


Deep Semi-Supervised Image Classification Algorithms: a Survey


Ani Vanyan

(YerevaNN, Yerevan, Armenia)

 <https://orcid.org/0000-0002-6203-7146>, ani@yerevann.com)

Hrant Khachatryan

(YerevaNN; Department of Informatics and Applied Mathematics, Yerevan State University,
Yerevan, Armenia)

 <https://orcid.org/0000-0002-1544-5649>, hrant@yerevann.com)

Abstract: Semi-supervised learning is a branch of machine learning focused on improving the performance of models when the labeled data is scarce, but there is access to large number of unlabeled examples. Over the past five years there has been a remarkable progress in designing algorithms which are able to get reasonable image classification accuracy having access to the labels for only 0.1% of the samples. In this survey, we describe most of the recently proposed deep semi-supervised learning algorithms for image classification and identify the main trends of research in the field. Next, we compare several components of the algorithms, discuss the challenges of reproducing the results in this area, and highlight recently proposed applications of the methods originally developed for semi-supervised learning.

Keywords: Machine learning, Semi-supervised learning, Consistency regularization, Image classification, Survey

Categories: I.2.6, I.4, A.1

DOI: 10.3897/jucs.77029

1 Introduction

The rise of convolutional neural networks over the last decade revolutionized image analysis. With large scale human-annotated datasets like ImageNet [Deng et al., 2009], it is possible to learn an image classifier with close-to-perfect accuracy. The next challenge is to get similar results with significantly less human labeling effort.

In this paper we describe the latest advances in semi-supervised learning, a branch of machine learning focused on improving the performance of an algorithm using a small set of labeled and a large set of unlabeled samples.

These algorithms rely on the premise that obtaining unlabeled data is cheap. Although we have to note that most of these algorithms fail when the distribution of the unlabeled data is different from the distribution of the labeled data. In fact, some methods use special tricks that explicitly require the distributions to be the same.

There is a lot of literature on semi-supervised learning. Many recent papers refer to two classical works for a general overview [Chapelle et al., 2006, Zhu and Goldberg, 2009]. In this survey we will focus on more recent algorithms which build on top of existing neural network architectures designed for regular supervised learning. These algorithms introduce regularization terms to the loss functions, augment the inputs and

perform various tricks developed for self-supervised learning. This paper notably does not cover transductive SVMs (which were a popular method in early 2000s), graph-based methods and methods based on generative models.

In parallel work, [Ouali et al., 2020] compiled another survey on semi-supervised learning for image classification, including graph-based and generative methods. Our work has more focus on critical analysis and comparison of the proposed algorithms.

This paper is an extension of the work presented at CODASSCA 2020 workshop [Vanyan and Khachatrian, 2020]. The paper is organized as follows. We start from basic definitions. Section 3 continues with the description of consistency regularization and describes the first attempts to make it reliable for a few benchmark datasets. Sections 4 and 5 describe the two major research directions in deep semi-supervised learning research. Section 6 discusses the important details of the algorithms and the challenges of proper evaluation and reproducibility. Finally, Section 7 shows two applications of the methods developed for semi-supervised learning.

2 Definitions

Let X denote the labeled dataset with samples $(x, p) \in X$. Here, p is a one-hot C -dimensional vector indicating the ground-truth label of x , where C is the number of classes in the dataset. U denotes a dataset without labels. It is assumed that the samples in X and U are drawn from the same distribution.

Let $f_\theta(x)$ be a function (neural network) with parameters θ . It outputs a probability distribution on the labels. The goal of deep semi-supervised learning is to learn parameters θ using labeled and unlabeled datasets such that f_θ produces correct labels for unseen samples.

Data augmentation is a critical component in many semi-supervised learning algorithms. By $Augment(x)$ we denote a *stochastic* operation that augments the sample x such that its label remains the same. In case of images, $Augment(x)$ might crop or resize the image, change the brightness or color saturation etc. By $H(\cdot, \cdot)$ we denote the cross entropy: $H(p, q) = -\sum_i p_i \log(q_i)$.

Most of the algorithms described in this paper are being tested on popular image classification datasets: CIFAR-10, CIFAR-100 [Krizhevsky et al., 2009], SVHN [Netzer et al., 2011] and ImageNet [Deng et al., 2009]. CIFAR datasets have 50000 images for training, SVHN has 73257 images. All these datasets are designed for supervised learning, so to use them in a semi-supervised setup, part of their labels is hidden from the algorithms during the training. On the other hand, the validation sets are usually kept intact, which makes these setups a little bit unrealistic. Some papers also perform experiments on STL-10 dataset [Coates et al., 2011] which by design has a large subset of unlabeled examples. SVHN also has a special extension called SVHN-extra with 531K additional images which is sometimes used as an additional source of unlabeled images.

3 Consistency regularization

The main concept that drives research in semi-supervised learning for the past five years is called consistency regularization. The core idea is to make sure the neural network produces similar results for the augmented versions of the same unlabeled image. It is

enforced by an additional term in the loss function:

$$L_U = \frac{1}{|U|} \sum_{x \in U} \|f_\theta(\text{Augment}(x)) - f_\theta(\text{Augment}(x))\|_2^2 \quad (1)$$

Note that $\text{Augment}(x)$ is a stochastic function, and f_θ might also be stochastic (e.g. due to dropout). So the difference is most likely non-zero.

3.1 Π -model

As far as we know, consistency regularization was first introduced in an algorithm called Π -model [Laine and Aila, 2017]. In particular the authors used the following loss function:

$$L = L_X + \lambda(t)L_U \quad (2)$$

$$L_X = \frac{1}{|X|} \sum_{(x,p) \in X} H(p, f_\theta(x)) \quad (3)$$

Here, $\lambda(t)$ is the relative weight of the consistency loss term, which slowly grows from zero to its final value λ over the training process. λ is a hyperparameter. In Π -model, $\text{Augment}(x)$ means just two operations: translation by $a \sim \text{Uniform}(-2, 2)$ pixels and a random horizontal flip (for all datasets, except SVHN). This augmentation strategy is called *weak augmentation* in many subsequent papers.

3.2 The problem of the unstable target

One critical problem with the above formulation of the consistency loss is that it is not stable. This was discovered in the same paper and a partial solution was given. The authors suggested to update one of the terms in the consistency loss (the one with no augmentation) less often and slowly. In particular, they update it once per epoch, and use exponential moving average of the outputs of the snapshots taken at each epoch. This trick is called temporal ensembling.

$$f_{temp.ens}(x) = \alpha f_{temp.ens}(\text{Augment}(x)) + (1 - \alpha)f_\theta(\text{Augment}(x)) \quad (4)$$

$$L_U = \frac{1}{|U|} \sum_{x \in U} \|f_\theta(\text{Augment}(x)) - f_{temp.ens}(x)\|_2^2 \quad (5)$$

The first formula is computed once per epoch. Indeed, this helped to stabilize the training and improve the performance.

3.3 Mean Teacher

The authors of Mean Teacher algorithm [Tarvainen and Valpola, 2017], presented in NIPS 2017, gave a better solution to the unstable target problem. They use two separate models: a Student network with θ parameters and a Teacher with θ' parameters. Student is trained as usual. Teacher is not trained via backpropagation. Instead, its weights are updated at each iteration using the weights from the Student network:

$$\theta'_t = \alpha \theta'_{t-1} + (1 - \alpha)\theta_t \quad (6)$$

On unlabeled examples, the Teacher network provides the learning target:

$$L_U = \frac{1}{|U|} \sum_{x \in U} \|f_{\theta}^{student}(Augment(x)) - f_{\theta'}^{teacher}(Augment(x))\|_2^2 \quad (7)$$

This work highlighted another problem in the space of semi-supervised learning algorithms. The number of hyperparameters in such setups is getting quite large: the choice of the neural architecture (backbone); the ratio of labeled and unlabeled examples in a batch; early stopping criteria; decay rate α in the exponential moving average formula; learning rate schedule; weight decay.

In such conditions, one way to make comparisons with earlier work more fair is to re-implement the old models in the same codebase. The authors of Mean Teacher re-implemented Π -model. Mean teacher consistently worked better than the previous methods.

4 Stronger Augmentations

One line of research to improve the performance of semi-supervised algorithms is to use various kinds of data augmentation techniques, so that the inputs given to the two branches of the neural model (or, to the two separate networks) are sufficiently different.

4.1 Virtual Adversarial Training and Entropy Minimisation

Instead of using data-specific augmentation functions, the authors of [Miyato et al., 2018] suggest to generate an adversarial example. The idea is similar to the adversarial training method introduced in [Goodfellow et al., 2014b], when the regular loss function is applied to the perturbed version of the input sample:

$$L_{adv} = H(p, f_{\theta}(x + r_{adv})) \quad (8)$$

$$r_{adv} = \arg \max_{r: \|r\| < \epsilon} H(p, f_{\theta}(x + r)) \quad (9)$$

Note that r_{adv} can be approximated using Fast Gradient Sign method introduced in the same paper: $r_{adv} = \epsilon \text{sgn}(\nabla_x H(p, f_{\theta}(x)))$. Also note that this operation requires access to the correct label p . For unlabeled examples, we do not have p , so we calculate r_{adv} by taking the perturbation which changes the prediction of the network by the largest magnitude (measured by cross-entropy):

$$r_{adv} = \arg \max_{r: \|r\| < \epsilon} H(f_{\theta}(x), f_{\theta}(x + r)) \quad (10)$$

The authors of [Miyato et al., 2018] suggest a fast approximation of this operation.

Another notable difference in VAT and previous methods is that VAT uses cross-entropy instead of Euclidean distance in the consistency loss term.

Another problem with semi-supervised learning methods is the lack of confidence in predictions on unlabeled examples. On the other hand, it is assumed that each unlabeled image belongs to only one class, so the prediction on each image should have low entropy. There are several ways to achieve that, but the first one, as far as we could find, was

introduced in the same paper. It adds an additional term into the loss function to minimize the entropy of predictions.

$$L_{ent} = \frac{1}{|X| + |U|} \sum_{x \in X \cup U} H(f_{\theta}(x)) \quad (11)$$

$$H(p) = - \sum_i p_i \log p_i \quad (12)$$

4.2 MixMatch

MixMatch is an algorithm that combines many ideas, including consistency regularization, exponential moving average of network weights and a special trick for obtaining new unlabeled examples called MixUp.

MixUp, introduced in [Zhang et al., 2017], is a way to construct new samples by taking a convex combination of existing samples. For each pair of samples (x_1, p_1) and (x_2, p_2) , MixUp performs the following steps:

1. Sample $\lambda \sim \text{Beta}(\alpha, \alpha)$
2. $\lambda' = \max(\lambda, 1 - \lambda)$ to make sure it's close to 1
3. $x' = \lambda' x_1 + (1 - \lambda') x_2$
4. $p' = \lambda' p_1 + (1 - \lambda') p_2$
5. Return (x', p')

The second step was introduced in MixMatch and makes sure that the samples from $\text{MixUp}(A, B)$ are "closer" to A .

MixUp can be interpreted as an advanced augmentation algorithm. Its main difference from regular methods is that it constructs new samples based on *two* original samples. As these two samples can have different labels, the resulting sample's label is set to be the convex combination of the two labels in the C dimensional space, where C is the number of classes.

The semi-supervised learning algorithm used in MixMatch paper is essentially the same as in the papers discussed above, except there is an additional stage of modifying both the labeled and unlabeled sets. This stage is called MixMatch.

$$\begin{aligned} X', U' &= \text{MixMatch}(X, U, T, K, \alpha) \\ L_X &= \frac{1}{|X'|} \sum_{(x', p') \in X'} H(p', f_{\theta}(x')) \\ L_U &= \frac{1}{|U'|} \sum_{(x', q') \in U'} \|q' - f_{\theta}(x')\|_2^2 \\ L &= L_X + \lambda(t) L_U \end{aligned}$$

In short, $\text{MixMatch}(X, U)$ function applies MixUp to both labeled and unlabeled examples, and uses the average prediction of multiple augmented versions of the same unlabeled image. As taking average might reduce the entropy in the predicted distribution,

the authors perform an additional step of sharpening the probabilities with temperature T which is another hyperparameter.

The authors attempt to follow the setup used in [Oliver et al., 2018]. They note, that the best values for two of the new hyperparameters they have introduced do not vary in the datasets they have tested on: sharpening temperature is always set to $T = 0.5$ and the number of augmentations performed on the same unlabeled image is always $K = 2$. Instead, the best values for α hyperparameter of the Beta distribution used in MixUp and the coefficient λ in the main loss function are different for CIFAR versions and SVHN.

To make evaluations more stable, they use an exponential moving average of the model parameters with a decay rate of 0.999 when evaluating on the validation set. The results beat all previously reported results. This is the first paper which tests the performance of SSL algorithms on CIFAR-10 with only 250 labels.

4.3 Unsupervised Data Augmentation

Unsupervised Data Augmentation (UDA) is another model quite similar to VAT, which replaces virtual adversarial example generation with a very strong augmentation. In particular, they use RandAugment [Cubuk et al., 2019b], which at the time was the strongest data augmentation method known for CIFAR datasets. RandAugment is inspired by AutoAugment [Cubuk et al., 2019a]. AutoAugment uses a search method to combine all image processing transformations in the Python Image Library (PIL) to find a good augmentation strategy. In RandAugment, search is not used, instead the augmentations are uniformly sampled from the same set of transformations in PIL. Basically, RandAugment is simpler and requires no labeled data as there is no need to search for optimal policies. It is important to note, that it is not obvious how RandAugment should be configured for other datasets. As with many other models, the branch of the network which guesses the label on the non-augmented version of the image uses a fixed copy of weights and does not pass the gradient through.

Additionally, UDA uses a training technique, called Training Signal Annealing (TSA), to reduce overfitting when there is a huge gap between the amount of unlabeled data and that of labeled data. TSA gradually releases the “training signals” of the labeled examples as training progresses. It ignores a labeled example if the model’s confidence on that example is higher than a predefined threshold, which increases according to a schedule. The threshold for the confidence is increased during the training by one of the three rules: logarithmic, linear and exponential.

UDA is tested on CIFAR-10 and SVHN, but also on several sentence classification tasks. To perform data augmentation on sentences, they used backtranslation: translated each sentence into French and back into English using an existing machine translation model. Although they reported state-of-the-art results on almost all benchmarks, this paper was also rejected from ICLR 2020 due to the lack of novelty¹. Later it was accepted at NeurIPS 2020.

4.4 ReMixMatch

The team behind MixMatch made their algorithm even more complicated by adding two more components. The resulting system, called ReMixMatch [Berthelot et al., 2019], was presented at ICLR 2020 conference. The two main additions are:

¹ <https://openreview.net/forum?id=ByeL1R4FvS>

1. Distribution alignment. The predicted probabilities on a batch of unlabeled examples are scaled to match the distribution of the labels present in the labeled subset. This allows to get significantly higher accuracies in CIFAR-100 with very limited labeled examples. In practice, the scaling coefficients are estimated using a running average of 128 batches.
2. Anchored augmentation. The target label (or probability distribution over labels) for unlabeled examples is determined using *weakly* augmented versions of the images, while the prediction for the same images is computed by using *strongly* augmented versions. Weak augmentation is the same augmentation used in previous works. Strong augmentation used in ReMixMatch is called CTAugment. CTAugment uniformly samples transformations from Python Image Library to apply to the images (just like RandAugment) but dynamically updates the magnitudes for each operation during the training. CTAugment's advantage is that it does not require an optimization on a supervised task and has no sensitive hyperparameters, so it can easily be integrated in semi-supervised models. The authors provide the following intuition for CTAugment: for each transformation CTAugment learns the likelihood that it will produce an image which is classified correctly.

There are few other tricks, like using a self-supervised loss of predicting the rotation angle (an idea borrowed from S4L model, see Section 5.2). ReMixMatch shares values with MixMatch for multiple hyperparameters, but notably, the number of strongly augmented samples used in the consistency loss term is changed from $k = 2$ to $k = 8$. In addition to the experimental setups used in previous papers, the authors report performance on CIFAR-10 with only 40 labels, although they mention that they had to change one hyperparameter to make their model work in that setup: the coefficient for the loss term responsible for rotation prediction.

4.5 FixMatch

FixMatch [Sohn et al., 2020] is another iteration on this direction. Similar to UDA, it uses weakly augmented version of an image to guess the label, and forces the network to output the same label on a strongly augmented version of the same image. FixMatch uses CutOut [DeVries and Taylor, 2017] along with RandAugment or CTAugment as a strong augmentation, which makes it even stronger. In contrast to UDA and other methods, FixMatch performs $\arg \max$ on the guessed label, so it essentially becomes equivalent to pseudo-labeling. Additionally, FixMatch ignores the guessed labels if the confidence is lower than $\tau = 0.95$ threshold.

Unlike MixMatch, FixMatch does not change the λ weight of the consistency loss term during the training. The thresholding operation likely compensates for that. λ is always set to 1. FixMatch uses SGD with a cosine annealing schedule. It does not use MixUp, sharpening or distribution alignment.

The paper does extensive analysis on the role of various components, which is relatively rare in this field. The outcomes of their analysis might be helpful in subsequent research:

1. Stochastic gradient descent with momentum performs better than Adam optimizer
2. Nesterov momentum is not significantly better than the regular momentum
3. Weight decay is extremely important. Changing weight decay value by 10x might result in 10% absolute increase in error rate.

4. Sharpening instead of pseudo-labeling does not significantly change the results, so pseudo-labeling is chosen for simplicity.
5. The batch size for unlabeled data is set to be $\mu = 8$ larger than the one for labeled data. $\mu < 8$ results in worse performance. $\mu > 8$ does not improve the performance.
6. Cosine decay of learning rate performs better than no decay. The difference between linear and cosine decay schemes is not significant.
7. ImageNet requires a completely different set of hyperparameters (See Appendix C of [Sohn et al., 2020]). The experiments are performed with batch size 1024 for labeled and 5120 for unlabeled samples.

Finally, FixMatch reports performance on an extremely low label scenario which the authors call “barely supervised learning”. When only one image is available per class, the performance of FixMatch strongly depends on the choice of that single image. They show that when the samples are “representative” of the class, the performance can reach 80% using only 10 labels (one per class). On the other hand, if the samples are poorly chosen, the accuracy is below 40%. The “representativeness” is measured by a technique which requires all available labels, so this paper does not suggest an automatic method to determine the best samples.

4.6 AlphaMatch

AlphaMatch [Chengyue Gong, 2020] is another improvement on top of UDA. The authors propose to use alpha-divergence to measure the label consistency instead of Euclidean distance. They replace the consistency regularization with

$$\mathbb{E}_{x \sim U, x' \sim P_x} [D_\alpha(p_{\theta_t}(\cdot|x) || p_\theta(\cdot|x'))], \quad (13)$$

here P_x is a distribution that prescribes a random augmentation on x that keeps the label of x unchanged, $D_\alpha(\cdot||\cdot)$ is the alpha divergence with $\alpha \in (0, 1) \cup (1, \infty)$, defined as:

$$D_\alpha(p_{\theta_t}(\cdot|x) || p_\theta(\cdot|x')) = \frac{1}{\alpha(\alpha - 1)} E_{y \sim p_{\theta_t}(\cdot|x)} [\rho D_\alpha(y|x)] - 1, \quad (14)$$

with

$$\rho D_\alpha(y|x) := \left(\frac{p_{\theta_t}(y|x)}{p_\theta(y|x')} \right)^{\alpha-1}. \quad (15)$$

Notice that $D_\alpha(\cdot||\cdot)$ reduces to KL divergence when $\alpha \rightarrow 0$ or 1.

The authors show that using large α (e.g., $\alpha = 1.5$) can potentially obtain better results than smaller α , but the training is more unstable and may eventually diverge to worse results. This is because the iterative update of parameters does not correspond to an optimization of a well-defined objective and has no guarantees to converge in theory. To address this problem, they propose an optimization-based framework for consistency matching, which yields an EM-like algorithm with some convergence guarantees.

5 Convergence with self-supervised learning

Another direction of research in semi-supervised learning is its convergence with self-supervised learning methods. Self-supervised learning is a class of representation learning

algorithms where the supervision signal can be computed without human annotations. Representations learned using these methods allow quick fine-tuning for downstream tasks. There is a rich literature in this area, and some of the ideas have been borrowed to semi-supervised learning. It is important to note that strong augmentations are also critical in this line of research.

5.1 Methods based on Contrastive Predictive Coding

In [Oord et al., 2018], a novel method for unsupervised representation learning was introduced. It is based on the so called InfoMax principle, which attempts to maximize mutual information between representations of different “views” of the image (usually defined as various patches extracted from the same image). The representation is trained by predicting the representation of the closest patch using a contrastive loss, inspired by ideas from metric learning. The method is called Contrastive Predictive Coding. The representations learned using this method act as high quality features for downstream tasks. The model has many technical details, e.g. using PixelCNN for aggregating representations of patches.

[Henaff et al., 2020] extends this model with several tricks (e.g. layer normalization, random flipping of patches, etc.) and applies it to semi-supervised setups. In particular, they learn an unsupervised representation based on contrastive predictive coding using all images from ImageNet dataset, and then use the labeled subset to train another classifier on top of the learned representations. Note that the classifier is not a linear shallow model, it is another ResNet. This setup is pretty hard to compare with other semi-supervised models. The authors report results with various percentages of labeled examples. There are two commonly used benchmarks for semi-supervised setup for ImageNet: with 1% labeled data (10 samples per class) and with 10% labeled data. CPCv2 is applied to both setups.

This paper was submitted to ICLR 2020, but was rejected. Later, another paper critically analyzed the results obtained using CPC and similar methods based on maximizing mutual information, and concluded that the experimental successes presented in those papers are mostly due to similarities of these methods to deep metric learning (the triplet loss, hard negative mining etc.) and not because of the quality of mutual information maximization [Tschannen et al., 2019]. Later, this direction resulted in SimCLR, a more sophisticated contrastive learning framework [Chen et al., 2020].

5.2 Semi-supervised Self-supervised Learning

Another complicated, multi-stage algorithm was described in [Zhai et al., 2019], published in ICCV 2019. They suggest to integrate self-supervised learning techniques into semi-supervised learning. In particular, they focus on two known self-supervised learning methods:

1. Rotation. Each unlabeled image is rotated by 90, 180 and 270 degrees and along with the original one are given to a classifier which attempts to predict the rotation angle (4-class classification). The classifier has a ResNet backbone, so it learns to extract useful features.
2. Exemplar. Two augmented versions of the same image are passed through the classifier and the learned representations are trained to be similar. Triplet loss is used to avoid collapse of representations.

These methods produce representations without using any labels. The new method suggested in this paper, called semi-supervised self-supervised learning (S4L) adds a regular classification loss term to the loss function, which is computed only for labeled examples. At each iteration, two equal sized batches are sampled: one from the set of labeled examples, another one from the set of unlabeled examples. The loss for rotation prediction or exemplar is computed either on unlabeled batch only, or on both batches. This choice does not affect the final performance of these models.

In the last part of the paper the authors describe a three-stage system which brings state-of-the-art results on ImageNet:

1. Train a semi-supervised model using Virtual Adversarial Training (along with entropy minimisation) with an additional classifier to predict rotation of the image.
2. Use the model obtained from the first stage to generate pseudo labels for all images of ImageNet. The labels are generated by taking the average of predictions across five random crops and four rotations of the same image. Train the same algorithm on the dataset using the predicted labels. Initialize the weights from the network obtained in the first stage, and then train for 18 epochs while decaying the learning rate after 6th and 12th epochs.
3. Fine tune the model obtained in the second stage by using only the original labels. This step is trained with weight decay $3 \cdot 10^{-3}$ and learning rate $5 \cdot 10^{-4}$ for 20 epochs. Learning rate is decayed 10x every 5 epochs.

The resulting model is called MOAM (mix of all models). The number of design choices made in MOAM make it impractical to use for other datasets.

5.3 CoMatch

In CoMatch [Junnan Li, 2020b], the ideas of consistency regularization, entropy minimization and contrastive learning converge into a single architecture. Additionally, CoMatch integrates ideas of graph-based SSL. The authors interpret contrastive learning as a form of class-agnostic consistency regularization, which enforces the same image with different augmentations to have similar embeddings, while different images have different embeddings.

CoMatch jointly learns the encoder $f(\cdot)$, the classification head $h(\cdot)$, and the projection head $g(\cdot)$. Each image has two compact representations: a class probability p produced by the classification head and a low-dimensional embedding z produced by the projection head. Each of these two representations is trained with the help of the other. The classification head is trained using ground-truth labels and memory-smoothed pseudo-labels of (weakly supervised versions of) unlabeled images. Pseudo-labels are smoothed by aggregating information from nearby samples in the embedding space. The projection head is trained using contrastive learning on a pseudo-label graph W_q , where samples with similar pseudo labels are encouraged to have similar embeddings. Pseudo-label graph W_q is the target to train an embedding graph W_z . Embeddings are computed on strongly augmented versions of the images.

CoMatch jointly optimizes three losses: a supervised classification loss on labeled data L_x , an unsupervised classification loss on unlabeled data, which is defined as the cross-entropy between the pseudo-labels q_b and the model's predictions, and a graph-based contrastive loss on unlabeled data L_u^{ctr} .

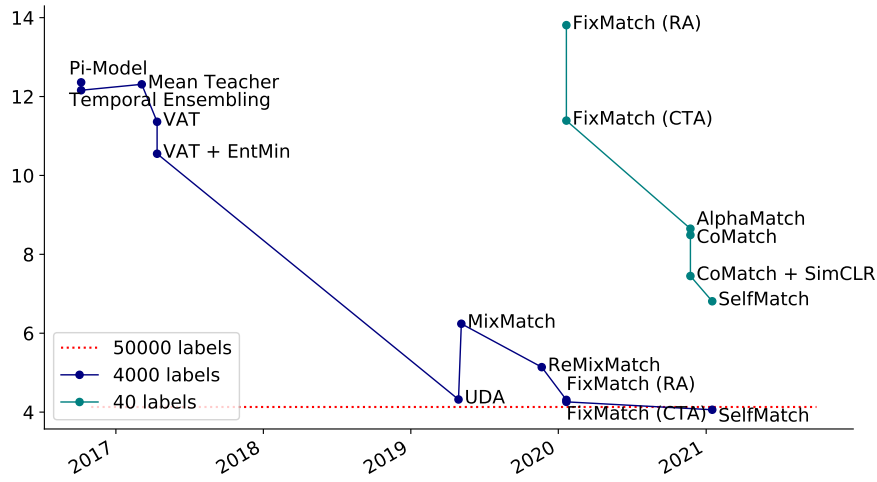


Figure 1: Error rate on CIFAR-10 with 4000 and 40 labeled examples over time.

$$L = L_x + \lambda_{cls} L_u^{cls} + \lambda_{ctr} L_u^{ctr}$$

$$L_u^{cls} = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \mathbb{1}(\max q_b \geq \tau) H(q_b, p(y|Aug_s(u_b)))$$

where Aug_s refers to strong augmentations, B is the batch size for labeled images, μB is the batch size for unlabeled images, λ_{cls} and λ_{ctr} are scalar hyperparameters to control the weight of the unsupervised losses. Note that, similar to FixMatch, pseudo-labels with low confidence are not used for supervision. Please refer to the paper for more details. The authors also report the performance of this architecture followed by self-supervised pretraining based on SimCLR.

5.4 SelfMatch

SelfMatch [Byoungjip Kim et al., 2021] is a more simple approach to combine consistency regularization and contrastive self-supervised learning. SelfMatch consists of two stages: self-supervised pre-training based on contrastive learning followed by semi-supervised fine-tuning with consistency regularization. For the first stage, SelfMatch adopts SimCLR [Chen et al., 2020]. For data augmentation in this stage SelfMatch uses random crop and color distortion. The second stage is essentially the FixMatch algorithm with RandAugment for strong augmentations. This approach reaches 4.06% error rate on CIFAR-10 with 4000 labeled examples, compared to 4.13% error rate of a fully supervised baseline.

6 Discussion

6.1 Towards the performance of fully supervised methods

As seen in Table ??, the performance of semi-supervised algorithms on all major benchmarks improved significantly over the past four years.

Method	CIFAR-10	CIFAR-10	SVHN	STL-10
	4k labels	40 labels	1k labels	1k labels
II-Model	12.36 ± 0.31		4.82 ± 0.17	
Temporal ensembling	12.16 ± 0.24		4.42 ± 0.16	
Mean Teacher	12.31 ± 0.28		3.95 ± 0.19	
VAT	11.36 ± 0.34		5.42 ± 0.22	
VAT + EntMin	10.55 ± 0.05		3.86 ± 0.11	
MixMatch	6.24 ± 0.06	47.54 ± 11.50	3.27 ± 0.31	10.18 ± 1.46
ReMixMatch	5.14 ± 0.04	19.10 ± 9.64	2.83 ± 0.30	6.18 ± 1.24
UDA	4.32 ± 0.08	29.05 ± 5.93	2.23 ± 0.07	
FixMatch (RA)	4.26 ± 0.05	13.81 ± 3.37	2.28 ± 0.11	7.98 ± 1.50
FixMatch (CTA)	4.31 ± 0.15	11.39 ± 3.35		5.17 ± 0.63
AlphaMatch		8.65 ± 3.38		
CoMatch		8.49 ± 2.15		
CoMatch + SimCLR		7.45 ± 1.02		
SelfMatch	4.06 ± 0.08	6.81 ± 1.08	2.51 ± 0.07	

Table 1: Error rates of the described algorithms on CIFAR-10, SVHN and STL-10.

Method	Top-1	Top-5
UDA	31.22	11.20
FixMatch	28.54	10.87
CPC v2 (fixed)		9.50
CPC v2 (fine-tuned)		8.80
S4L MOAM	30.27	11.20
S4L MOAM + pseudo label	28.44	10.04
S4L MOAM full	26.79	8.77

Table 2: Error rate of the described algorithms on ImageNet with 10% labels.

Fig. 1 shows the advancement on CIFAR-10 with 4000 and 40 labeled examples over time. Current methods essentially repeat the success of the fully supervised baseline with the same network architecture (50K labeled examples) using just 8% of the labels. On SVHN, which is arguably an easier task, similar performance is achieved with roughly 1.5% of the labels. Note that it is not known whether this success will easily translate to more complicated neural architectures which achieve around 1% error rate on the fully labeled CIFAR-10 (e.g. [Wang et al., 2019]).

For ImageNet, S4L paper reports 5.9% Top-5 (21.43% Top-1) error rate for a fully supervised baseline with a version of ResNet-50 backbone. As seen in Table ??, the described models have yet to reach such performance with 10% of the labels.

6.2 Evaluation challenges

The number of variables in semi-supervised setups is so large that is quite hard to compare different algorithms. In [Oliver et al., 2018], a serious attempt has been made to create a fair comparison setup for CIFAR-10 and SVHN. In particular, the authors:

- Re-implemented the best known methods in a single code repository

- Fixed the backbone classifier network: WideResNet-28-2 with batch normalization and leaky ReLU
- Fixed the optimizer: Adam with fixed β_1 and β_2
- Used fixed data augmentation and preprocessing strategy, (except that SVHN does not use horizontal flips).
- Used equal hyperparameter tuning budget for all algorithms. This is implemented by running 1000 trials of Gaussian-Process-based black box optimization in Google Cloud.

The model selection was performed on the full validation set, which is not a realistic scenario, and it is acknowledged by the authors. Their hyperparameter search resulted in different initial learning rates for Adam optimizer for different methods. Also, in case of VAT, the best value for ϵ (which controls the magnitude of adversarial perturbation) turned out to be different for CIFAR-10 and SVHN.

The authors report the following issues they discovered in their analysis:

1. Fully-supervised baselines trained on limited labels are not tuned correctly in many papers. The authors suggest to use the same budget for tuning the hyperparameters of the fully supervised setup. They discover that with more fair experimental setup, the difference between the supervised baselines and the new algorithms is actually lower. The authors also showed that with much stronger regularization it is possible to reach 13.4% error rate on CIFAR-10 with 4000 labels.
2. Transfer learning from (resized) ImageNet is a strong baseline, and it is ignored in most papers. The best result they got from transfer learning (12.09% error) is better than the best result in semi-supervised learning (13.13% error).
3. All models assume that the distribution of unlabeled examples follows the distribution of labeled examples. It is shown that when this assumption does not hold, using unlabeled examples might hurt the performance.
4. To analyze the role of additional unlabeled examples, the authors use SVHN-Extra dataset and monitor the performance on SVHN given different number of unlabeled examples. They show that some methods get worse performance when exposed to too many unlabeled examples. The authors also analyze the effect of the number of labeled examples.
5. Finally, the authors show that in a more realistic treatment of validation data, when the size of the validation set is just 10% of the (labeled) training set, then it is not feasible to reliably distinguish between strongly and weakly performing models.

This work had some positive impact on further research in semi-supervised learning. Almost all subsequent papers working on CIFAR-10 and SVHN used the same underlying architecture, most authors re-implemented previous best models in the same codebase. Unfortunately, this is still not the case with experiments on ImageNet, experimental setups still vary a lot.

There is less progress with respect to other problems highlighted in this paper. For example, the authors still continue to use full validation sets for model selection and for comparing different algorithms. S4L paper is the only one we could find that devotes a special section on the discussion of this issue.

Transfer learning from larger image datasets is also ignored in most papers. The synergy of transfer learning and semi-supervised learning is not yet explored.

	MixMatch	ReMixMatch	FixMatch (RA)
ReMixMatch paper		84.92	
FixMatch paper	52.46	80.90	86.19
CoMatch paper	52.90		83.02

Table 3: Accuracy of various implementations of the same algorithm on CIFAR-10 with 40 labels.

6.3 Re-implementations of the algorithms

As recommended in [Oliver et al., 2018], several authors have re-implemented previously introduced algorithms in their own codebase, and reported the performance of their versions. This is a good practice, which helps to highlight how different implementations can change the final scores.

We demonstrate this on a setup with a relatively high variance: CIFAR-10 with just 40 labels. This setup was first studied in ReMixMatch paper. Later, the authors of FixMatch algorithm ran the same setup with their own implementation of it, along with MixMatch, which was never tried on this setup before. Finally, CoMatch paper reports the results using their own implementations of MixMatch and FixMatch. The results are reported in Table ?? . Although most papers report the median of five runs, the difference in scores across implementations ranges from 0.5 to 4 percentage points.

6.4 Augmentation strategies

As demonstrated above, data augmentation has a critical role in most semi-supervised learning algorithms. Table 4 shows the evolution of augmentation strategies used for consistency regularization. RandAugment, which was developed for general-purpose image classification task, was the first augmentation which was demonstrated to improve the performance in UDA [Xie et al., 2019]. Many coefficients used in RandAugment have been originally tuned on the full datasets, which was not fair in the SSL setup. CTAugment was developed in the ReMixMatch paper [Berthelot et al., 2019] for the needs of semi-supervised learning. FixMatch showed that CutOut provides significant boost on top of other strong augmentation methods.

6.5 Ignoring examples

One of the challenges in semi-supervised learning algorithm is to identify samples which will harm the training process and ignore them. The first condition under which a sample might be ignored is when there are very few labeled examples and there is a risk that the model will overfit on them. UDA is the only algorithm we know that explicitly ignores those labeled examples for which the network gives predictions with a confidence higher than a threshold. UDA decreases this threshold over time.

The second condition applies to unlabeled images. The prediction of a model on an unlabeled example might be incorrect and can harm the training. Pi-model attempts to solve this by having very small coefficient for the consistency regularization in the early phases of training, and gradually increasing it later. Newer algorithms keep this coefficient fixed. Instead, in FixMatch, the unlabeled images for which the predictions have low confidence are simply ignored. CoMatch uses the same strategy.

Method	Augmentation on labeled examples	Augmentation on unlabeled examples
Π -model	Weak	Weak
Mean Teacher	Weak	Weak
Virtual Adversarial Training	None	Adversarial
UDA	Weak	RandAugment
MixMatch	Weak	MixUp + Weak
ReMixMatch	Weak	MixUp + CTAugment
FixMatch	Weak	RandAugment/CTAugment + CutOut
AlphaMatch	Weak	CTAugment
CoMatch	Weak	RandAugment
SelfMatch	Weak	RandAugment

Table 4: Augmentation strategies of SSL algorithms which include a form of consistency regularization.

AlphaMatch develops the “softer” version of this idea from FixMatch. They show that when α in the alpha-divergence term is greater than 1, the samples with larger confidence get more weight in the training.

7 Applications of semi-supervised learning

The success of the consistency regularization in semi-supervised learning algorithms motivated its applications in other machine learning problems. Here we show two examples.

7.1 Learning with noisy labels (DivideMix)

Supervised learning with a dataset with noisy labels is an interesting problem in machine learning research. DivideMix [Junnan Li, 2020a] is a recently proposed strategy to learn with noisy labels using semi-supervised learning. DivideMix trains two networks in parallel. At the end of each epoch, both networks model their per-sample loss distributions with a Gaussian Mixture Model to divide the dataset into a clean labeled set which is assumed to have correct labels, and an unlabeled set which is assumed to have noisy labels. Each of the networks is trained with the two (labeled and unlabeled) defined by the other network using a slightly modified version of the MixMatch algorithm described above. The split between clean and noisy sets induced by one network is not used by itself to avoid confirmation bias.

7.2 Stabilizing Generative Adversarial Networks (CR-GAN)

Generative Adversarial Networks [Goodfellow et al., 2014a] (GANs) are the dominant method for image generation. A GAN consists of two neural networks: a generator and a discriminator, which are trained together in a minimax game. Discriminator’s role is to detect whether the given image is real (comes from the dataset) or it is generated by Generator. Essentially, discriminator performs binary classification of images. The biggest problem with GANs is the stability of their training. In [Zhang et al., 2020] a remedy

to this problem is proposed based on consistency regularization. The authors propose an additional loss term for the Discriminator's training objective which encourages it to have similar internal representations for an image and its augmented version. They show that this simple change leads to significant performance and stability improvements in many kinds of GANs.

8 Conclusion

In this paper we have reviewed the recent developments in semi-supervised learning algorithms designed for image classification tasks. Most of the methods are based on consistency regularization and actively use various data augmentation strategies. Few methods attempt to combine these with ideas from unsupervised representation learning and metric learning. A significant progress in terms of accuracy on several benchmarks has been observed, which motivated applications of consistency regularization in other tasks. Still most semi-supervised learning methods suffer from instability with respect to hyperparameters and implementation details, and fail when the distribution of unlabeled samples is shifted. We expect the future work in this area to focus on making the methods more stable for increased usage in real-world applications.

Acknowledgements

We would like to thank the reviewers for useful feedback. This work was supported by the RA Science Committee, in the frames of the research project No. 20TTAT-AIa024.

References

- [Berthelot et al., 2019] Berthelot, D., Carlini, N., Cubuk, E. D., Kurakin, A., Sohn, K., Zhang, H., and Raffel, C. (2019). Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. *arXiv preprint arXiv:1911.09785v2 / accepted at ICLR '2020*.
- [Byoungjip Kim et al., 2021] Byoungjip Kim, Jinho Choo, Y.-D. K., Joe, S., Min, S., and Gwon, Y. (2021). Selfmatch: Combining contrastive self-supervision and consistency for semi-supervised learning. *arXiv preprint arXiv:2101.06480*.
- [Chapelle et al., 2006] Chapelle, O., Schölkopf, B., and Zien, A. (2006). *Semi-Supervised Learning*. MIT Press.
- [Chen et al., 2020] Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020). A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*.
- [Chengyue Gong, 2020] Chengyue Gong, Dilin Wang, Q. L. (2020). Alphamatch: Improving consistency for semi-supervised learning with alpha-divergence. *arXiv preprint arXiv:2011.11779*.
- [Coates et al., 2011] Coates, A., Ng, A., and Lee, H. (2011). An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223.
- [Cubuk et al., 2019a] Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V. (2019a). Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 113–123.
- [Cubuk et al., 2019b] Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. (2019b). Randaugment: Practical data augmentation with no separate search. *arXiv preprint arXiv:1909.13719*.

- [Deng et al., 2009] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.
- [DeVries and Taylor, 2017] DeVries, T. and Taylor, G. W. (2017). Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*.
- [Goodfellow et al., 2014a] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014a). Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, pages 2672–2680. Curran Associates, Inc.
- [Goodfellow et al., 2014b] Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014b). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.
- [Henaff et al., 2020] Henaff, O. J., Srinivas, A., Fauw, J. D., Razavi, A., Doersch, C., Eslami, S. M. A., and van den Oord, A. (2020). Data-efficient image recognition with contrastive predictive coding.
- [Junnan Li, 2020a] Junnan Li, Richard Socher, S. C. H. (2020a). Dividemix: Learning with noisy labels as semi-supervised learning.
- [Junnan Li, 2020b] Junnan Li, Caiming Xiong, S. H. (2020b). Comatch: Semi-supervised learning with contrastive graph regularization. *arXiv preprint arXiv:2011.11183*.
- [Krizhevsky et al., 2009] Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.
- [Laine and Aila, 2017] Laine, S. and Aila, T. (2017). Temporal ensembling for semi-supervised learning. *ICLR*.
- [Miyato et al., 2018] Miyato, T., Maeda, S.-i., Koyama, M., and Ishii, S. (2018). Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993.
- [Netzer et al., 2011] Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning.
- [Oliver et al., 2018] Oliver, A., Odena, A., Raffel, C. A., Cubuk, E. D., and Goodfellow, I. (2018). Realistic evaluation of deep semi-supervised learning algorithms. In *Advances in Neural Information Processing Systems*, pages 3235–3246.
- [Oord et al., 2018] Oord, A. v. d., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.
- [Ouali et al., 2020] Ouali, Y., Hudelot, C., and Tami, M. (2020). An overview of deep semi-supervised learning. *arXiv preprint arXiv:2006.05278*.
- [Sohn et al., 2020] Sohn, K., Berthelot, D., Li, C.-L., Zhang, Z., Carlini, N., Cubuk, E. D., Kurakin, A., Zhang, H., and Raffel, C. (2020). Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685v2*.
- [Tarvainen and Valpola, 2017] Tarvainen, A. and Valpola, H. (2017). Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, pages 1195–1204.
- [Tschannen et al., 2019] Tschannen, M., Djolonga, J., Rubenstein, P. K., Gelly, S., and Lucic, M. (2019). On mutual information maximization for representation learning. *arXiv preprint arXiv:1907.13625, accepted at ICLR 2020*.
- [Vanyan and Khachatryan, 2020] Vanyan, A. and Khachatryan, H. (2020). A survey on deep semi-supervised learning algorithms. *Proceedings of Collaborative Technologies and Data Science in Artificial Intelligence Applications Workshop*, pages 123–134.
- [Wang et al., 2019] Wang, L., Xie, S., Li, T., Fonseca, R., and Tian, Y. (2019). Sample-efficient neural architecture search by learning action space. *arXiv preprint arXiv:1906.06832*.

- [Xie et al., 2019] Xie, Q., Dai, Z., Hovy, E., Luong, M.-T., and Le, Q. V. (2019). Unsupervised data augmentation. *arXiv preprint arXiv:1904.12848v6*.
- [Zhai et al., 2019] Zhai, X., Oliver, A., Kolesnikov, A., and Beyer, L. (2019). S4I: Self-supervised semi-supervised learning. In *Proceedings of the IEEE international conference on computer vision*, pages 1476–1485.
- [Zhang et al., 2017] Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. (2017). mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.
- [Zhang et al., 2020] Zhang, H., Zhang, Z., Odena, A., and Lee, H. (2020). Consistency regularization for generative adversarial networks. In *International Conference on Learning Representations*.
- [Zhu and Goldberg, 2009] Zhu, X. and Goldberg, A. B. (2009). Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130.