Pseudo-Label : The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks

Dong-Hyun Lee SAYIT78@GMAIL.COM
Nangman Computing, 117D Garden five Tools, Munjeong-dong Songpa-gu, Seoul, Korea

Abstract

We propose the simple and efficient method of semi-supervised learning for deep neural networks. Basically, the proposed network is trained in a supervised fashion with labeled and unlabeled data simultaneously. For unlabeled data, *Pseudo-Label*s, just picking up the class which has the maximum network output, are used as if they were true labels. Without any unsupervised pre-training method, this simple method with dropout shows the state-of-the-art performance.

1. Introduction

Recently, deep neural networks have achieved great success in hard AI tasks (Bengio et al., 2012). All of the successful methods for training deep neural networks have something in common : they rely on an unsupervised learning algorithm along with supervised learning of the whole network (Erhan et al., 2010). Most work in two phases. In a first phase, *unsupervised pre-training*, the weights of all layers are initialized by layer-wise unsupervised training. In a second phase, *fine-tuning*, the weights are trained globally in a supervised fashion. All of these methods also work in a semi-supervised fashion. We have only to use extra unlabeled data for unsupervised pre-training.

Several authors have recently proposed semi-supervised learning methods for training supervised and unsupervised tasks using same neural network *simultaneously*. In (Ranzato et al., 2008), the weights of each layer are trained by minimizing the combined loss function of an autoencoder and a classifier. In (Larochelle et al., 2008), Discriminative Restricted Boltzmann Machines models the joint distribution of an input vector and the target class. In (Weston et al., 2008), the weights of all layers are trained by minimizing the combined loss function of a global supervised task and a semi-supervised embedding as a regularizer.

In this article we propose the simpler way of training neural network in a semi-supervised fashion. Basically, the proposed network is trained in a supervised fashion with labeled and unlabeled data simultaneously. For unlabeled data, *Pseudo-Label*s, just picking up the class which has the maximum network output every weights update, are used as if they were true labels. In principle, this method can combine almost all neural network models and training methods. Especially, dropout technique (Hinton et al., 2012) can boost up model performance even for unlabeled data.

Several experiments on the well-known MNIST dataset prove that the proposed method shows the state-of-the-art performance. And this method earned second prize in ”ICML 2013 Challenges in Representation Learning: The Black Box Learning Challenge”.

2. Pseudo-Label for Deep Neural Networks

2.1. Deep Neural Networks

Pseudo-Label is the method for training deep neural networks in a semi-supervised fashion. In this article we will consider multi-layer neural networks with M layers of hidden units :

$$h_i^k = s^k\left(\sum_{j=1}^{d^k} W_{ij}^k h_j^{k-1} + b_i^k\right), \quad k = 1, ..., M+1 \quad (1)$$

where s^k is a non-linear activation function such as sigmoid, $f_i = h_i^{M+1}$ are output units used for predicting target class and $x_j = h_j^0$ are input values. The whole network can be trained by minimizing supervised loss function

$$\sum_{i=1}^{C} L(y_i, f_i(x)), \quad (2)$$

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28. Copyright 2013 by the author(s).

where C is the number of labels, y_i's is the 1-of-K code of the label, f_i is the network output for i'th label, x is input vector.

If the activation function of output units (s^{M+1}) is sigmoid, the loss function can be *Cross Entropy* :

$$L(y_i, f_i) = -y_i \log f_i - (1 - y_i) \log(1 - f_i) \qquad (3)$$

Rectified Linear Unit is receiving a great deal of attention recently (Glorot et al., 2011). This unit uses rectifier activation function :

$$s(x) = \max(0, x) \qquad (4)$$

This is biologically plausable more than sigmoid and hyperbolic tangent. Because rectifier network gives rise to real zeros of hidden activations and thus truly sparse representations, it can boost up the network performance.

2.2. Dropout

Dropout is a technique that can be applied to supervised learning of deep neural networks (Hinton et al., 2012) . On the network activations of each example, hidden unit is randomly omitted with a probability of 0.5. (Sometimes 20% dropout of visible units is also helpful.)

$$h_i^k = drop\left(s^k\left(\sum_{j=1}^{d^k} W_{ij}^k h_j^{k-1} + b_i^k\right)\right), \quad k = 1, ..., M \qquad (5)$$

where $drop(x) = 0$ with a probability of 0.5, otherwise $drop(x) = x$. Overfitting can be reduced by this technique to prevent complex co-adaptations on hidden representations of training data. Because in each weights update we train a different sub-model by omitting a half of hidden units, this training precedure is similar to bagging (Breiman, 1996), where many different networks are trained on different subsets of the data. But dropout is different from bagging in that all of the sub-models share same weights.

For successful SGD training with dropout, An exponentially decaying learning rate is used that starts at a high value. And momentum is used to speed up training.

$$\Delta W(t+1) = p(t)\Delta W(t) - (1 - p(t))\, \epsilon(t) < \nabla_W L > \qquad (6)$$

$$W(t+1) = W(t) + \Delta W(t) \qquad (7)$$

where,

$$\epsilon(t+1) = k\, \epsilon(t) \qquad (8)$$

$$p(t) = \begin{cases} \frac{t}{T} p_f + \left(1 - \frac{t}{T}\right) p_i & t < T \\ p_f & t \geq T \end{cases} \qquad (9)$$

with $k = 0.998$, $p_i = 0.5$, $p_f = 0.99$, $T = 500$, t is the current epoch, $< \nabla_W L >$ is the gradient of loss function, $\epsilon(0)$ is the initial learning rate. We use these parameters such as original dropout paper (Hinton et al., 2012), but don't use weight regularization.

2.3. Pseudo-Label

Pseudo-Label are target classes for unlabeled data as if they were true labels. We can just pick up the class that has maximum network output for each unlabeled sample.

$$y_i' = \begin{cases} 1 & \text{if } i = \text{argmax}_{i'} f_{i'}(x) \\ 0 & \text{otherwise} \end{cases} \qquad (10)$$

2.4. Training method with labeled and unlabeled data

The key point of this method is simultaneously training of labeled data and unlabeled data. For unlabeled data, *Pseudo-Label* that are re-calculated every weights update are used for the same loss function. But the total number of labeled data and unlabeled data is quite different and the training balance between them is quite important for network performance. So the overall loss function is

$$L = \sum_{m=1}^{n} \sum_{i=1}^{C} L(y_i^m, f_i^m) + \alpha(t) \sum_{m=1}^{n'} \sum_{i=1}^{C} L(y_i'^m, f_i'^m), \qquad (11)$$

where n is the number of mini-batch in labeled data for SGD, n' for unlabeled data, f_i^m is the output units of m's sample in labeled data, y_i^m is the label of that, $f_i'^m$ for unlabeled data, $y_i'^m$ is the pseudo-label of that for unlabeled data, $\alpha(t)$ is a coefficient balancing them.

The proper scheduling of $\alpha(t)$ is very important for network performance. If $\alpha(t)$ is too high, the predicting labels is difficult even for labeled data. Whereas if $\alpha(t)$ is too small, we cannot expect generalization performance. Furthermore, in order that the pseudo-labels of unlabeled data are similar to true labels as much as possible, $\alpha(t)$ must be zero for initial training epochs.

$$\alpha(t) = \begin{cases} 0 & t < T_1 \\ \frac{t - T_1}{T_2 - T_1} \alpha_f & T_1 \leq t < T_2 \\ \alpha_f & T_2 \leq t \end{cases} \qquad (12)$$

with $T_1 = 100$, $T_2 = 600$, $\alpha_f = 0.4$.

3. Why does Pseudo-Label work?

3.1. Contractive Regularization using Saturation Region

Contractive Auto-Encoder (Rifai et al., 2011a) is an unsupervised representation learning algorithm that shows the state-of-the-art performance on image recognition tasks (Rifai et al., 2011b). This algorithm uses Jacobian Penalty term which encourages the mapping to the feature space to be contractive in the neighborhood of the training data. This implies an invariance or robustness of the representation for small variations of the input.

$$\|J_f(x)\|_F^2 = \sum_{i,j}\left(\frac{\partial h_i(x)}{\partial x_j}\right)^2 \tag{13}$$

In the case of sigmoid unit, this penalty term has the following expression :

$$\|J_f(x)\|_F^2 = \sum_{i}(h_i(1-h_i))^2\sum_{j}W_{ij}^2 \tag{14}$$

In other words, the further activations go into saturation region($h_i \to 1$ or $h_i \to 0$), the more the network regularize the Jacobian Penalty term.

Supervised learning using unlabeled data with Pseudo-Label can regularize network in such a way that the activations go into saturation region. This encourages an invariance or robustness of the representation for small variations of the input.

3.2. Low-Density Separation between Classes

The goal of semi-supervised learning is to improve generalization performance using unlabeled data. The *cluster assumption* states that the decision boundary should lie in low-density regions to improve generalization performance (Chapelle et al., 2005).

Recently proposed classificaiton algorithms using manifold learning such as Semi-Supervised Embedding and Manifold Tangent Classifier utilize this assumption. Semi-Supervised Embedding (Weston et al., 2008) uses embedding-based regularizer to improve the generalization performance of deep neural networks. Because neighbors of data points have more similar activations by embedding-based penalty term, It's more likely that data points in a high-density region have the same label. Manifold Tangent Classifier (Rifai et al., 2011b) encourages the network output to be insensitive to variations in the directions of low-dimensional manifold. The same purpose is achieved.

Our method encourages the network output to be near 1-of-K code of labels. Because of robustness of the representation for small variations of the input, neighbors of data points have more similar outputs near 1-of-K code of label. So it's more likely that data points in a high-density region have the same label.

Table 1. Classification error on the MNIST test set with 600, 1000 and 3000 labeled training samples. We compare our method with results from (Weston et al., 2008; Rifai et al., 2011b)

METHOD	600	1000	3000
NN	11.44	10.7	6.04
CNN	7.68	6.45	3.35
SVM	8.85	7.77	4.21
TSVM	6.16	5.38	3.45
CAE	6.3	4.77	3.22
DBN-rNCA	8.7	-	3.3
EmbedNN	5.97	5.73	3.59
MTC	5.13	**3.64**	**2.57**
Pseudo-Label	**4.96** ± 0.14	4.28 ± 0.20	2.91 ± 0.09

4. Experiments

4.1. Handwriting Digit Recognition(MNIST)

MNIST is one of the most famous dataset in deep learning literature. For comparision, We used semi-supervised setting of MNIST such as (Weston et al., 2008; Rifai et al., 2011b). We reduced the size of the labeled training set to 600, 1000 and 3000.[1] The training set has the same number of samples on each label. For validation set, We picked up 1000 labeled examples separately. We used validation set for determining some hyperparameters. The remaining data were used for unlabeled data. Because we could not get the same split of data set, Several experiments on random split were done using the identical network and parameters.

We used the neural network with 1 hidden layer. Rectified Linear Unit is used for hidden unit, Sigmoid Unit is used for output unit. The number of hidden units is 5000. For optimization, We used mini-batch Stocastic Gradient Descent with dropout.[2] The initial learning rate is 1.5 and the number of mini-batch is 32 for labeled data, 256 for unlabeled data. These parameter was determined using validation set.

Table 1 compares our method with results from (Weston et al., 2008; Rifai et al., 2011b). Our method

[1] We omitted the case of 100 labeled training set because the results heavily depended on split.

[2] We didn't use any weight regularization because the performance was best without it.

shows the state-of-the-art performance although our method is very simple : We use only 1 hidden layer without pre-training. The training scheme is less complex than Manifold Tangent Classifier and computationally expensive similarity matrix between samples used in Semi-Supervised Embedding is not needed.

5. Conclusion

In this work, we have shown a simple and efficient way of semi-supervised learning for neural networks. Without unsupervised pre-training and computationally expensive similarity matrix, The proposed method shows the state-of-the-art performance.

Acknowledgments

References

Bengio, Y., Courville, A., and Vincent, P. Representation learning: A review and new perspectives. *arXiv preprint arXiv:1206.5538*, 2012.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P. A., Vincent, P., and Bengio, S. Why does unsupervised pre-training help deep learning?. *The Journal of Machine Learning Research*, 2010, 11: 625-660.

Ranzato, M., and Szummer, M. Semi-supervised learning of compact document representations with deep networks. In *Proceedings of the 25th international conference on Machine learning.* ACM, 2008. p. 792-799.

Larochelle, H., and Bengio, Y. Classification using discriminative restricted Boltzmann machines. In *Proceedings of the 25th international conference on Machine learning.* ACM, 2008. p. 536-543.

Weston, J., Ratle, F., and Collobert, R. Deep learning via semi-supervised embedding. In *Proceedings of the 25th international conference on Machine learning.* ACM, 2008. p. 1168-1175.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

Glorot, X., Bordes, A., and Bengio, Y. Deep Sparse Rectifier Networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics. JMLR W&CP Volume.* 2011. p. 315-323.

Breiman, L. Bagging predictors. *Machine learning.* 1996, 24.2: 123-140.

Rifai, S., Vincent, P., Muller, X., Glorot, X., and Bengio, Y. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11).* 2011. p. 833-840.

Rifai, S., Dauphin, Y., Vincent, P., Bengio, Y., and Muller, X. The manifold tangent classifier. *Advances in Neural Information Processing Systems*, 2011, 24: 2294-2302.

Chapelle, O., and Zien, A. Semi-supervised classication by low density separation. *AISTATS*, 2005, (pp. 5764).