Realistic Evaluation of Deep Semi-Supervised Learning Algorithms
(2018)

Avital Oliver, Augustus Odena, Colin Raffel, Ekin D. Cubuk, Ian J. Goodfellow
Resume

May 3, 2019

1 Introduction

The goal of this paper is to propose new ways of evaluating the semi supervised approaches in a way that reflect the real world applications of semi supervised learning (SSL), these measures are:

- having equal hyperparameter tuning for both SSL and only using supervised labels,
- A model with carefully chosen parameters and regularization in a SSL setting can even do better when using labeled examples only, so it is important to use the same model for different evaluations
- taking into consideration the pretraining of our classifiers on different labeled dataset, given that it can give better results only using a supervised method,
- Taking into consideration the possibility of having different distribution of classes between the labels and unlabeled examples, which can degrate the SSL performances quite significantly,
- Only a small dataset for validation would be used in a real setting given the abondance of labeled examples

2 Imroved Evaluation

2.1 A shared implementation

To correctly evaluate and compare the semi supervised methods, we must have a shared impelementation using, in addition to the same dataset, the same base model and the same details (parameter initialization, data processing, augmentation, regularization, ...), which was not the case in earlier approaches.

2.2 High quality supervised base line

The goal of SSL is to obtain better performance using the combination of $\mathcal{D}$ and $\mathcal{D}_{\mathcal{UL}}$ than what would be obtained with $\mathcal{D}$ alone, so a possible base line is to train the same model in the fully supervised setting using only $\mathcal{D}$ and use it as a baseline.

2.3 Comparaison to transfer learning

An other base line to compare against is to use a pretrained model on some of the publicily availbe dataset and fine tune it on $\mathcal{D}$ and compare it the SSL approaches.

2.4 Considering class distribution mismach

In real world, we might want to augment the labeled data $\mathcal{D}$, and it is quite likely that these new and unlabeled images will come from the same underling distribution, so this affect of differing ditributions from labeled and unlabeld data must be studied.

2.5 Realistic small validation sets

In SSL we compare the perforances of the model in a validation set that is larger than the training set, which is not the case in real applications, so we must do hyperparameters tuning on a smaller dataset and that is reflective on the real world usage, if we want to have an approximation of the validation error with a given number of sample we can use the Hoeffdings inequality:

$$\mathbf{P}(|\overline{V} - \mathbb{E}[V]| < p) > 1 - 2\exp\left(-2np^2\right)$$

Say if wa want to be 95% confident that the different between our validation error $\overline{V}$ and the true error $\mathbb{E}[V]$ is less that $p = 1\%$, we'll need $20,000 = n$ examples.

3 Semi supervised methods

In supervised learning, we are given a training dataset of input-target pairs $(x, y) \in \mathcal{D}$ sampled from an unknown joint distribution $p(x, y)$. Our goal is to produce a prediction function $f(\theta)$ parametrized by θ which produces the correct target y for previously unseen samples from $p(x)$. For example, choosing θ might amount to optimizing a loss function which reflects the extent to which $f_\theta(x) = y$ for $(x, y) \in \mathcal{D}$. In SSL we are additionally given a collection of unlabeled input datapoints $x \in \mathcal{D}_{UL}$, sampled from $p(x)$. We hope to leverage the data from $\mathcal{D}_{UL}$ to produce a prediction function which is more accurate than what would have been obtained by using $\mathcal{D}$ its own.

From a broad perspective, the goal of SSL is to use $\mathcal{D}_{UL}$ to augment $f(\theta)$ with information aboutthe structure of $p(x)$. For example, $\mathcal{D}_{UL}$ can provide hints about the shape of the data manifold which can produce a better estimate of the decision boundary between different possible target values.

3.1 Consistency Regularization based methods

Consistency regularization describes a class of methods with following goal: add a realistic perturbations to the input so that $x \to \hat{x}$ and where x is unlabeled ($x \in \mathcal{D}_{UL}$) and this should not significantly change the output of $f_\theta(x)$. we then train the model to minimise $d\left(f_\theta(x), f_\theta(\hat{x})\right)$ where $d(.,.)$ measure the distance between the clean and perturbed predictions, it can be MSE or KL-divergence.

$\prod$-Model The simplest setting in which to apply consistency regularization is when the prediction function $f_\theta(\hat{x}$ is itself stochastic, i.e. it can produce different outputs for the same input x. This is common when $f_\theta(\hat{x}$ is a neural network due to common regularization techniques such as data augmentation, dropout, and adding noise. These regularization techniques themselves are typically designed in sucha way that they ideally should not cause the models prediction to change, and so are a natural fit forconsistency regularization.

Temporal Ensembling/Mean Teacher A difficulty with the $\prod$-Model approach is that it relies on a potentially unstable target prediction, namely the second stochastic network prediction which can rapidly change over the course of training. So to obtain stable target output $\overline{f}_\theta(x)$ for $x \in \mathcal{D}_{UL}$. *Temporal Ensembling* uses an exponentially accumulated average of outputs of $f_\theta(x)$ for the consistency target, and *Mean Teacher* instead proposes to use a prediction function parametrized by an exponentially accumulated average of θ over training.

Virtual Adversarial Training Instead of relying on the built-in stochasticity of $f_\theta(x)$, Virtual Adversarial Training (VAT) directly approximates a tiny perturbation r_{add} to add to x which would most significantly affect theoutput of the prediction function. An approximation to this perturbation can be computed efficiently as:

$$r \sim \mathcal{N}\left(0, \frac{\xi}{\sqrt{\dim(x)}} I\right)$$

$$g = \nabla_r d\left(f_\theta(x), f_\theta(x+r)\right)$$

$$r_{adv} = \epsilon \frac{g}{\|g\|}$$

3.2 Entropy-Based

A simple loss term which can be applied to unlabeled data is to encourage the network to make confident (low-entropy) predictions for all examples, regardless of the actual class predicted. Assuming a categorical output space with K possible classes (e.g. a K-dimensional softmax output), this gives rise to the entropy minimization term:

$$-\sum_{k=1}^{K} f_\theta(x)_k \log f_\theta(x)_k$$

On its own, entropy minimization has not been shown to produce competitive results compared to the other methods described, However, entropy minimization was combined with VAT to obtain state-of-the-art results

3.3 Pseudo-Labeling

Pseudo-labeling is a simple heuristic which is widely used in practice because of its simplicity and generality, all that it requires is that the model provides a probability value for each of the possible labels. It proceeds by producing pseudo-labels for $\mathcal{D}_{UL}$ using the prediction function itself over the course of training. Pseudo-labels which have a corresponding class probability which is larger than a predefined threshold are used as targets for a standard supervised loss function applied to $\mathcal{D}_{UL}$.