Peer-to-Peer Networking and Applications
https://doi.org/10.1007/s12083-018-0702-9

Naive semi-supervised deep learning using pseudo-label

Zhun Li[1] · **ByungSoo Ko**[1] · **Ho-Jin Choi**[1]

Received: 19 February 2018 / Accepted: 19 November 2018

Abstract

To facilitate the utilization of large-scale unlabeled data, we propose a simple and effective method for semi-supervised deep learning that improves upon the performance of the deep learning model. First, we train a classifier and use its outputs on unlabeled data as pseudo-labels. Then, we pre-train the deep learning model with the pseudo-labeled data and fine-tune it with the labeled data. The repetition of pseudo-labeling, pre-training, and fine-tuning is called naive semi-supervised deep learning. We apply this method to the MNIST, CIFAR-10, and IMDB data sets, which are each divided into a small labeled data set and a large unlabeled data set by us. Our method achieves significant performance improvements compared to the deep learning model without pre-training. We further analyze the factors that affect our method to provide a better understanding of how to utilize naive semi-supervised deep learning in practical application.

Keywords Deep learning · Semi-supervised learning · Pseudo-label · Pre-training

1 Introduction

It is known that deep learning models, such as convolutional neural networks (CNNs) [19] and long short-term memory networks (LSTMs) [14], have yielded state-of-the-art results for many complex tasks. One of the key factors behind these results is the massive number of labeled data sets that can be used to optimize tens of millions of parameters. It is becoming easier to collect large-scale unlabeled data thanks to the rapid development of data collection and storage technology. However, obtaining a massive amount of labeled data is still expensive and time consuming. For example, it took more than seven years (2010-2017) to construct the famous image data set ImageNet [32], which contains 14 million images by far.[1] By contrast, users of Facebook upload hundreds of millions [2] photos per day. As obtaining these labeled data may require considerable manpower and material resources, deep learning researchers have to wait a long time before they gather enough data to train a useful model, leading to the research question: How can we utilize these unlabeled data when we only have small amount of training data? Solving this problem is becoming increasingly profitable, especially in the deep learning domain.

Improving the performance of supervised learning tasks using unlabeled data is one of the main topics of semi-supervised learning. The key is to make use of large-scale unlabeled data to provide auxiliary training. Much research exists on semi-supervised learning in the deep learning domain. One set of approaches makes use of unlabeled and labeled data simultaneously. The proposed models are trained to minimize modified loss functions, which usually contain both unsupervised and supervised terms [16, 21, 24, 29, 30, 33, 34, 38, 40]. Another set of approaches utilizes unlabeled and labeled data separately. The proposed models are first pre-trained in an unsupervised manner [10, 15, 28]

This article is part of the Topical Collection: *Special Issue on Big Data and Smart Computing in Network Systems*
Guest Editors: Jiming Chen, Kaoru Ota, Lu Wang, and Jianping He

✉ Ho-Jin Choi
hojinc@kaist.ac.kr

[1] Knowledge Engineering and Collective Intelligence Lab, School of Computing, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Republic of Korea

[1] http://image-net.org

[2] According to Facebook 2017 Annual Report, the daily active users (DAUs) of Facebook were about 1.40 billion. Assuming that one in ten users upload photos per day, users of Facebook upload at least 140 million photos per day.

Published online: 10 December 2018

to learn features, then use the features for supervised purposes. Some early studies conduct layer-wise pre-training [3, 13].

Previous research has focused mainly on improving the deep learning model itself without leveraging other machine learning models. In the case of a small sample, some machine learning models perform better than deep learning models. For example, in Section 4 of this paper, it is shown that Support Vector Machine and Random Forest perform better than CNN with only 100 training data. Encouraged by these results, we study the efficacy of utilizing other machine learning models to improve the deep learning model in a semi-supervised learning fashion. We propose a simple and novel method of utilizing unlabeled data for semi-supervised learning. The primary procedure of our proposed method comprises the following four steps:

1. Train a classifier (pseudo-labeling model) with labeled data.
2. Predict labels (pseudo-label) for unlabeled data using the classifier.
3. Pre-train the deep learning model with pseudo-labeled data from the previous step.
4. Fine-tune the model with labeled data.

Compared with other semi-supervised deep learning approaches, our method has the following advantages:

1. It leverages the advantage of other machine learning models or even the domain expertise, which can be transformed into the label of unlabeled data, to solve small-sample problems.
2. It is easy to implement. We construct a classifier first and feed its outputs on unlabeled data to the deep learning model. Therefore, there is no need to design a loss function or conduct layer-wise training.
3. It can be combined with any deep learning model. We use pseudo-labels as if they were true labels. In theory, they can be used by any supervised learning model.

We conduct a large number of experiments covering different data sets, several sizes of data, and various pseudo-labeling and deep learning models. We show through experiments that our proposed method can improve upon the performance of the deep learning model significantly. Furthermore, we discuss several factors that may affect the semi-supervised performance, including the pseudo-labeling model accuracy on the unlabeled data set (pseudo-labeling accuracy), the unlabeled data size and the number of repetitions of our proposed procedure.

In Section 2, we introduce related work on semi-supervised deep learning. In Section 3, we describe our proposed method. We evaluate our method in Section 4. We show the performance of our method on the Mixed National Institute of Standards and Technology (MNIST) data set [20] in Section 4.1, the Canadian for Advanced research (CIFAR)-10 data set [17] in Section 4.2, and the Internet Movie Database (IMDB) data set in Section 4.3. We present our conclusions in Section 5.

2 Related work

There are many methods for semi-supervised learning, we only focus on semi-supervised classification related to deep learning. In this section, we survey semi-supervised learning methods for deep neural networks and introduce studies similar to our paper.

There is an extensive literature in semi-supervised learning field, see [41, 42] for a survey in semi-supervised learning. With the development of deep learning, most of widely used semi-supervised learning methods have been combined with deep neural networks. One set of semi-supervised deep learning methods is based on generative models, including Variational Autoencoders [16, 23] and Generative Adversarial Networks [7, 8, 25, 35, 36]. Compared to these models, our approach is discriminative and does not need to estimate input distribution. Another set of methods is based on graph-based methods [22, 38]. These methods need to define a similarity of data points by a graph, while our method does not utilize a graph. Multiview learning (e.g. co-training) [1, 6] and disagreement-based semi-supervised learning (e.g. tri-training) [5], which can be seen as a variant of multiview learning, are also combined with deep learning. Our approach is different from these methods, as it requires neither the existence of multiple views nor construction of diverse modules. Another set of methods combines deep learning with entropy regularization (e.g. S3VMs) [12]. Our approach is founded on the same principle with entropy regularization. It differs from standard entropy regularization in two important ways. First, we do not need to choose the balancing hyper-parameter λ, as we train model in an alternating iterative manner. Second, our approach is a wrapping algorithm and can connect any learner to deep learning model without changing their inner workings. There are also many researchers propose novel regularizers utilizing perturbation-based methods [2, 24, 27, 30, 34] and ensembling-based methods [18, 39]. Our approach differs from these methods by leveraging unlabeled data with pseudo-labels and wrapping other models in a sequential way.

Among classic semi-supervised methods, self-training is the simplest algorithm for semi-supervised learning. The main idea is that train the model on the labeled data first and use it to predict the labels for the unlabeled data. Then, re-train the model on the augmented set of labeled data and unlabeled data with predicted labels, and the procedure repeats. Although self-training is simple to use,

it is prone to error by reinforcing poor predictions. To our knowledge, there is little research on self-training based semi-supervised deep learning. Our approach keeps the advantages of self-training, while differs from it in three ways. First, we do not only teach model by itself, but also utilize a good classifier for the first round pseudo-labeling. Second, we do not mix the unlabeled data with the labeled data, but keep them separate and use them in a alternate way. This is very important to prevent the model from reinforcing itself by generating incorrectly labeled data. Third, we do not re-train models from scratch, but train them in a fine-tuning manner.

Lee [21] proposes a simple semi-supervised learning method for deep neural networks using pseudo-label. The proposition is to train deep neural networks in a supervised manner with labeled and unlabeled data concurrently. The author uses a denoising auto-encoder to initialize a deep neural network in the unsupervised pre-training phase. In the fine-tuning phase, the network is trained with labeled and unlabeled data simultaneously, where unlabeled data are labeled with a class that has a maximum predicted probability. This method relies on a trade-off coefficient to balance the labeled data and unlabeled data. The author proposes an empirical function, by which the coefficient is slowly increased, to help the optimization process. Although our approach utilizes the same concept of pseudo-label, it is obviously different from this work in the following ways. First, we use labeled data and unlabeled data alternately and do not need to choose a proper scheduling of the trade-off coefficient. Second, our approach is a wrapping algorithm, which means the choice of learner is completely open. This is important for many real world tasks, where the learners can be complicated black boxes and can not be changed arbitrarily.

3 Method

3.1 Pseudo-labeling, pre-training, and fine-tuning

One of the easiest ways to use unlabeled data is to generate labels for them automatically. Suppose we have a small labeled data set $\mathcal{L} = \{(x_1^l, y_1^l), ..., (x_{n_l}^l, y_{n_l}^l)\}$ and a large unlabeled data set $\mathcal{U} = \{x_1^u, ..., x_{n_u}^u\}$, where $n_u \gg n_l$. The key idea is that we can train a classifier M_P using labeled data $\mathcal{L}$ and predict the label of unlabeled data $\mathcal{U}$ with the classifier.

In this paper, we call this classifier M_P the pseudo-labeling model. We call the output of the pseudo-labeling model M_P on unlabeled data $\mathcal{U}$ a pseudo-label, which is denoted $\hat{\mathbf{y}} = \{\hat{y}_1^u, ..., \hat{y}_{n_u}^u\}$, where $\hat{y}_i^u = M_P(x_i^u)$. Accordingly, the process of predicting outputs is called pseudo-labeling. In particular, pre-training refers to training models using unlabeled data with pseudo-labels, which are denoted $\hat{\mathcal{U}} = \{(x_1^u, \hat{y}_1^u), ..., (x_{n_u}^u, \hat{y}_{n_u}^u)\}$. The basic procedure of our proposed method comprises four steps.

1. Train pseudo-labeling model M_P with labeled data $\mathcal{L}$.
2. Perform pseudo-labeling for unlabeled data $\mathcal{U}$ with pseudo-labeling model M_P.
3. Pre-train deep learning model M_{NN} with pseudo-labeled data $\hat{\mathcal{U}}$.
4. Fine-tune the deep learning model M_{NN} with labeled data $\mathcal{L}$.

Moreover, the predicted output of some classifiers (deep learning model, Gaussian process classifier, etc.) is probabilistic; that is, each pseudo-label has a corresponding probability. Thus, we can not only use all the pseudo-labeled data together, but also specify a reasonable probability interval and just use pseudo-labels within the interval. Usually, a pseudo-label with a higher probability means a higher pseudo-labeling accuracy. However, the use of only a pseudo-label with a higher probability also means that we have to shrink the size of the pseudo-labeled data.

3.2 Naive semi-supervised deep learning

Supposing that pre-training a deep learning model M_{NN} with pseudo-labeled data can improve the final test accuracy of the model, employing a fine-tuned model M'_{NN} to re-label unlabeled data is a very natural idea to enhance pseudo-labeling. Then, we can repeat pseudo-labeling, pre-training, and fine-tuning until the validation accuracy converges. We summarize this process into Algorithm 1, and name it naive semi-supervised deep learning.

Algorithm 1 Naive semi-supervised deep learning algorithm

```
procedure NAIVESEMI(L, U)
    M_P.fit(L)
    ŷ = M_P.predict(U)
    for i = 0; i < N; i++ do
        Û = [ U, ŷ ]                    ▷ Pseudo-labeling
        M_NN.fit(Û)                     ▷ Pre-training
        M_NN.fit(L)                     ▷ Fine-tuning
        ŷ = M_NN.predict(U)
    end for
end procedure
```

In Algorithm 1, the hyper-parameter N is the number of repetitions. N should be large enough to ensure the convergence of validation accuracy. One important thing is that the M_P is trained from scratch, however, M_{NN} is always trained in a fine-tuning way. To ensure the accuracy converges more stably, it is necessary to add the information of population distribution into the procedure. For example, if the samples come from a multinomial distribution with

the same event probability $p_i, k = 1, 2, ..., C$, where C is the number of classes. We should balance the number of samples in each class of pseudo-labeled data before pre-training the model to ensure that no particular class is over-represented.

3.3 Data augmentation and entropy regularization

In this section, we explain how the unlabeled data help the training in naive semi-supervised deep learning.

3.3.1 Data augmentation

Training deep learning model on a small data set leads to the problem of overfitting. There are many ways to mitigate overfitting, such as dropout and early-stopping. Data augmentation is another widely-used technique to solve this problem. Almost every state-of-the-art deep learning model utilizes data augmentation. Pseudo-labeling can be seen as a kind of data augmentation method on condition that we have a large scale of unlabeled data. Although pseudo-labeled data contain high label noise, they still can improve the generalization capability of the model. Some research [31], including the experimental results in this paper, has shown that deep neural networks are able to generalize after training on massively noisy data.

3.3.2 Entropy regularization

Essentially, naive semi-supervised deep learning is a special case of entropy regularization. We first give a brief introduction about entropy regularization, then clarify the connection between our approach and entropy regularization.

There are some previous works [4, 26] focus on the problem of when unlabeled data are informative. These studies conclude that the information in unlabeled data decreases as classes overlap. Therefore, if we want to utilize unlabeled data to help supervised learning, we need to assume that classes are separated by a low-density area. This low-density separation assumption is also known as cluster assumption, which is widely used in semi-supervised research.

Yves Grandvalet and Yoshua Bengio [12] propose to use Shannon's conditional entropy as a measure of class overlap and use entropy regularization as a means to benefit from unlabeled data in the framework of maximum a posteriori estimation. The learning criterion is formulated as

$$\begin{aligned} C(\theta, \lambda; \mathcal{L}, \mathcal{U}) &= \ell(\theta; \mathcal{L}) - \lambda H(y^u|x^u; \mathcal{U}) \\ &= \sum_{i=1}^{n_l} \ln P(y_i^l|x_i^l; \theta) \\ &\quad + \lambda \sum_{i=1}^{n_u} \sum_{k=1}^{C} P(k|x_i^u; \theta) \ln P(k|x_i^u; \theta) \end{aligned} \tag{1}$$

where θ is the parameter, C is the number of classes, and y^u is the unknown label of unlabeled data. The first term $\ell(\theta; \mathcal{L})$ is conditional log-likelihood, the second term $H(y^u|x^u; \mathcal{U})$ is entropy regularization, and λ is a coefficient balancing these two terms. The whole model can be trained by maximizing the log-likelihood function on labeled data and minimizing the conditional entropy of class probabilities on the unlabeled data.

In naive semi-supervised deep learning, the loss function contains several parts. At the first cycle in Algorithm 1,

$$\begin{aligned} Loss = \sum_{i=1}^{n_l} L_P(y_i^l, M_P(x_i^l)) + \sum_{i=1}^{n_u} L_{NN}(\hat{y}_i^u, M_{NN}(x_i^u)) \\ + \sum_{i=1}^{n_l} L_{NN}(y_i^l, M'_{NN}(x_i^l)) \end{aligned}$$

where the first term is the loss function of pseudo-labeling model, the second term is the loss function of deep learning model in pre-training stage, and the last term is the loss function of deep learning model in fine-tuning stage. Since the second cycle, we use the fine-tuned model M'_{NN} as the pseudo-labeling model M_P. Hence, the loss function is reduced to the following form.

$$Loss = \sum_{i=1}^{n_l} L_{NN}(y_i^l, M'_{NN}(x_i^l)) + \sum_{i=1}^{n_u} L_{NN}(\hat{y}_i^u, M_{NN}(x_i^u))$$

When L_{NN} takes the standard cross-entropy form, this loss function is equivalent to entropy regularization in Eq. 1, with the coefficient $\lambda = 1$. The optimization problem of minimizing $Loss$ is non-convex with respect to parameters θ and in this paper, we solve it through an alternating iterative strategy.

Intuitively, entropy regularization can avoid boundaries in dense regions by encouraging predictability (minimizing entropy). Therefore, it makes the unlabeled data beneficial to the learning process. The balancing coefficient λ is set to one in our approach, however, we can control the two terms by adjusting the number of updates. In our experiments, we propose to use early-stopping to trade-off labeled and unlabeled data automatically.

4 Experiment

In order to prove the effectiveness of our proposed method, we conduct experiments with the MNIST, CIFAR-10, and IMDB data sets. We also show that our proposed method can be easily used with various models and data sets.

With the MNIST data set, we compare different pseudo-labeling models and give an insight into the performance

of pseudo-labeling, pre-training, and fine-tuning. We repeat the experiments and make claims about the statistical significance of comparisons. We also analyze key factors that affect our proposed method, such as pseudo-labeling accuracy, the size of unlabeled data, and the number of repetitions. With the CIFAR-10 and IMDB data sets, our focus is exclusively on the performance of naive semi-supervised deep learning. In our experiments, we mainly focus on small sample cases; therefore, we need to pay particular attention to avoiding overfitting. We exploit dropout and early stopping as well as the relative simple deep learning models.

4.1 CNN on MNIST

We randomly choose 10, 50, 100, and 300 samples for each class from the standard 60,000 training samples as the training set and the remaining data as unlabeled data. For training the CNN model, we randomly choose 20% from the training set as the validation set, which is used for early stopping. We use the standard 10,000 test samples as a hold-out test set.

Intuitively, we should choose the best pseudo-labeling model according to the small labeled data set at hand. To demonstrate the effect of using different pseudo-labeling models, we intentionally exploit several models for pseudo-labeling. To the best of our knowledge, CNN [37] achieves the best accuracy on the MNIST data set, while Support Vector Machine (SVM) [9] also gives competitive results. [11] evaluates 179 classifiers using 121 data sets, and concludes that Random Forest (RF) and SVM with Gaussian kernel are clearly better than the remaining ones. Therefore, in our experiments, the pseudo-labeling model includes a Support Vector Machine, a Random Forest, and the deep learning model itself, as well as random and ground truth labeling. The hyper-parameters of the SVM and RF are selected by grid search using validation data. In order to simplify the experiment, we choose a medium hyper-parameter value for all the labeled set, i.e. Gaussian kernel with $c = 10$ and $gamma = 0.01$ for SVM, 200 trees for RF. We choose a simple CNN model from Keras[3] for the MNIST classification task. The CNN model comprises two convolutional layers and two fully-connected layers. We conduct 2×2 max pooling after the second convolutional layer and use the dropout technique for generalization purposes before the first fully-connected layer. We use a categorical cross-entropy loss function and an Adadelta optimizer. Note that we train the model without any data augmentation.

[3] https://github.com/keras-team/keras/blob/master/examples/mnist_cnn.py

4.1.1 Effectiveness of pre-training

We first conduct a set of experiments to test the effectiveness of our proposed method. For each labeled set, we train four models: CNN, CNN with SVM pseudo-labeling, CNN with RF pseudo-labeling, and CNN with CNN pseudo-labeling. In these experiments, we set the number of repetitions as one. We repeat each experiment thirty times and compute the mean of them as the final accuracy.

As shown in Fig. 1, the classification accuracy of CNN with pseudo-labeling on the test set is significantly higher than that of CNN in all experiments. This means that pre-training CNN with pseudo-labeled data indeed improves the performance of the CNN model against the model trained only with labeled data. Note that when the amount of labeled data is small, the improvement is more significant.

4.1.2 Pseudo-labeling models comparison

In this section, we conduct a series of experiments to compare the effects of different pseudo-labeling models. We then give suggestions for pseudo-labeling model selection. The first method is random labeling. As the MNIST data set contains ten classes, the pseudo-labeling accuracy of the random labeling method is approximately 10%. We pre-train the CNN with this random labeled data and then fine-tune the model with labeled data. This method is used as a lower bound benchmark; the pseudo-labeling accuracy of any effective pseudo-labeling model should be higher than this benchmark. For the upper bound benchmark, we use the ground truth of unlabeled data, which means 100% pseudo-labeling accuracy, for pre-training the CNN and then fine-tune the CNN with labeled data. A better pseudo-labeling model should be closer to this benchmark. The main pseudo-labeling model includes the SVM, RF, and CNN itself. For the case of 10 samples in each class,

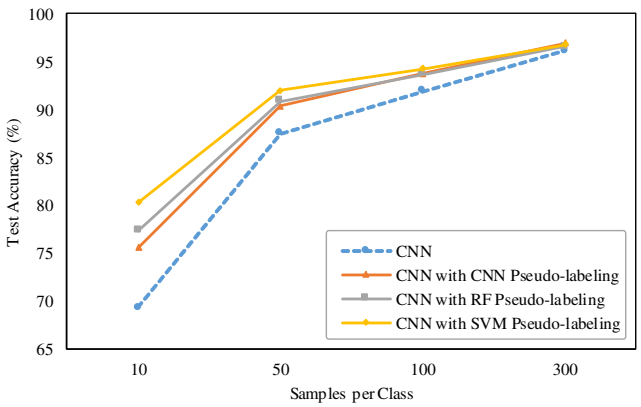

Fig. 1 The accuracy of CNN with pseudo-labeling and CNN with labeled data only on the MNIST test set. Each point represents the mean of the thirty experimental results

Table 1 Comparison of pseudo-labeling models accuracy (± std. dev) (10 Samples per Class)

Pseudo-labeling model	Pseudo-labeling model accuracy on the test set (%)	Pseudo-labeling model accuracy on the unlabeled Set (%)	Pre-trained CNN accuracy on the test set (%)	Fine-tuned CNN accuracy on the test set (%)
Random	—	10 (±0)	11.1 (±2.9)	68.6 (±6.2)
CNN	69.3 (±3.9)	68.5 (±3.6)	70.8 (±4.3)	75.5 (±2.8)
RF	74.8 (±2.1)	73.9 (±1.9)	77.1 (±2.2)	77.3 (±2.3)
SVM	78.6 (±2.1)	77.9 (±2)	80.3 (±2.1)	80.2 (±2.2)
Ground truth	—	100 (±0)	99 (±0.2)	99 (±0.2)

Table 1 shows the comparison results of these pseudo-labeling models. For the situation of 50, 100, and 300 samples in each class, the results are similar.

As shown in Table 1, the test accuracy of the baseline CNN model trained with the total 100 labeled data is 69.3%. RF and SVM perform better than CNN with 74.8% and 78.6% test accuracy, respectively. For all three pseudo-labeling models, pseudo-labeling accuracy is a little lower than the corresponding test accuracy.

We then pre-train the CNN model using these pseudo-labeled data and validate the classification accuracy on the test data. We observe that when pseudo-labeling accuracy is very low, such as with random labeling, we cannot improve the performance of CNN by pre-training. However, if the pseudo-labeling accuracy is sufficiently high, pre-training with pseudo-labeled data leads to better performance. As shown in Table 1, the test accuracies of the baseline CNN, RF, and SVM are 69.3%, 74.8%, and 78.6%, respectively. The test accuracies of the CNN model pre-trained with CNN, RF, and SVM pseudo-labeling increase with different degrees. The test accuracies of the fine-tuned CNN model are 75.5%, 77.3%, and 80.2%, respectively, which are significantly higher than the test accuracy of the corresponding pseudo-labeling models (pairwise t-test p-value $< 4.1 \times 10^{-16}$) and baseline CNN model (t-test p-value $< 2.5 \times 10^{-9}$).

We have three observations based on the results in Table 1. Firstly, the fine-tuned CNN model is at least as good as the baseline CNN model. CNN with CNN, RF, and SVM pseudo-labeling are significantly better than the baseline CNN model. The performance of CNN with random labeling is statistically the same as the baseline model (p-value ≈ 0.70). Secondly, the fine-tuned CNN model performs better than the pseudo-labeling model. Although the performance improvements are different, the test accuracy of fine-tuned CNN models are significantly higher than that of the pseudo-labeling model. Last, a better pseudo-labeling model leads to a better fine-tuned model. Compared to CNN with CNN pseudo-labeling, CNN with RF or SVM pseudo-labeling improves the test accuracy more significantly.

It seems that deep learning models can take advantage of other classifiers and perform even better. In practice, we should choose the best classifier as the pseudo-labeling model according to the specific problem at hand.

4.1.3 Naive semi-supervised CNN

Previous experiments have shown the effectiveness of pre-training with pseudo-labeled data. In this section, we test the performance of naive semi-supervised deep learning. In our experiments, we set the number of repetitions as 10 and focus on the cases of 10 and 50 samples per class. We repeat each experiment thirty times. One experiment takes less than 10 minutes on a GeForce GTX 1080 GPU.

Table 2 summarizes the results for the performance of naive semi-supervised deep learning with 10 repetitions. As

Table 2 Naive semi-supervised CNN with 10 repetitions on MNIST data set

Fine-tuned model	Accuracy on the test set (± std. dev)(%)	
	10 Samples per class	50 Samples per class
CNN	70.7 (±2.6)	87.2 (±0.9)
CNN with CNN pseudo-labeling	80.7 (±1.9)	93.7 (±0.6)
RF	74.5 (±1.2)	87.9 (±0.6)
CNN with RF pseudo-labeling	80.7 (±1.6)	93.8 (±0.7)
SVM	78.5 (±1.7)	90.2 (±0.5)
CNN with SVM pseudo-labeling	82.4 (±1.9)	94.2 (±0.7)

shown in the table, the test accuracy of CNN with pseudo-labeling is significantly higher than that of the baseline CNN model in both 10 and 50 samples per class cases. In 50 samples per class case, SVM pseudo-labeling improves the test accuracy from 87.2% to 94.2%, which is quite close to the test accuracy (99%) of the CNN model trained with all ground truth data. Moreover, the standard deviation of CNN test accuracy declines significantly.

To illustrate the effectiveness of repetition, we calculate the difference in test accuracy between the CNN with pseudo-labeling and the baseline CNN model in the 10 samples per class case with 1 and 10 repetitions. The comparison results are shown in Fig. 2. We find that repeating the pseudo-labeling, pre-training, and fine-tuning procedure is indeed helpful to improving the model accuracy. However, most of the improvement results from the first repetition. The improvement margins diminish when we conduct more repetitions. We also find that the CNN with CNN pseudo-labeling model gains larger improvement, which means that the importance of repetition increases when we have a weak pseudo-labeling model.

Naive semi-supervised CNN can also improve the robustness of the CNN model due to a small data set. We have shown that CNN with CNN pseudo-labeling decreases the standard deviation of test accuracy from 0.9% to 0.6% in 50 samples per class case with 10 repetitions. To give a concrete illustration about this advantage, we plot all the thirty experimental results in Fig. 3. As shown in the figure, when we train the model with a different training set, the validation accuracy of CNN with CNN pseudo-labeling swings wildly. However, the test accuracy of CNN with CNN pseudo-labeling is relatively stable. By comparing two solid lines in Fig. 3, we observe that the test accuracy of CNN with CNN pseudo-labeling presents higher stability

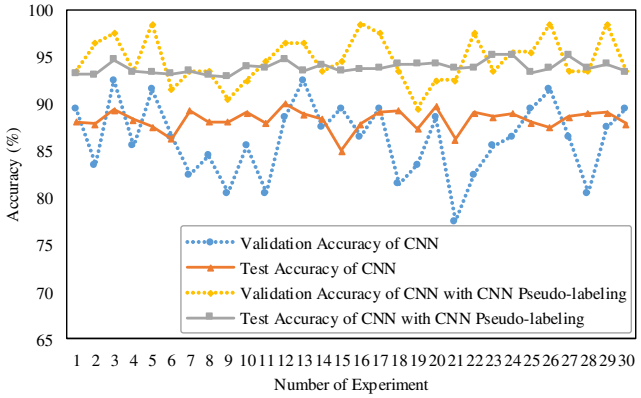

Fig. 3 Robustness improvement of the CNN with CNN pseudo-labeling against the baseline CNN in 50 samples per class case with 10 repetitions

than that of the baseline CNN model. In addition, we also observe that although the validation accuracy swings wildly, its mean value is very close to the mean of test accuracy for both CNN and CNN with CNN pseudo-labeling. This implies that there is no significant overfitting in these experiments.

4.1.4 Factors analysis

This subsection details the analysis of how the pseudo-labeling accuracy and unlabeled data (pseudo-labeled data) size affect pre-training performance, and how to choose the number of repetitions.

Pseudo-labeling accuracy In Section 4.1.2, we observe that better pseudo-labeling model leads to better fine-tuned

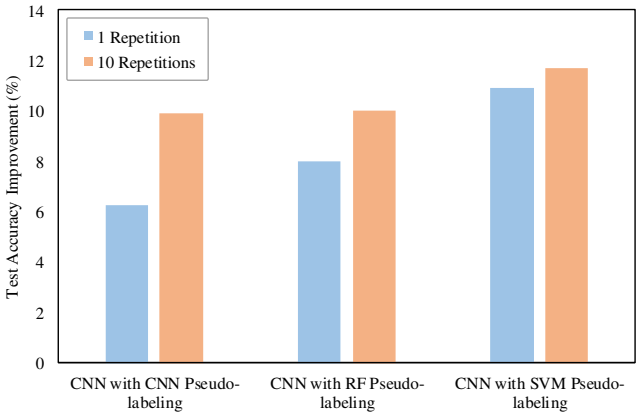

Fig. 2 Test accuracy improvement of the CNN with pseudo-labeling against the baseline CNN in 10 samples per class case. The first value of each pseudo-labeling model is the test accuracy improvement with 1 repetition. The second value is the test accuracy improvement with 10 repetitions

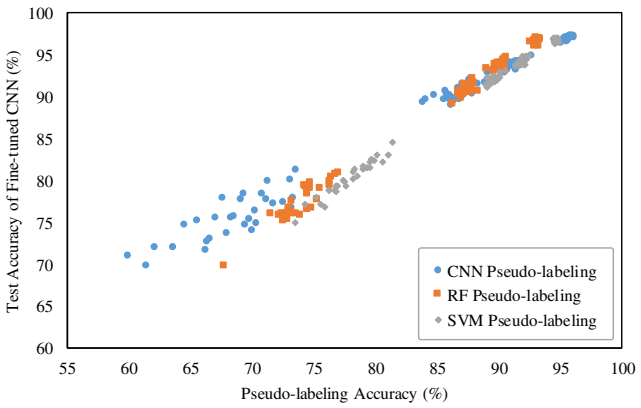

Fig. 4 The relationship between pseudo-labeling accuracy and fine-tuned CNN test accuracy. 360 random experiments contain three pseudo-labeling models (CNN, RF, and SVM) and four sizes of training set (10, 50, 100, and 300 samples per class). We repeat each experiment thirty times

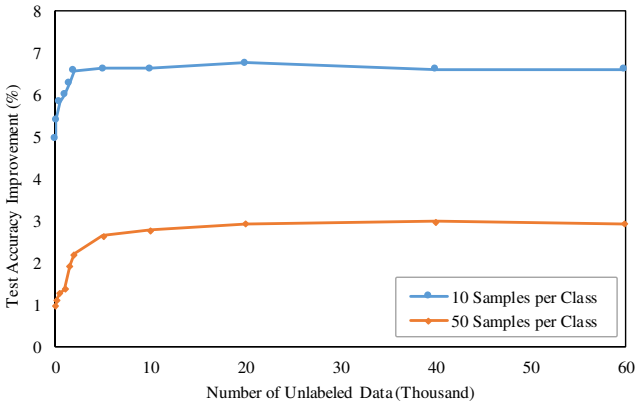

Fig. 5 The relationship between unlabeled data size and fine-tuned CNN test accuracy improvement. Each point represents the average accuracy of thirty experimental results

model. To evidence this, we draw a scatter plot of pseudo-labeling accuracy and fine-tuned CNN test accuracy with 360 random experimental results. As shown in Fig. 4, pseudo-labeling accuracy and fine-tuned CNN test accuracy have a strong positive correlation, no matter for CNN pseudo-labeling, RF pseudo-labeling, or SVM pseudo-labeling.

Pseudo-labeled data size The deep learning model is known to require considerable labeled training data to achieve better performance. This is also true for pseudo-labeled data. As shown in Fig. 5, the test accuracy improvement of the fine-tuned CNN model grows quickly as pseudo-labeled data size increases. However, the improvement margins diminish when we pre-train the CNN with more pseudo-labeled data. Taking the case of 50 samples per class as an example, 20,000 pseudo-labeled data seems enough for pre-training. In practice, unless limited by computational power, we should use all of the unlabeled data to obtain a more robust result.

Fig. 6 One example to illustrate how to choose the number of repetitions. Labeled data includes 50 samples per class. Pseudo-labeled data is balanced before pre-training

Table 3 Naive semi-supervised CNN with 10 repetitions on CIFAR-10 data set

Fine-tuned model	Accuracy on the test set (± std. dev)(%) with n samples per class		
	$n = 500$	$n = 1,000$	All
CNN	53.4 (±2)	60.5 (±1.8)	76.6 (±0.7)
CNN with CNN pseudo-labeling	58.1 (±1.1)	64.9 (±1.3)	—

The number of repetitions To determine the number of repetitions, one way may be to find the point after which the accuracy on the validation set stops growing; this is exactly the idea of early stopping. From Fig. 6, 25 is an option for the number of repetitions. However, given only a few labeled data, accuracy on the validation set could fluctuate dramatically. In that case, an empirical value would be a better choice.

4.2 CNN on CIFAR-10

In this section, we test the performance of naive semi-supervised CNN on the CIFAR-10 data set. The CIFAR-10 data set consists of 60,000 labeled 32×32 color images in 10 classes, with 6000 images per class. There are 50,000 training images and 10,000 test images. Like the MNIST data set, we randomly choose 500 and 1,000 samples from each class of training images as the training set and use the remaining data as unlabeled data. We randomly choose 20% of the training set as the validation set and use the standard 10,000 test images as the test set.

For the CIFAR-10 classification task, deep learning models have given state-of-the-art results. In our experiments, we choose a simple CNN model from Keras [4] and only consider pseudo-labeling with itself. The CNN model comprises four convolutional layers and two fully-connected layers. We train the model with a categorical cross entropy loss function and a root mean square propagation (RMSprop) optimizer, but without data augmentation. We set the number of repetitions as 10 and repeat each experiment ten times. One experiment takes less than 10 minutes on a GeForce GTX 1080 GPU.

The experimental results are summarized in Table 3. The test accuracy of the CNN model with pseudo-labeling is significantly higher than that of the baseline CNN model in both 500 and 1,000 samples per class cases. Improvement of the 500 samples case is a little higher than that of the 1,000 samples case. In addition, the standard deviation of CNN test accuracy is reduced significantly by using our method.

[4] https://github.com/keras-team/keras/blob/master/examples/cifar10_cnn.py

Table 4 Naive semi-supervised LSTM with 5 repetitions on IMDB data set

Fine-tuned model	Accuracy on the test set (± std. dev)(%) with n samples per class		
	$n = 500$	$n = 1,000$	All
LSTM	68.5 (±3)	74.1 (±3)	82.9 (±0.4)
LSTM with LSTM pseudo-labeling	70.9 (±2.1)	75.8 (±1.5)	—

4.3 LSTM on IMDB

IMDB Movie reviews sentiment classification is a binary sentiment classification task in the natural language processing (NLP) domain. The IMDB data set contains 50,000 movies reviews, labeled by positive or negative sentiment. There are 25,000 training reviews and 25,000 test reviews. Reviews have been preprocessed, and each review is encoded as a sequence of word indexes, which means the overall frequency in the data set. For more information, refer to the "Datasets" part of the Keras documentation.[5]

In our experiments, we choose a simple LSTM model from Keras [6] and only consider pseudo-labeling with LSTM in 500 and 1,000 samples per class cases. We set the number of repetitions as 5 and repeat each experiment ten times. One experiment takes less than one hour on a GeForce GTX 1080 GPU.

As shown in Table 4, although the improvement is not as dramatic as with the MNIST and CIFAR-10 experiments, the test accuracy of the LSTM model with pseudo-labeling is higher than that of the baseline LSTM model in both 500 (pairwise t-test p-value $\approx 4.1 \times 10^{-4}$) and 1,000 samples (pairwise t-test p-value ≈ 0.028) per class cases. The experiments indicate that our proposed method can also be applied to the RNN model. Like the results on MNIST and CIFAR-10, the standard deviation of LSTM test accuracy also declines significantly when using naive semi-supervised deep learning.

5 Conclusion

In this paper, we propose a novel method of making use of large-scale unlabeled data to improve deep learning performance. We analyze the underlying principle from data augmentation and entropy regularization aspects. We also exploit various pseudo-labeling models, including SVM, RF, and the deep learning model itself, and conduct experiments to demonstrate the performance of naive semi-supervised deep learning. The experimental results show that the test accuracy of the deep learning model is significantly improved by adding the process of pre-training with pseudo-labeled data. Utilizing the best pseudo-labeling model in the case of small samples leads to the most significant improvement, such as SVM pseudo-labeling on the MNIST data set with only 100 samples. Moreover, we analyze the factors that affect our proposed method, including the pseudo-labeling accuracy and unlabeled data (pseudo-labeled data) size. Then, we give a detailed description on how to choose the number of repetitions. These detailed analyses are helpful to understand and utilize naive semi-supervised deep learning.

In our method, the pseudo-label seems like a kind of "connection" that transfers information from the pseudo-labeling model to the deep learning model. Technically, any kind of methods that can guess a label, such as machine learning classifiers, transfer learning models, clustering methods, rule-based methods, or even domain expertise can be used in our procedure. In other words, naive semi-supervised deep learning can take the advantages of other models and make a better model based on deep learning. This is the biggest difference between our proposed method and currently existing semi-supervised deep learning methods.

Nevertheless, there are several limitations upon naive semi-supervised deep learning. First, obviously, it needs more efforts in selecting the pseudo-labeling model. Second, it may take more time on training the model as our approach uses the labeled and unlabeled data in an alternate way (not parallel). Third, it can not improve performance if the problem structure is not matching with low-density separation assumption. Concretely, naive semi-supervised deep learning assumes that classes are separated by a low-density area. Nonetheless, if the classes are not well-separated, unlabeled data would not be beneficial for the learning. For example, in Section 4.1.2, when the pseudo-labels are random, pre-training with pseudo-labeled data cannot improve model performance. As naive semi-supervised deep learning is a wrapper algorithm, it is hard to analyze in general. Therefore, it is still an open question to find the exact critical conditions to ensure convergence.

In fact, Zhu and Goldberg [42] pointed out that blindly selecting a semi-supervised learning method for a specific task would not necessarily improve performance over supervised learning. Unlabeled data could lead to worse performance with the wrong assumptions. In our case, naïve semi-supervised deep learning may lead to worse performance, if the unlabeled data do not meet the low-density separation assumption. Nevertheless, it is possible to prevent the worse performance by empirical practices. For

[5] https://keras.io/datasets/#imdb-movie-reviews-sentiment-classification

[6] https://github.com/keras-team/keras/blob/master/examples/imdb_lstm.py

example, listed below are some of the guidelines we have found through the work in this paper:

1. Use a validation set to evaluate the effectiveness of naïve semi-supervised deep learning for a specific task.
2. Choose the best classier as the pseudo-labeling model according to the specific problem at hand.
3. Use all of the unlabeled data to obtain a more robust result unless limited by computational power.
4. Balance the number of samples in each class of pseudo-labeled data before pre-training the model to ensure that no particular class is over-represented.

In future, we plan to look for the exact conditions that ensure the convergence of our approach. It would also be interesting to compare our approach with other semi-supervised deep learning methods.

Acknowledgements This work was funded by the Korea Meteorological Administration Research and Development Program under Grant KMI(2017-00410). This research was also supported by Korea Electric Power Corporation. (Grant number:R18XA05)

References

1. Ardehaly EM, Culotta A (2017) Co-training for demographic classification using deep learning from label proportions. In: 2017 IEEE international conference on data mining workshops (ICDMW), IEEE, pp 1017–1024
2. Bachman P, Alsharif O, Precup D (2014) Learning with pseudo-ensembles. In: Advances in neural information processing systems, pp 3365–3373
3. Bengio Y, Lamblin P, Popovici D, Larochelle H (2007) Greedy layer-wise training of deep networks. In: Advances in neural information processing systems, pp 153–160
4. Castelli V, Cover TM (1996) The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing parameter. IEEE Trans Inf Theory 42(6):2102–2117
5. Chen DD, Wang W, Gao W, Zhou ZH (2018) Tri-net for semi-supervised deep learning. In: Proceedings of the Twenty-Seventh international joint conference on artificial intelligence, IJCAI-18, International Joint Conferences on Artificial Intelligence Organization, pp 2014–2020. https://doi.org/10.24963/ijcai.2018/278
6. Cheng Y, Zhao X, Cai R, Li Z, Huang K, Rui Y (2016) Semi-supervised multimodal deep learning for rgb-d object recognition. In: IJCAI, pp 3345–3351
7. Chongxuan L, Xu T, Zhu J, Zhang B (2017) Triple generative adversarial nets. In: Advances in neural information processing systems, pp 4088–4098
8. Dai Z, Yang Z, Yang F, Cohen WW, Salakhutdinov RR (2017) Good semi-supervised learning that requires a bad gan. In: Advances in neural information processing systems, pp 6510–6520
9. Decoste D, Schölkopf B (2002) Training invariant support vector machines. Mach Learn 46(1-3):161–190
10. Dosovitskiy A, Springenberg JT, Riedmiller M, Brox T (2014) Discriminative unsupervised feature learning with convolutional neural networks. In: Advances in neural information processing systems, pp 766–774
11. Fernández-Delgado M, Cernadas E, Barro S, Amorim D (2014) Do we need hundreds of classifiers to solve real world classification problems? J Mach Learn Res 15(1):3133–3181
12. Grandvalet Y, Bengio Y (2006) Entropy regularization. Semi-supervised learning, pp 151–168. https://doi.org/10.7551/mitpress/9780262033589.001.0001
13. Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313(5786):504–507
14. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
15. Jarrett K, Kavukcuoglu K, LeCun Y et al (2009) What is the best multi-stage architecture for object recognition? In: 2009 IEEE 12th international conference on Computer vision, IEEE, pp 2146–2153
16. Kingma DP, Mohamed S, Rezende DJ, Welling M (2014) Semi-supervised learning with deep generative models. In: Advances in neural information processing systems, pp 3581–3589
17. Krizhevsky A, Hinton G. (2009) Learning multiple layers of features from tiny images
18. Laine S, Aila T (2016) Temporal ensembling for semi-supervised learning. arXiv:1610.02242
19. LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324
20. LeCun Y, Cortes C, Burges C. J. (1998) The mnist database of handwritten digits
21. Lee DH (2013) Pseudo-label: the simple and efficient semi-supervised learning method for deep neural networks. In: Work shop on challenges in representation learning, ICML, vol 3, p 2
22. Luo Y, Zhu J, Li M, Ren Y, Zhang B (2017) Smooth neighbors on teacher graphs for semi-supervised learning. arXiv:1711.00258
23. Maaløe L., Sønderby C. K., Sønderby S. K., Winther O. (2016) Auxiliary deep generative models. In: International conference on machine learning, pp 1445–1453
24. Miyato T, Maeda S. I., Koyama M., Ishii S. (2017) Virtual adversarial training: a regularization method for supervised and semi-supervised learning. arXiv:1704.03976
25. Odena A (2016) Semi-supervised learning with generative adversarial networks. arXiv:1606.01583
26. O'neill TJ (1978) Normal discrimination with unclassified observations. J Am Stat Assoc 73(364):821–826
27. Park S, Park JK, Shin SJ, Moon IC (2017) Adversarial dropout for supervised and semi-supervised learning. arXiv:1707.03631
28. Radford A, Metz L, Chintala S (2015) Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv:1511.06434
29. Ranzato M, Szummer M (2008) Semi-supervised learning of compact document representations with deep networks. In: Proceedings of the 25th international conference on machine learning, ACM, pp 792–799
30. Rasmus A, Berglund M, Honkala M, Valpola H, Raiko T (2015) Semi-supervised learning with ladder networks. In: Advances in neural information processing systems, pp 3546–3554
31. Rolnick D, Veit A, Belongie S, Shavit N (2017) Deep learning is robust to massive label noise. arXiv:1705.10694
32. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M et al (2015) Imagenet large scale visual recognition challenge. Int J Comput Vis 115(3):211–252
33. Sajjadi M, Javanmardi M, Tasdizen T (2016) Mutual exclusivity loss for semi-supervised deep learning. In: 2016 IEEE

international conference on image processing (ICIP), IEEE, pp 1908–1912
34. Sajjadi M, Javanmardi M, tasdizen T (2016) Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In: Advances in neural information processing systems, pp 1163–1171
35. Salimans T, Goodfellow I, Zaremba W, Cheung V, Radford A, Chen X (2016) Improved techniques for training gans. In: Advances in neural information processing systems, pp 2234–2242
36. Springenberg JT (2015) Unsupervised and semi-supervised learning with categorical generative adversarial networks. arXiv:1511.06390
37. Wan L, Zeiler M, Zhang S, Le Cun Y, Fergus R (2013) Regularization of neural networks using dropconnect. In: International conference on machine learning, pp 1058–1066
38. Weston J, Ratle F, Mobahi H, Collobert R (2012) Deep learning via semi-supervised embedding. In: Neural networks: tricks of the trade, Springer, pp 639–655
39. Yan Y, Xu Z, Tsang IW, Long G, Yang Y (2016) Robust semi-supervised learning through label aggregation. In: AAAI, pp 2244–2250
40. Zhao JJ, Mathieu M, Goroshin R, LeCun Y (2015) Stacked what-where auto-encoders. CoRR arXiv:abs/1506.02351
41. Zhu X (2006) Semi-supervised learning literature survey. Computer Science University of Wisconsin-Madison 2(3):4
42. Zhu X, Goldberg AB (2009) Introduction to semi-supervised learning. Synthesis lectures on artificial intelligence and machine learning 3(1):1–130

Zhun Li received the Bachelor of Science degree in Mathematics from Peking University, China in 2006 and the Master of Economics degree in Statistics from Renmin University, China in 2008. He is currently working toward the Master of Science degree in Computer Science from Korea Advanced Institute of Science and Technology (KAIST) in Korea. His research interests include deep learning and data mining.

ByungSoo Ko is currently a deep learning and computer vision research engineer in NAVER, Korea. He received the Bachelor of Science degree in Computer Science from Chungnam National University (CNU), Korea in 2016 and the Master of Science degree in Computer Science from Korea Advanced Institute of Science and Technology (KAIST), Korea in 2018. His research interests include deep learning for computer vision with a special focus on image retrieval and neural network architecture.

Prof. Ho-Jin Choi is with the School of Computing at KAIST, Korea since 2009. He received BS (1982) in Computer Engineering from Seoul National University, Korea, MSc (1985) in Computing Software and Systems Design from Newcastle University, UK, and PhD (1995) in Artificial Intelligence from Imperial College London, UK. From 1982 to 1989 he worked for DACOM Inc., Korea; from 1997 to 2002 for Korea Aerospace University, Korea; and from 2002 to 2009 for Information and Communications University (ICU), Korea. In 2002 and 2003 he visited Carnegie Mellon University (CMU), USA, and since then served as an adjunct professor of CMU for the Master of Software Engineering (MSE) program. Since 2010 he has been participating in the Systems Biomedical Informatics Research Center at Seoul National University. He is a member of IEEE Computer Society, ACM, and AAAI. He also serves in the boards of directors for the Software Engineering Society of Korea, and for the Artificial Intelligence Society of Korea. His current research areas include artificial intelligence, natural language processing and biomedical informatics.