Leveraging Unlabeled Data: Techniques in Semi-Supervised Learning

Nathaniel Shineman
Department of Computer Science
Ohio State University
Columbus, OH 43210
`shineman.5@buckeyemail.osu.edu`

Frederick Zhang
Department of Computer Science
Ohio State University
Columbus, OH 43210
`zhang.9975@buckeyemail.osu.edu`

Abstract

Semi Supervised Learning has proven to be a cost effective alternative to traditional fully supervised methods, while providing the same degree of user control over available classes. By limiting the expensive process of data analysis to only a fraction of the data used, SSL can greatly reduce costs. In this survey, we will examine several representative methods of SSL, along with a detailed analysis of the state of the art methods and what makes them more efficient than older alternative.

1 Introduction

Semi-supervised learning (SSL) occurs in a setting where we have access to a vast amount of data, only a small portion of which is labelled. Mathematically put, SSL is to train a model by utilizing a small amount of labeled data $\{(X_i, Y_i)\}_{i=1}^{n}$ and a large amount of unlabeled data $\{X_i\}_{i=1}^{m}$, where m » n. Recently, numerous novel neural models have been proposed for SSL setting. This paper will summarize several representative SSL algorithms by comparing similarities among models and discussing unique merits of each individual algorithm.

Algorithms In this survey we will begin with an early model of SSL: the self-ensembling model. A self-ensembling approach ensembles a model's predictions under different perturbations and uses each of them as feedbacks for learning. This method uses a single model under different configurations in order to make the model's predictions more robust [9]. Four typical neural network-based models will be discussed in detail; the Γ Model (Ladder Model), Virtual Adversarial Training, Π Model, and Temporal Ensembling [8, 7, 12, 6]. We will then compare these methods with the newer Mean Teacher model which uses a pair of neural nets to more effectively train towards an average prediction [11], and the MixMatch algorithm which combines a variety of techniques used in previous models to achieve state of the art results [1].

2 Background

2.1 Ensembling

A well documented fact of machine learning is that an ensemble of multiple neural networks generally yields better predictions than a single network. Some typical ensembling techniques are Bagging Predictors [2], Adaboost [5] and Random Forest [3]. Numerous studies have shown that ensembling multiple models can significantly improve accuracy and alleviate overfitting. Besides these explicit methods, many regularization methods have also implicitly adopted the idea of ensembling such as Dropout [10].

Preprint. Under review.

The concept of self-ensembling is first defined in [6] as "a consensus prediction is generated for the unknown labels using the outputs of the network-in-training under different regularization and input augmentation conditions. Generally speaking, self-ensembling is identical to classical ensembling approaches except that it operates on a single copy of the model, rather than using a majority prediction over multiple models. The key components of self-ensembling models are the classification module and consistency module. The classification module trains the model to predict the correct label for a given input by learning from labeled data. The main power of SSL comes from the consistency module, which utilizes unlabeled data. Different models have different definitions or usages of the consistency module, but all share the goal of integrating unlabeled data into existing supervised learning models to help complement the labeled data. This allows us to train a more robust and general neural network with less labeled data.

2.2 Using Unlabeled Data

Deep neural networks are notoriously data hungry and traditionally rely on large quantities of labeled data. This data is expensive and time consuming to collect as it requires human experts to analyze it by hand to provide ground truth labels to train a model. In contrast unlabeled data is cheap and simple to acquire, but comes with the obvious drawback of having no ground truth model to train a model. As such the primary goal of semi-supervised learning is to better leverage that data.

2.2.1 Data Augmentation

Data augmentation is a common technique that trains a model to make consistent predictions across various distortions of an image. By applying random crops, rotations, or blurs to an image the amount of unlabeled data available for training increases greatly. Although it has no ground truth label, a good model should output the same class distribution for any augmentation of the same image [1, 4]. By using augmented data we train the classifier to find the most consistently discriminating features of each class. Mathematically the model should learn features such that

$$||P_{model}(y|Augment(x);\theta) - P_{model}(y|Augment(x);\theta||_2^2 \tag{1}$$

Where P_{model} is the output of a generic model and $Augment(x)$ is a random stochastic transformation. It is important to note that the stochastic nature of the Augment method means that the two augmented data points are not identical [1].

Data augmentation can be accomplished in a variety of ways. Most traditionally rely on a random crop, blur, or rotation but can include other techniques such as information deletion [4]. In this method, a section of the input is hidden and the classifier is forced to train without that data. In the context of computer vision, this is accomplished through pixel dropping and possibly replacing the lost information with random noise. This forces the model to learn less important features and improve its ability to generalize a class [4].

2.2.2 Entropy Minimization

Entropy minimization relies on the traditional assumption that a decision boundary should not pass through a high density area of the data distribution [1]. Essentially the model must be forced to output only high confidence predictions. This is especially key when evaluating unlabeled data since there is no ground truth label to train towards. Entropy minimization can be accomplished either explicitly by adding a term to the unlabeled loss function that minimizes entropy, or implicitly by creating one hot vectors from high confidence predictions from the model [1].

2.2.3 Generic Regularization

This form of regularization forces the model to generalize better to unseen data. By imposing some constraint on the model we can prevent the model from memorizing the training data allow it to better generalize to unseen data. There are a variety of methods to achieve regularization, and frequently it is possible to combine two or more methods to further improve results. These methods can range from information deletion [4]–as mentioned with regard data augmentation–to a weight decay, in which the model weights are multiplied by a value slightly less than one to prevent them from growing too large. More recent research has begun to focus on more effective forms of regularization that greatly improve the accuracy and generalization of models trained with SSL [14, 13]. The importance

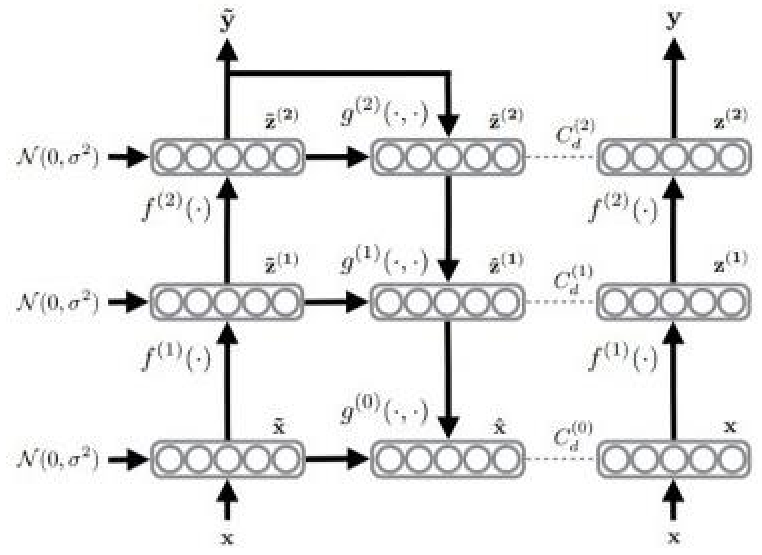

Figure 1: A conceptual illustration of the Ladder network when L = 2

of a strong regularization method cannot be overstated, and is the key to creating a model that can accurately classify unseen data.

3 Algorithms

Here we examine each of the algorithms that we compare at a high level. Each of these models represents the state of the art at some point in the history of SSL.

3.1 Γ Model (Ladder Model)

The Γ model is the oldest and the pioneer work in self-ensembling SSL. It is referred to as the Ladder Model due to the shape of model architecture. Ladder networks have been mostly used in computer vision where many forms of perturbation and data augmentation are available [8].

Figure 1 demonstrates the flow of the Γ model. For each training example–labeled or unlabeled–the model will first generate a prediction on the clean example as a proxy label, y. It will then generate a prediction on a perturbed version of the example, $\tilde{y}$. By minimizing the loss from the "corrupted" label to the proxy label, the model learns features that are invariant to noise added on the corrupted feedforward path. Beyond what is shown in the diagram, the labeled training data will be further used to minimize the loss from the proxy label to the ground truth in order to train a working classifier.

3.2 Virtual Adversarial Training (VAT)

Instead of randomly distorting the feature space of a training example through augmentation or dropout, we can directly apply the worst possible perturbation for the model. In this case the worst possible perturbation is to transform the input into its adversarial example. While adversarial training requires access to the labels to perform these adjustments, VAT requires no labels and is thus suitable for SSL [7]. The model structure of VAT is almost the same as Adversarial Training. The only difference is that a consistency/discrepancy cost is introduced to VAT. By doing so, it seeks to make the model robust to perturbations in directions to which it is most sensitive [7]. The loss term for VAT is

$$KL[p(\cdot|x;\hat{\theta})||p(\cdot|x+r_{v-adv};\theta)] \tag{2}$$

$$where\ \ r_{v-adv} = \underset{r,||r||\leq\epsilon}{argmax} KL[p(\cdot|x;\hat{\theta})||p(\cdot|x+r;\hat{\theta})] \tag{3}$$

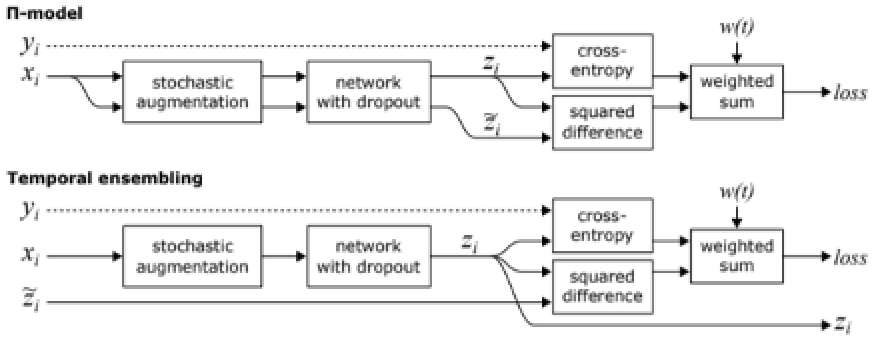

Figure 2: Structure of the training pass in the Π model and Temporal Ensembling model.

By minimizing Equation (2), a classifier is trained to be smooth and resistant to distortions introduced to training data [7]. Equation (3) above can be considered a special consistency cost. Unlike most consistency modules whose goal is to minimize the discrepancy, the consistency cost in Virtual Adversarial Training is maximized. That is, pick a r_{v-adv} such that the KL divergence is maximized. The KL divergence is maximized to reach the point of least similarity between the output labels if the feature vectors of two input examples are exactly opposite to each other. Therefore, a pair of adversarial examples will lead to two opposite outputs.

3.3 Π Model

This model is similar to the Γ model, but rather than treating clean predictions as proxy labels the Π model ensembles the predictions of the model under two different perturbations of the input data and two different dropout conditions [11]. This mitigates the issues caused by inaccurate predictions on the clean data which can occur in the Γ model. Figure 3 demonstrates the training process for the Π model. Given a training example, it is first duplicated, then each copy undergoes a stochastic augmentation and is passed to a shared network with a dropout layer to add random noise. Afterwards, we apply a squared difference loss on the two outputs z and $\tilde{z}$ to minimize the discrepancy between the two perturbed inputs. If the ground truth label is available, one output will also be used to compute the cross-entropy loss to train a better classification module.

3.4 Temporal Ensembling

Instead of ensembling over the same model under different noise configurations, we can ensemble over different models. However, training separate models is expensive. Instead, we can ensemble the predictions at different timesteps from a single model [6]. Temporal Ensembling comes from its unique feature of ensembling predictions over multiple timesteps. The lower diagram in Figure 3 demonstrates the training process of Temporal Ensembling. Given a training example we again apply a stochastic augmentation and forward the new data to a shared network with a dropout layer to get output z. z is then passed into two unique loss functions. A cross-entropy function computes the classification loss improve classification accuracy. A squared loss function computes the discrepancy from the target label $\tilde{z}$ to minimize the consistency cost. Finally, z will be used to update $\tilde{z}$ as an exponential average and used as the new target label for x. The technical novelty of Temporal Ensembling comes from the third component, which implements the idea of ensembling predictions over multiple timesteps. Temporal Ensembling also gains an approximate 2x speedup over the Π model [6].

3.5 Mean Teacher

Averaging all past predictions of a model is not scalable, as training a model can take thousands of passes over the training data. So instead it is possible to average the model weights. Mean Teacher stores an exponential moving average of the model's past parameters [11]. Figure 4 shows the training process of Mean Teacher, which uses a pair of models trained individually. First a pretrained student model is copied for the teacher model. Each training example will flow through both models under

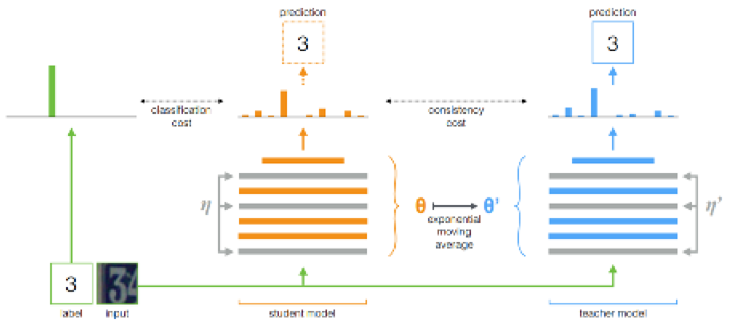

Figure 3: Structure of the Mean Teacher model.

different data augmentations and dropout conditions. For every example the teacher model is used to obtain a proxy label $\tilde{z}$, which will be compared to the prediction z from the student model. The goal of the teacher is to lead the student so that the discrepancy between z and $\tilde{z}$ will be minimized [11]. If the training data is labeled, the classification cost is also computed against the ground truth label. The student weights are updated via gradient descent, and the teacher weights are an exponential moving average of the past student weights.

3.6 Mixup and CutMix

Mixup is a regularization technique that trains a model on convex pairs of data instead of individual data points. It is designed to be a replacement for ERM, which the authors suggest encourages memorization of training data when used to optimize large neural nets [14]. Mixup overlays a pair of training points, and mixes their labels proportionately. Using this combined point the model learns features that might otherwise be ignored. This encourages the model to display linear behavior between points and mitigates overfitting of the training data. That in turn allows the model to better generalize to unseen data, which does not fit the exact feature distribution exhibited in the training data [14]. However, one drawback of Mixup is its tendency to generate unnatural looking images leading to poor localization [13]. This makes it much less suitable for object detection tasks than image classification. To alleviate this, the authors of CutMix [13] suggest that instead of overlaying two images, we cut and paste a random segment of an image onto another. This creates a combined image that both shares features from its two sources, and maintains a distinct region for each class. CutMix has been shown to exhibit significant improvements in both localization and transfer learning capabilities over Mixup [13].

3.7 MixMatch

MixMatch is a state of the art SSL algorithm that provides a more efficient means of leveraging unlabeled data. Instead of computing multiple loss terms, MixMatch unifies the three dominant approaches to SSL–data augmentation, entropy minimization, and general regularization–into a single loss function [1]. First we generate pseudo labels for the unlabled data in a batch by applying K augmentations to an image x, and computing the average model prediction across them. A sharpening function is then applied to decrease the entropy of the model's high confidence predictions.

$$Sharpen(p,t)_i := p_i^{\frac{1}{T}} / \textstyle\sum_{j=1}^{L} p_j^{\frac{1}{T}} \tag{4}$$

The sharpening function can been seen in equation (4) [1]. This becomes the training target for the unlabeled image. For labeled data, a single augmentation is also applied to improve model consistency. The labeled and unlabeled augmentations are combined into a single set W, and Mixup applied between labeled data and random examples from W, and unlabeled data with random examples from W [1]. These points are then returned to be classified by the model and used to compute the final loss term. The combined loss is a computed as

$$X', U' = MixMatch(X, U, T, K, \alpha) \tag{5}$$

$$\mathcal{L}_x = \frac{1}{X'} \sum_{x,p\epsilon X'} H(p, P_{model}(y|x;\theta)) \tag{6}$$

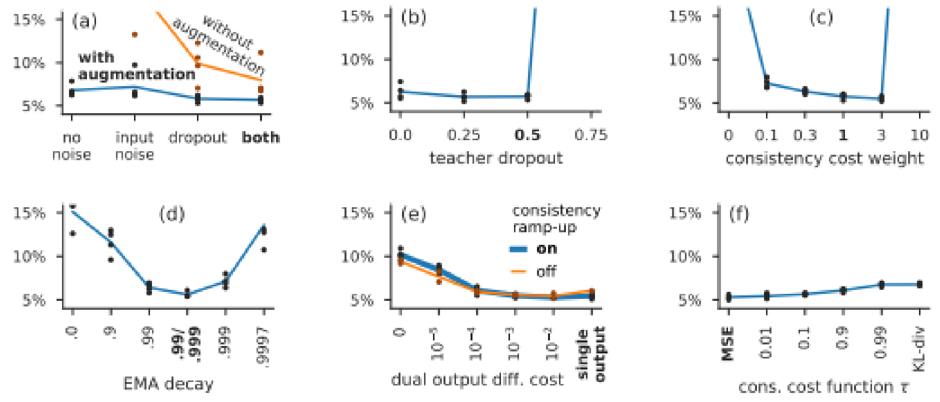

Figure 4: Ablation study on Mean Teacher using SVHN [11]

$$\mathcal{L}_u = \frac{1}{U'} \sum_{x,q\epsilon U'} ||q - P_{model}(y|u;\theta)||_2^2 \tag{7}$$

$$\mathcal{L} = \mathcal{L}_x + \lambda_U \mathcal{L}_U \tag{8}$$

Where X' and U' are the results from Mixup across the labeled and unlabeled data sets and their ground truth or pseudo-labels, Equation (6) is a standard cross entropy loss, Equation (7) is a squared L_2 loss, and λ_U is a the weight decay on the unsupervised loss [1]. Minimizing this loss term elegantly achieves consistency regularization across multiple augmentations, entropy minimization of unlabeled training data, and strong generalization through Mixup.

4 Expiriments

MixMatch achieves state of the art results in image classification tasks on a variety of datasets. As the most resent SOTA algorithm, we will focus on the experiments performed by the authors of [1]. In all cases, the algorithms have been re-implemented in the same code base to insure consistency in testing.

CIFAR On CIFAR-10–shown in Figure 4–with only 250 labels, MixMatch outperforms the next best algorithm at 250 labels (VAT) by over 4.5x. Additionally with 250 labels, it nearly equals the performance of the overall next best model (Mean Teacher) using 4000 labels [1]. Using 4000 labels MixMatch achieves nearly fully supervised accuracy, 6.24% compared to 4.17%. Using a wider network for both CIFAR-10 and CIFAR-100 (26 million parameters compared to 1.5 million in the first test), the authors directly compare the results of MixMatch and MeanTeacher and its predecessor Stochastic Weight Averaging (SWA). On CIFAR-100 MixMatch achieves an error rate of 25.88%, compared to 28.80% from SWA [1].

SVHN On SVHN MixMatch even further outperformed the other algorithms. With just 250 labels, MixMatch not only outperforms every other tested algorithm, including Mean Teacher with 4000 labels, but reaches almost fully supervised accuracy. When incorporating the SVHN+Extra dataset, the authors of [1] were able to achieve even stronger results. With the higher ratio of unlabeled to labeled data and only 250 labels, MixMatch is able to achieve a 2.22% error rate, compared to 2.59% when fully supervised learning is used to train the model on the standard SVHN training dataset (73,257 labels).

Mean Teacher Ablation Studies In evaluating Mean Teacher, the authors of **(author?)** [11] examine the effect of removing random noise (Figure 4(a) and 4(b)), the EMA decay rate (Figures 4(c) and 4(d)), separating the classification and consistency modules (Figure 4(e)), and KL-divergence instead of MSE for consistency cost (Figure 4(f)). Their results show that the input augmentation and dropout are vital to the success of the student model, but that adding dropout to the teacher has

Table 1: MixMatch ablation study error rates on CIFAR-10 [1]

Ablation	250 Labels	4000 Labels
MixMatch	11.80	6.00
MixMatch without distribution averaging (K=1)	17.09	8.06
MixMatch with K = 3	11.55	6.23
MixMatch with K = 4	12.45	5.88
MixMatch without temperature sharpening (T = 1)	27.83	10.59
MixMatch with parameter EMA	11.86	6.47
MixMatch without MixUp	39.11	10.97
MixMatch with MixUp on labeled only	32.16	9.22
MixMatch with MixUp on unlabeled only	12.35	6.83
MixMatch with MixUp on separate labeled and unlabeled	12.26	6.50

minimal effect. The study also demonstrates that using values outside of the "optimal" range for either consistency cost or weight decay causes performance to degrade sharply. Note that using a weight decay of 0 make Mean Teacher a modified Π model [11]. Most importantly, this study also demonstrates that a moderately coupled loss term provides optimal performance. Too tightly or too loosely coupling the consistency and classification loss results in worse performance. Both of these concepts strongly inform ideas implemented in MixMatch.

MixMatch Ablation Studies The authors of MixMatch [1] also include an ablation study to validate their results. Since MixMatch has a variety of key components, an exhaustive list of the expiriments is not included here, instead the results are summarized in Table 1.

5 Discussion

Since MixMatch and Mean Teacher have achieved the most recent state of the art results, our discussion will focus primarily on comparing these two. Mean Teacher has much in common with the older Π model, but uses a pair of models that improve each other cyclically. As shown in the ablation study, it also couples the classification and consistency modules leading to much stronger results [11]. MixMatch achieves a similar coupling affect by unifying the loss term for both labeled and unlabeled data, and combining the benefits of both using Mixup. Both algorithms, along with every other successful SSL algorithm, require some form of input perturbation on unlabeled data to achieve strong results. The most important features that a model can learn from unlabeled data are those that are invariant to augmentation, which indicates that optimizing the augmentation process is the key to further developing SSL algorithms. However, the MixMatch ablation study also demonstrates that there is a limit to what augmentation can achieve. Experiments using more than two augmentations lead to diminishing returns, or even worse results as the classifier becomes confused.

Furthermore, both Mean Teacher and MixMatch use a unique weight decay that prevents the model from relying too heavily on features learned earlier in the training process [1, 11]. In the case of Mean Teacher the weight decay is adjusted after several passes on the training data to further reduce the teacher model's dependence on early student predictions. Mix Match alleviates this issue by using a single model prediction on the un-augmented image as a proxy label, much like the Γ model. This bypasses the need for two models, and is shown in Table 1 to slightly improve performance when compared to using the EMA instead. Overall this make MixMatch easier to implement while maintaining higher accuracy.

The results seen in Figures 5 and 6 demonstrate the most promising aspect of MixMatch when compared to the other methods discussed here. MixMatch is able to achieve comparable performance to Mean Teacher with 1/16 of the labeled data [1]. On SVHN it achieves nearly fully supervised accuracy with only 250 labels. These low label cases show that MixMatch demonstrates unprecedented sample efficiency, which vastly improve the prospects of SSL. Since the primary goal of SSL is to reduce the cost associated with acquiring labeled data, the ability of a model to achieve near supervised results with less labeled data is paramount. As discussed **Expiriments**, when the SVHN Extra set was added to the training data MixMatch was able to outperform fully supervised learning on SVHN. The authors summarize this finding with the following example: given the 73257 examples from

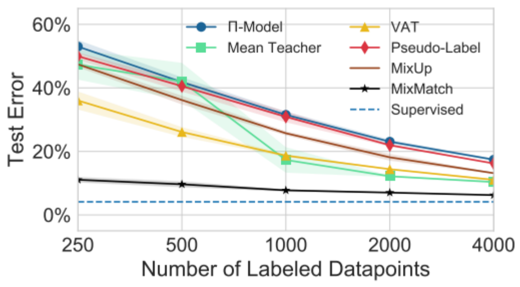

Figure 5: Performance of various algorithms on CIFAR-10 dataset. MixMatch achieves an error rate of 11.08% with only 250 labels, while VAT achieves 36.03% error rate with 250 labels and Mean Teacher requires 40000 labels to achieve a comparable 10.36% error rate [1].

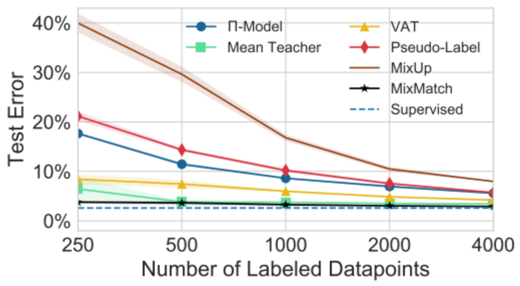

Figure 6: Performance of various algorithms on SVHN dataset. With 250 labels MixMatch nearly achieves fully supervised learning rate [1].

SVHN with 250 labels, you have the choice of acquiring 8x more unlabeled data to use MixMatch, or 293x more labeled data to use fully supervised learning [1]. The performance of MixMatch in their tests indicates that using the unlabeled data and applying MixMatch will perform better than using fully supervised learning, while also likely being much cheaper than acquiring additional labeled data.

References

[1] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. Mixmatch: A holistic approach to semi-supervised learning, 2019.

[2] Leo Breiman. Bagging predictors, 1994.

[3] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[4] Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Gridmask data augmentation, 2020.

[5] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119 – 139, 1997.

[6] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning, 2016.

[7] Takeru Miyato, Andrew M. Dai, and Ian Goodfellow. Adversarial training methods for semi-supervised text classification, 2016.

[8] Antti Rasmus, Harri Valpola, Mikko Honkala, Mathias Berglund, and Tapani Raiko. Semi-supervised learning with ladder networks, 2015.

[9] Sebastian Ruder and Barbara Plank. Strong baselines for neural semi-supervised learning under domain shift, 2018.

[10] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[11] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results, 2017.

[12] David Warde-Farley and Ian Goodfellow. 11 adversarial perturbations of deep neural networks. *Perturbations, Optimization, and Statistics*, 311, 2016.

[13] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features, 2019.

[14] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization, 2017.