evaluated for the logits and if there's agreement between the predicted output class ,the data-pseudo label pair is sent for training to the student model using the cross entropy loss .

(5) For the data instances in unlabeled set where there's disagreement, label interpolation using the Mix-up scheme is implemented to obtain the pseudo label. The interpolated pseudo-label and data pair are sent for training to the student model.

Since GPU's allow for faster batch matrix multiplications, i separate an incoming unlabeled batch into agreement and disagreement batches(by the teachers) during training. The total loss for the student model is a combination of a normal loss and a mixup loss with the pseudo labels and is as follows :-

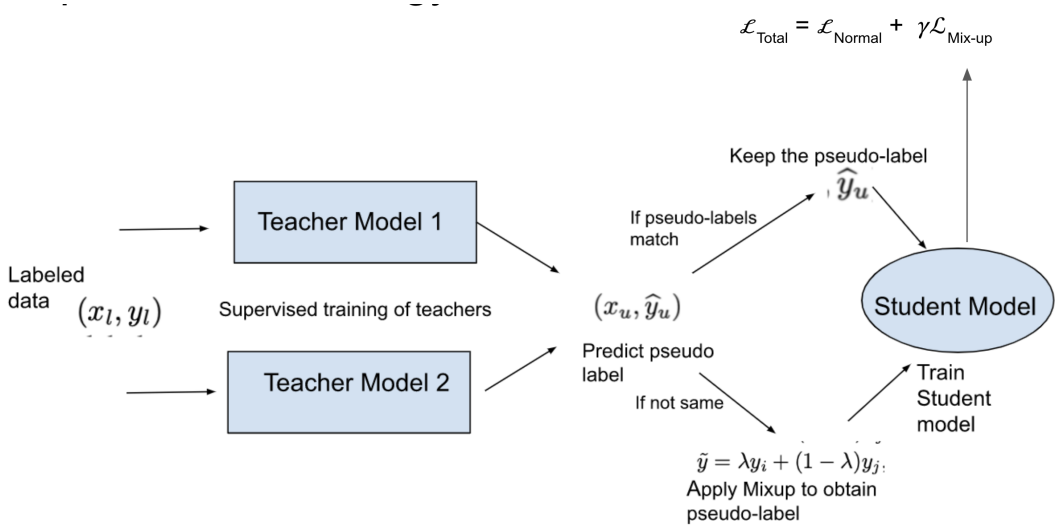
Here a Cross entropy loss is used between the output of the model and pseudo-labels. Gamma is an introduced hyperparameter which is there to control the

$$\mathcal{L}_{\text{Total}} = \mathcal{L}_{\text{Normal}} + \gamma \mathcal{L}_{\text{Mix-up}}$$

Figure 5.1: Mixup and labeled loss

contribution of mixup loss. It is set to 1 for my experiments.

For CIFAR-10 , 4,000 labeled examples are kept as labeled data while 41,000 examples are used as unlabeled data. The test set for CIFAR-10 is standard and consists of 10,000 examples. For SVHN [46], 4,000 examples are used as labeled data whereas about 73,257 examples are used as unlabeled data. The test set for SVHN is also standard, and has 26,032 examples. The batch size is set to 64 while training the teachers and 128 while training the student. I have used a held out validation split from those 4000 labeled examples of 1000 images and the training for the teacher models is done with Adam optimizer[10] and the student with the Stochastic Gradient Descent optimizer.



All my experimentations were done on Google Colab standard GPU, on Pytorch 2.0 . Due to the cumbersome nature of the models I only trained the teacher models from scratch for 30 epochs with an Early stopping strategy. In order to tackle the class imbalance problem, i train the teachers with data augmentation as an upsampling measure for the minority classes coupled with early stopping to reduce overfitting.For the student model, i first train the model for 20 epochs on the labeled data and then for another 20 epochs on the pseudo-labeled unsupervised data. For rest of the data set the number of epochs are set

to 30-40 epochs based on validation curve trends.

The learning rate is kept as 0.1 for all models and during the training of student, it is gradually decreased after every 10 epochs. Also, so that the student does not forget the cluster structure it has learned on the labeled data , i use a revision based training strategy where after every 10 epochs i train the student on labeled data for 1 epoch. The parameter alpha for the mixup augmentation is set to 1 for Cifar10 and Cifar100 which leads to interpolations through lambda being uniformly distributed between 0 and 1. Furthermore, i also implement RandAugment[17] based augmentation for Unsupervised Data Augmentation[18] as consistency regularization to test convergence. For the augmentation strategies in RandAugment I use the same as that used in Meta Psuedo Labels[6] for SVHN and Cifar 10/100 data.

Masked autoencoders (MAEs)[8] are neural networks that are trained to reconstruct a partially observed input. In MAEs, a subset of the input features is randomly selected and replaced with a mask. The task of the network is to predict the missing values based on the remaining observed features.MAEs can be useful as teacher models for semi-supervised learning because they are trained to capture high-level features of the data that are relevant for predicting the missing values. These features can be used to generate pseudo-labels for unlabeled data, which can then be used to train student models. Therefore, i have also trained teacher models on Cifar-10 with MAE's.The masking mechanism used in MAEs can also be beneficial for semi-supervised learning because it forces the network to learn more robust and generalizable representations of the input. By predicting missing values from only a subset of the input features, the network is forced to learn more abstract and higher-level representations of the input, which can be useful for transfer learning and improving performance on downstream tasks.

Mixing the label space using the Mixup technique can improve the generalization performance of a neural network. By combining two samples and their corresponding labels, Mixup encourages the model to learn linear relationships between input features and output labels, which can help it to better generalize to unseen data[3].The regularization effect of Mixup can also help to reduce overfitting and improve the model's ability to generalize to different data distributions. This is particularly useful in situations where the labeled data is limited or noisy, and the model needs to learn from unlabeled data as well.

I have also considered filtering out high entropy predictions by the teacher models where the average entropy for predicted distribution over all the classes for a particular data instance is above a certain threshold [23]. There exist appropriate binary masking functions in Pytorch which can achieve this with minimal computation and can improve the voting mechanism in the model about

a pseudo-label.

5.0.2 Learn your own teacher: Data Driven Label interpolation scheme

Mixing up the labels means that the model is forced to learn a more generalized representation of the data, as it has to consider the relationships between different classes in the dataset. This can improve the model's performance on unseen data and increase its robustness to label noise or errors. By minimizing the cross-entropy loss of the mixed-up labels, the model is encouraged to learn more discriminative features that can better differentiate between the different classes in the dataset.

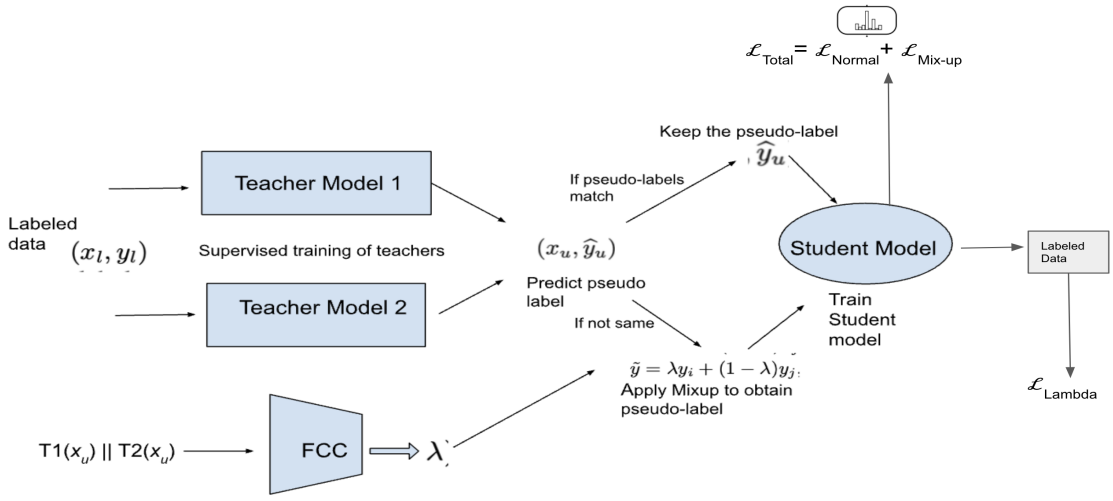
However, it's worth noting that mixing up the labels should be used with caution, as it may not always improve performance and could even hurt performance in some cases, particularly if the dataset is small or if the mixing strategy is not well-designed. It's important to carefully evaluate the performance of the model on validation or test data and experiment with different mixing strategies to find what works best for a given task and dataset.

Unlike the original MixUp , which uses a predefined distribution for the interpolation coefficient lambda and a unique value in each mini- batch, I consider using different interpolation coefficients in each mini-batch to improve the diversity and to make them all learnable.

$$\tilde{y_i} = \lambda_i y_i + (1 - \lambda_i) y_j$$

I incorporate a fully connected linear head in a custom pytorch neural model which produces one value between 0 and 1 which is the lambda which goes into mixing up labels for the strategy described before. Akin to MAML [19] and other meta-learning strategies such as that in Meta Pseudo Labels [20] , i test the performance of the previous label interpolation policy on a held out labeled validation set. I initialize the two heads ,lets call one the classification head which will take cross entropy loss on the aforementioned pseudo-labeling routine.

The other head ('lambda head') takes in as input the two teacher's decisions on an unlabeled instance and produces a fraction between 0 and 1. Effectively, I am learning the target distribution for which the student's expected cross entropy should be minimized. Moreover, by taking lets say the concatenation of the teacher's softmax logits , the model can meta-learn which teacher model's pre-



dictions should the model tend towards. Reproduced below is the computational graph for a custom ResNet-9 model .

5.0.2.1 Training routine for the meta strategy

As mentioned before, the loss for the meta-learner (lambda head) is calculated on a validation labeled set. This is done after i update the student model with the loss in figure 5.1. Therefore, there will be two backward passes in the network and we will have to retain the computational graph for the first pass. The second backward pass is admittedly computationally expensive but nevertheless a worthwhile strategy because in regular Mix-up a lot of time and computation has to be devoted to choosing the right hyper-parameter alpha ,which if not chosen wisely may lead under-fitting and degrade the performance of the classifier.

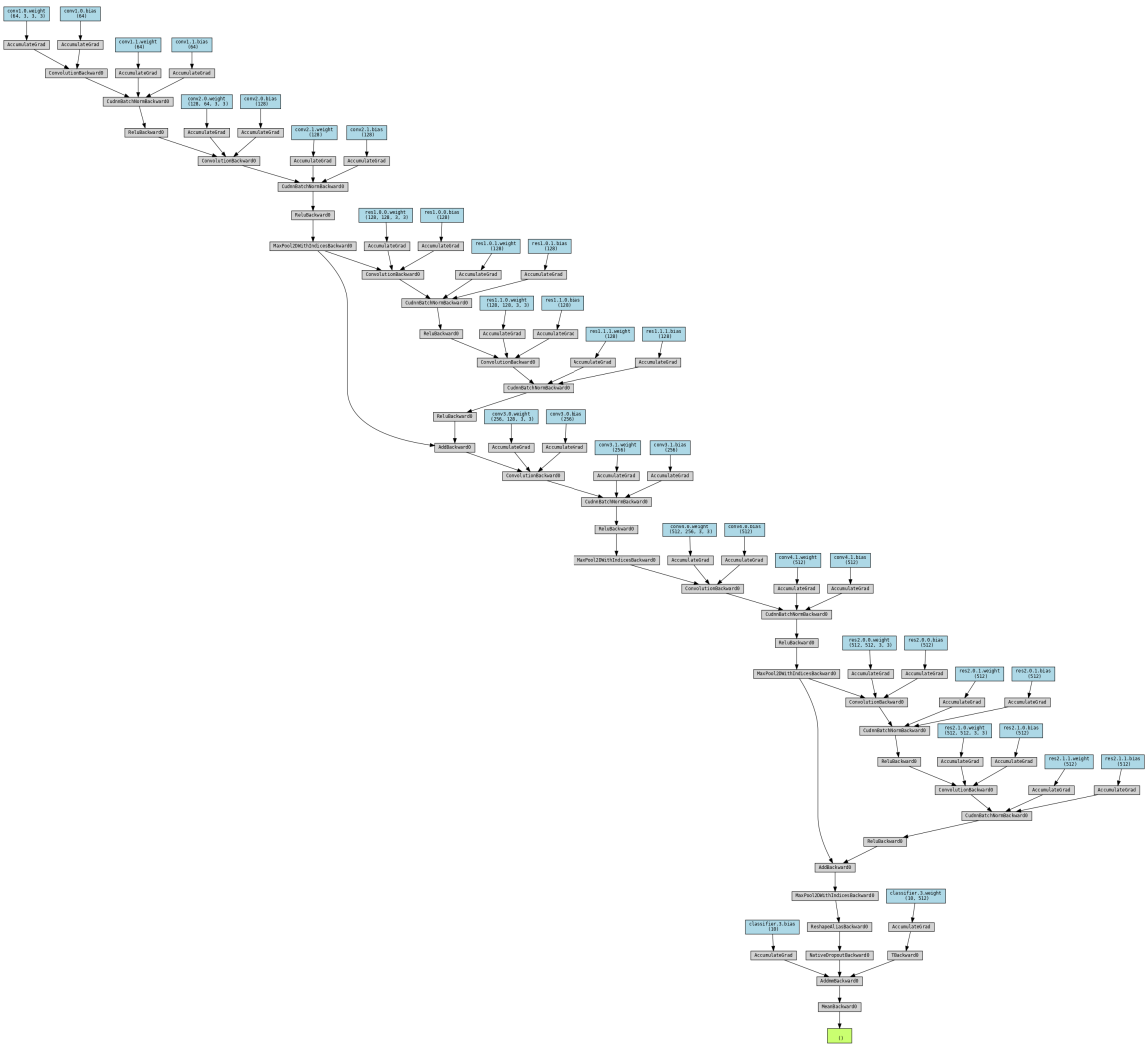


Figure 5.2: Computational Graph for ResNet meta model

Chapter 6

Results and Discussion

CIFAR-10	**Teacher Model 1**	**Teacher Model 2**	**Student Model**
Teacher Ensembling and Label Interpolation (TELI) (alpha fixed)	VGG-13 (61.2%)	VGG-11 (62.5 %)	ResNet-9 (64.6 %)
	VGG-13 (61.2%)	VGG-11 (62.5 %)	MobileNetv2 (67.8 %)
	MAE (62.7 %)	VGG-11 (62.5%)	ResNet-9 (71.4 %)
TELI + Meta-learned Lambda	**MAE (62.7 %)**	**VGG-11 (62.5%)**	**ResNet-18 (73.3 %)**
Baseline (Pseudo-labels by one teacher)	MAE (62.7 %)	—	ResNet-18 (63.2%)
	—	VGG-11 (62.5%)	ResNet-9 (62.7%)

Figure 6.1: Results on CIFAR-10

MNIST	**Teacher Model 1**	**Teacher Model 2**	**Student Model**
Teacher Ensembling and Label Interpolation (TELI) (alpha fixed)	Res-18 (75.2%)	VGG-11 (79.1 %)	MobileNetv2 (89.5 %)
(400 labels)	VGG-13 (77.2%)	VGG-11 (79.1 %)	ResNet-9 (90.1 %)
TELI + Meta-learned Lambda	**Res-18 (75.2%)**	**VGG-11 (79.1 %)**	**ResNet-9 (91.1%)**
Baseline (Pseudo-labels by one teacher)	Res-18 (75.2%)	—	MobileNetv2 (84.1%)
	—	VGG-11 (79.1 %)	ResNet-9 (85.1 %)

Figure 6.2: Results on MNIST with 400 labels

My results show that the proposed approach achieved better performance than the baseline CNN model trained only on pseudo labeled data by one teacher.

On CIFAR-10 dataset, the original model achieved an accuracy of 71.4 percent with 4000 labeled examples on the test set and 50000 unlabeled examples, compared to 63.2 percent accuracy of the baseline model.This is a huge difference considering how little the supervised set was. On CIFAR-100 dataset, the model achieved an accuracy of 44.8 percent with 4000 labeled examples and 50000 unlabeled examples, compared to 44.2 percent accuracy of the baseline model.On MNIST dataset, the model achieved an accuracy of 91.1 percent with 400 labeled examples and 59900 unlabeled examples, compared to 84.1 accuracy of the baseline model with pseudo label training with one teacher.

CIFAR-100	**Teacher Model 1**	**Teacher Model 2**	**Student Model**
Original strategy (4000 labels)	VGG-13 (42.2%)	VGG-11 (44.1 %)	ResNet-9 (44.0 %)
	VGG-13 (42.2%)	VGG-11 (44.1 %)	MobileNet (44.8%)
Baseline (Pseudo-labels by one teacher)	–	VGG-11 (44.1 %)	MobileNet (42.2 %)

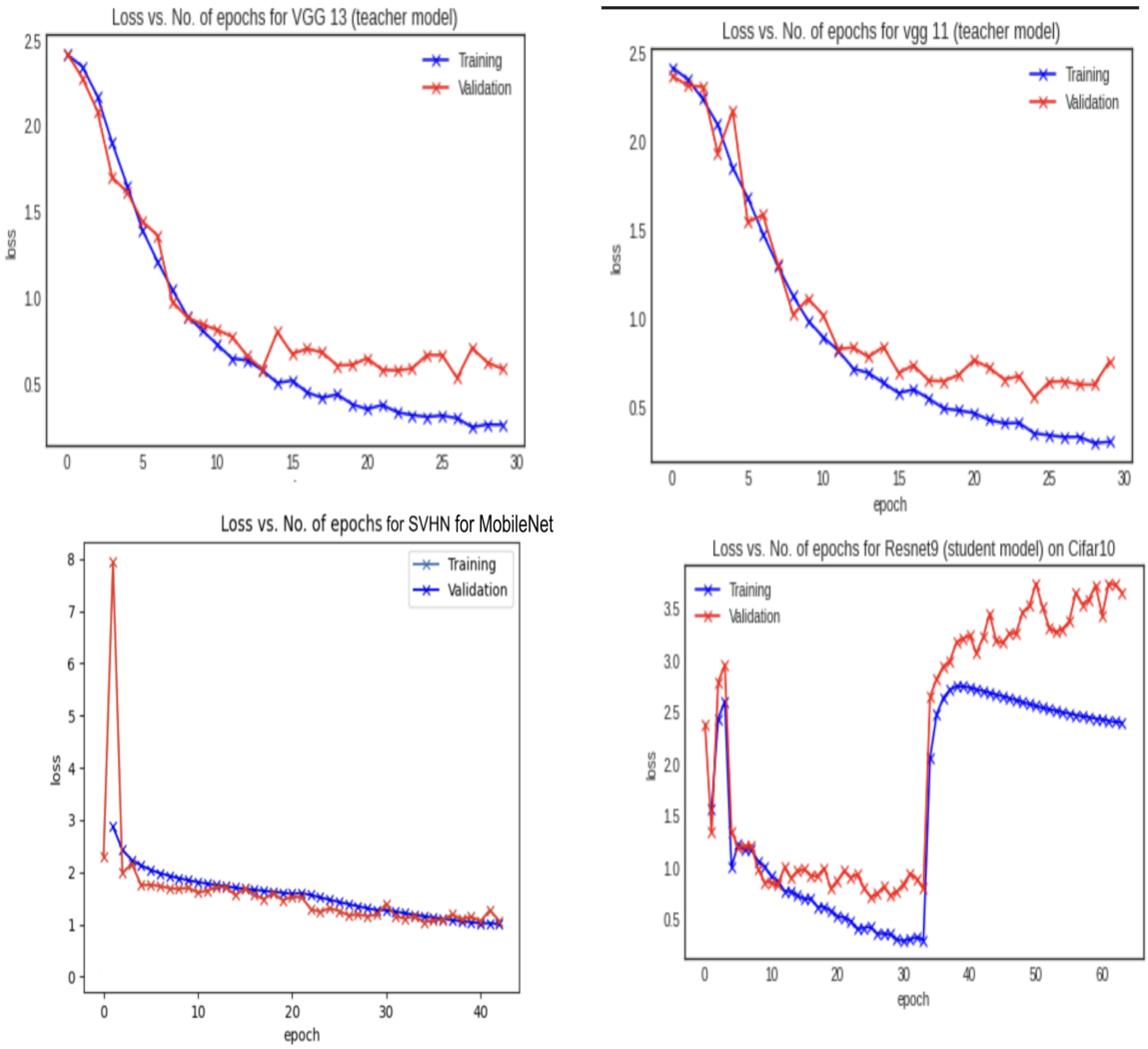
Figure 6.3: Results on CIFAR 100 table.png

SOTA Comparison

Model	**Cifar -10**	**SVHN**
Temporal Ensemble [13]	86.63 %	92.81 %
VAT [14]	86.87 %	94.65 %
ReMix-Match [15]	94.86 %	97.17 %
Fix-Match [16]	95.74 %	97.72 %
UDA[12]	94.53 %	97.11 %
Meta Pseudo Labels [6]	96.11 %	98.01 %

Figure 6.4: State of The Art comparison

I also performed Grad-CAM visualizations to visualize the parts of the image that the student model is focusing on for each class. The results show that the student model is able to focus on mostly the correct regions of the image for each class, indicating that the algorithm is learning meaningful representations.



These visualisations were produced on the validation set whose image distribution is different from the augmented training set which speaks volumes about my model's generalization abilities.

To further analyze the learned representations, I performed t-SNE visualization of the feature space of the student models and the teacher models. The results show that the cluster space has better separation of the student models as compared to the teacher models. This indicates that the student model is able to learn better representations that are more discriminative and useful for classification.

One thing that i had faced an issue with was with the flat-lining of the lambda value to 1. That would mean that the model is over-fitting lambda to one of the teacher. it could mean that the regularization term is not being effective in controlling the trade-off between the two teachers. In this case, you can try adding regularization to the lambda head itself. One possible way to do this

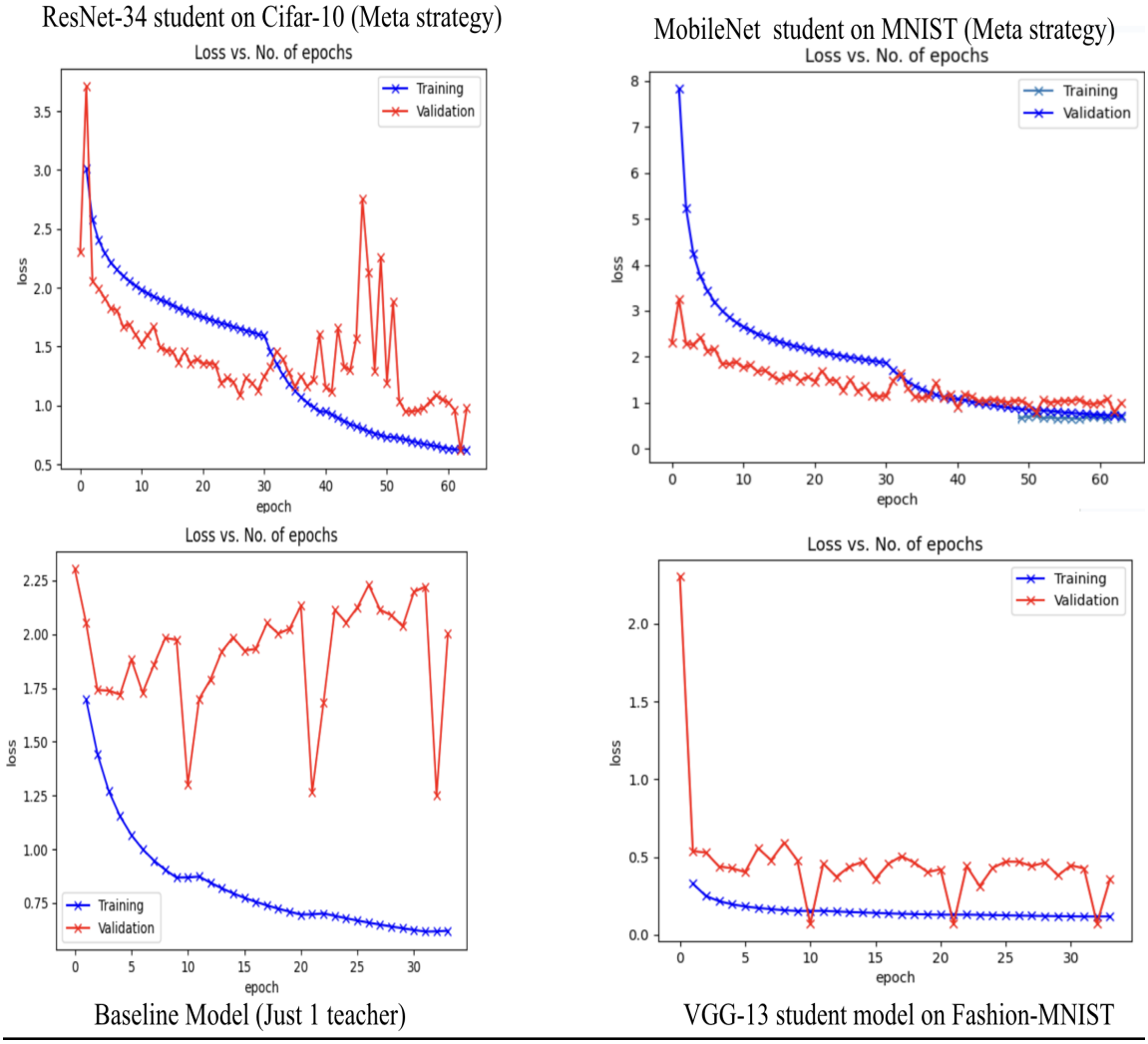


Figure 6.5: Training losses of different Models

is by adding L1 or L2 regularization to the weights of the lambda head. This will encourage the lambda values to be sparse or small, respectively, and prevent them from taking extreme values.

Alternatively, we can try using a different loss function for the lambda head that explicitly encourages a balanced trade-off between the two teachers. One example of such a loss function is the Kullback-Leibler (KL) divergence, which measures the difference between the predicted distribution and the target distribution. In this case, the target distribution would be a uniform distribution over the two teachers. By minimizing the KL divergence between the predicted and target distributions, we can encourage the lambda values to be balanced between the two teachers.

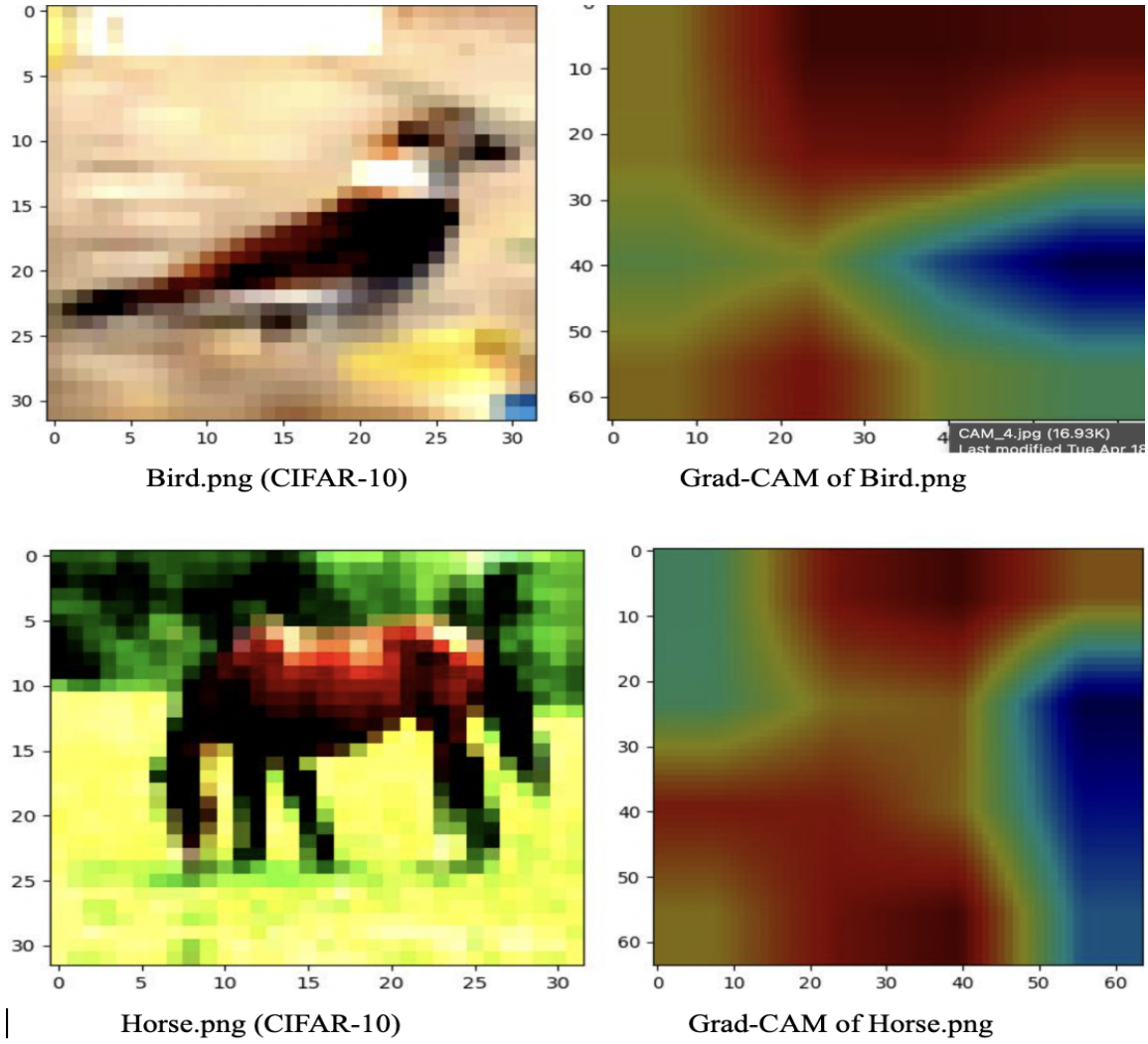


Figure 6.6: Grad-CAM Visualizations

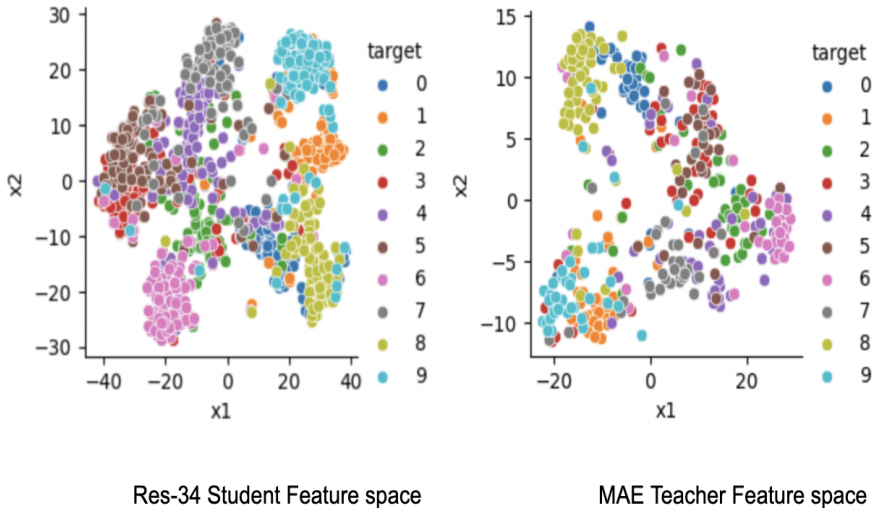


Figure 6.7: CIFAR-10 t-SNE plots of student and teachers

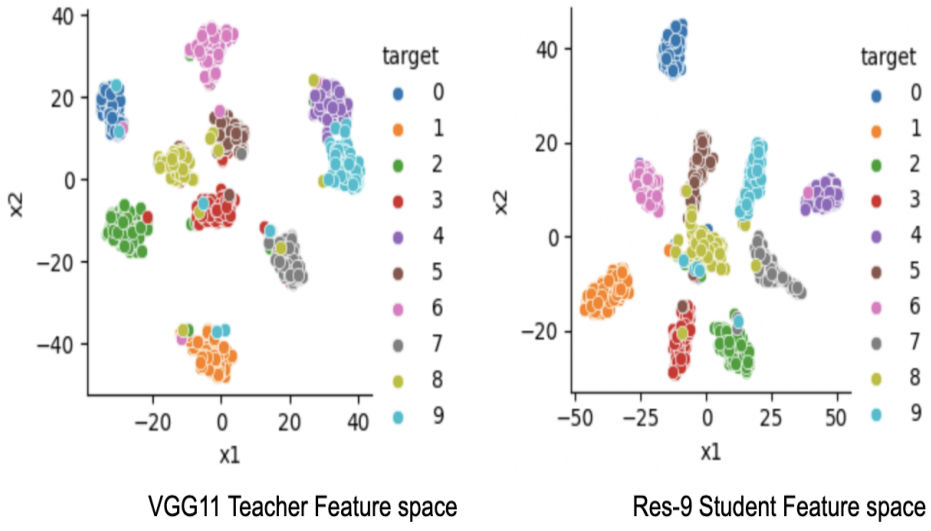


Figure 6.8: MNIST t-SNE plots of student and teachers

Chapter 7

Conclusion

In this study, I investigated the effectiveness of the pseudo label mixup method and teacher ensembling technique for semi-supervised learning on various image classification datasets including CIFAR-10, CIFAR-100, SVHN, MNIST, and FashionMNIST. Due to computational constraints,I were not able to reproduce the state-of-the-art results reported in recent studies. However, my experiments provide some insights into the performance of the proposed approach.For all experiments, I used a convolutional neural network (CNN) architecture with a mixup loss function and a lambda head for predicting the quality of pseudo labels. I also used two teacher models to generate pseudo labels for unlabeled data, and applied mixup to the pseudo labels to increase diversity. I trained the student model on labeled and unlabeled data with a combination of cross-entropy loss and mixup loss. I used the Adam and the SGD optimizers with learning rate schedulers with cosine annealing, with a learning rate of 0.001 for MNIST and 0.01 with CIFAR datasets and a weight decay of 0.0001, and trained the model for 30-50 epochs.

My results show that the proposed approach achieved better performance than the baseline CNN model trained only on labeled data.However, my results are still far from state-of-the-art results reported in recent studies. One possible reason for this is the computational constraints I faced. Our experiments were limited to a single GPU with limited memory and processing power. Therefore, I could not explore larger models and more extensive hyperparameter tuning, which may have led to better results.

Another possible reason for my suboptimal results is the limited diversity of the teacher models. I used only two teacher models to generate pseudo labels for unlabeled data, which may not have captured the full diversity of the data. Using more teacher models with different architectures and hyperparameters may lead to better performance. Furthermore, I observed that the performance of the proposed approach is highly dependent on the number of labeled examples.

My results show that the model achieved better performance with more labeled examples, especially in datasets with higher complexity such as CIFAR-100 and SVHN. Therefore, more lucrative avenues of utilizing the supervised data in a more smarter way can be explored ,such as that suggested by active learning.[21]

Another interesting observation is the effect of mixup on the quality of pseudo labels. I observed that applying mixup to the pseudo labels improved the performance of the model, especially in datasets with lower complexity such as MNIST and FashionMNIST. This indicates that mixup can increase the diversity of the data and help the model to learn more robust features.

Finally, I found that the lambda head played a critical role in the performance of the proposed approach. By predicting the quality of pseudo labels, the lambda head helped the model to distinguish between reliable and unreliable pseudo labels and focus on the more important ones.But, i couldn't produce results for this meta learning strategy because of the heavy computational nature of the algorithm.Performing two backward passes in a meta-learning algorithm can be computationally expensive because it involves recomputing the forward pass and backpropagation of gradients through the model twice. This can be particularly costly if the model is large or if the dataset used for meta-training is large.

Something akin to the Reptile algorithm can be used to implement a lighter algorithm[22]. It approximates the batch meta-gradient by performing a small number of inner gradient updates (typically one or a few) on the model using the task-specific data. This approach reduces the computational cost of the meta-learning process and has been shown to be effective in a wide range of tasks, including image classification and few-shot learning.

The key idea behind Reptile is to use the task-specific data to perform a small number of updates to the model parameters, and then use the updated model parameters to make predictions on a new task. This process is repeated for multiple tasks, and the final model parameters are obtained by taking a weighted average of the updated model parameters across all tasks.Reptile has been shown to be effective in approximating the batch meta-gradient for a wide range of tasks, and it can often achieve comparable or even better performance than other meta-learning algorithms that use two or more backward passes.

References

[1] *Lee- pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks.Pdf at master · emintham/papers*. .

[2] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, "Pseudo-labeling and confirmation bias in deep semi-supervised learning," in *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020.

[3] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond Empirical Risk Minimization," *arXiv [cs.LG]*, 2017.

[4] J. E. van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Mach. Learn.*, vol. 109, no. 2, pp. 373–440, 2020.

[5] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," *arXiv [cs.NE]*, 2016.

[6] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the eleventh annual conference on Computational learning theory*, 1998.

[7] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via Gradient-based localization," *arXiv [cs.CV]*, 2016.

[8] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked Autoencoders Are Scalable Vision Learners," *arXiv [cs.CV]*, 2021.

[9] "CIFAR-10 and CIFAR-100 datasets," *Toronto.edu*. [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html. [Accessed: 20-Apr-2023].

[10] "The Street view house numbers (SVHN) dataset," *Stanford.edu*. [Online]. Available: http://ufldl.stanford.edu/housenumbers/. [Accessed: 20-Apr-2023].

[11] *Lecun.com*. [Online]. Available: https://yann.lecun.com/exdb/mnist/. [Accessed: 20-Apr-2023].

[12] L. Gmail and G. Hinton, "Visualizing Data using t-SNE," *Jmlr.org*, 2008. [Online]. Available: https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf. [Accessed: 20-Apr-2023].

[13] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv [cs.CV]*, 2014.

[14] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *arXiv [cs.CV]*, 2015.

[15] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “MobileNetV2: Inverted residuals and linear bottlenecks,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[16] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv [cs.LG]*, 2014.

[17] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, “RandAugment: Practical automated data augmentation with a reduced search space,” *arXiv [cs.CV]*, 2019.

[18] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, “Unsupervised Data Augmentation for Consistency Training,” *arXiv [cs.LG]*, 2019.

[19] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” *arXiv [cs.LG]*, 2017.

[20] H. Pham, Z. Dai, Q. Xie, and Q. V. Le, “Meta Pseudo Labels,” in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.