Deep Growing Learning

Guangcong Wang[1], Xiaohua Xie[1,2,4]*, Jianhuang Lai[1,2,4], and Jiaxuan Zhuo[3]
[1]School of Data and Computer Science, Sun Yat-sen University, China
[2]Guangdong Key Laboratory of Information Security Technology
[3]School of Electronics and Information Technology, Sun Yat-sen University, China
[4]Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education
{wanggc3, zhuojx5}@mail2.sysu.edu.cn, {xiexiaoh6, stsljh}@mail.sysu.edu.cn

Abstract

Semi-supervised learning (SSL) is an import paradigm to make full use of a large amount of unlabeled data in machine learning. A bottleneck of SSL is the overfitting problem when training over the limited labeled data, especially on a complex model like a deep neural network. To get around this bottleneck, we propose a bio-inspired SSL framework on deep neural network, namely Deep Growing Learning (DGL). Specifically, we formulate the SSL as an EM-like process, where the deep network alternately iterates between automatically growing convolutional layers and selecting reliable pseudo-labeled data for training. The DGL guarantees that a shallow neural network is trained with labeled data, while a deeper neural network is trained with growing amount of reliable pseudo-labeled data, so as to alleviate the overfitting problem. Experiments on different visual recognition tasks have verified the effectiveness of DGL.

1. Introduction

Deep convolutional neural networks (CNNs) [22, 21, 12] have exhibited state-of-the-art performance in image classification [22, 21, 35, 4], and the prediction accuracy of the CNN-based classifier can be improved by training on massive amounts of data (*e.g.*, ImageNet [2]). However, the task of labeling huge amounts of samples is quite expensive and time-consuming. Recent research [28, 26, 15, 18, 23] reveals that the semi-supervised learning, which uses a large amount of unlabeled data in conjunction with a small amount of labeled data, is also greatly beneficial to train deep networks and has achieved leading results on the MNIST [22] and CIFAR-10 [20] benchmarks.

Driven by the significance of the semi-supervised deep learning, a question arises: how to exploit both labeled and unlabeled data to train deep neural networks in an end-to-end way? An obstacle to answering this question is the overfitting problem when training a deep neural network with a large number of parameters by exploiting a few labeled examples only. Existing works [28, 26, 18, 23, 15, 33] that may offer a partial solution to this problem can be generally categorized into three groups based on different schemes of collaborating labeled data with unlabeled data.

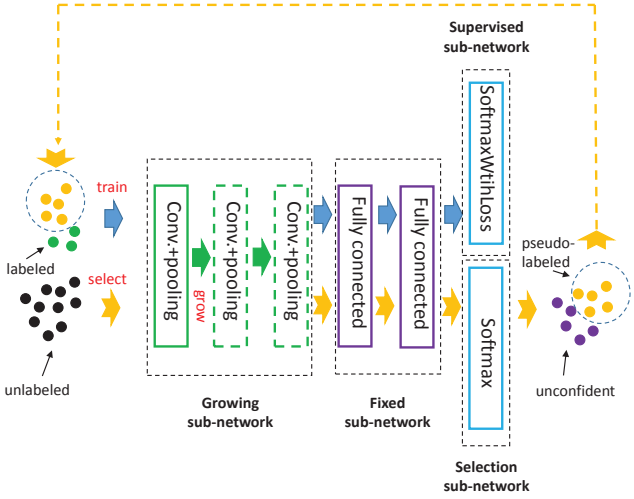


Figure 1: Overview. Deep growing learning can efficiently learn a deep convolutional network with a small amount of labeled data and a large amount of unlabeled data. Basically, it consists of three phases: first, it grows a new layer when meets some conditions. Second, we train the network using the labeled and pseudo-labeled data. Third, we predict the labels of all the unlabeled data and select the confident ones as pseudo-labeled data. This process is repeated until a certain termination condition is reached. The proposed architecture is comprised of four sub-networks: growing sub-network, fixed sub-network, supervised sub-network, and selection sub-network. (Best viewed in color)

*Corresponding author

In the first group, the semi-supervised learning is achieved by combining two phases: unsupervised pre-training and fine-tuning. The weights of all layers are firstly initialized by a layer-wise unsupervised training and then trained globally with a few labeled examples using the standard stochastic gradient-based optimization algorithm in a supervised fashion [13, 6, 18]. However, one would argue that training on labeled and unlabeled data separately should be inferior to a joint approach that trains on labeled and unlabeled data simultaneously because all the learning from the first stage may be "forgotten" in the second stage, and thus can not make full use of the unlabeled data.

In the second group, both supervised learning and unsupervised learning are integrated into a unified framework to jointly train on both labeled and unlabeled data. For example, [15, 33, 32, 25] introduced a semi-supervised regularizer into deep architectures, where labeled data (unlabeled data) is used as a regularizer and incorporated into a unsupervised (supervised) objective function to learn a metric embedding or probability distributions. Unfortunately, a new degradation problem has been exposed: these embeddings or probability distributions have not yet been learned at the beginning of the training, leading to an error catastrophe of the clustering. Different from using a semi-supervised regularizer, the literatures [36, 28, 26] are presented to integrate unsupervised and supervised learning into a unified network by employing the former one for low-lever image reconstruction while the latter one for high-level image classification.

In the third group, labeled and unlabeled data are exploited alternately. This kind of methods often resorts to a heuristic self-training scheme for training deep neural networks, where a classifier is firstly trained using only the labeled data and then applied to the unlabeled data to generate more pseudo-labeled examples for supervised learning; this process is repeated until a certain termination condition is reached [23]. The drawback of this method is the intractable problem of overfitting where the method uses a few labeled examples to train a deep neural network at the beginning.

Along with the line of third-group method, we propose a novel *growing learning framework*. Instead of training a fixed deep neural network all the time, we introduce a growing network initialized with a very shallow neural network to avoid overfitting. Basically, our training scheme consists of three phases: growing network, training network, and selecting pseudo-labeled examples. First, the network automatically grows a new layer if meets some conditions. Second, labeled and pseudo-labeled examples (if any) are used to train the network. Third, the trained model is used to predict the labels of all the unlabeled data and to select the confident ones as pseudo-labeled data. This process is repeated until reaching the termination condition. Fig. 1 illustrates the framework of our method.

Growing learning can be implicitly explained as an EM-like algorithm, which alternates between automatically growing convolutional layers and selecting reliable pseudo-labeled data. Specifically, the better (deeper) network we train, the more pseudo-labeled examples we get, and the growth of the pseudo-labeled examples in turn helps to train the deeper network. Growing learning can be also thought of as analogous to a human learning process which does not happen all at once, but builds upon and is shaped by previous knowledge. With the accumulation of experience (data), human beings can boost themselves up by gradually changing the structure and function of neurons [19, 11] (called neuroplasticity) and thus can learn a highly complex task.

Deep growing learning can be realized by automatically adding new layers on the network if meets some conditions. The entire network can still be trained end-to-end by SGD with backpropagation, and can be easily implemented using common libraries (*e.g.*, Caffe [17]).

In summary, this paper makes three main contributions.

- First, like the human learning process, we propose a novel deep growing learning method, where with the accumulation of data (experience) increasing, the ability of the classifier is improved by dynamically growing new layers.
- Second, the deep growing learning can be regarded as an EM-like process, where the deep network alternately iterates between growing convolutional layers and selecting reliable pseudo-labeled data, and thus provides a new paradigm well for semi-supervised deep learning by automatically fitting growing amount of reliable pseudo-labeled data.
- Third, extensive experiments show that the proposed method is effective in visual recognition tasks on both MNIST and CIFAR-10 datasets.

The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 introduces our deep growing learning model. The experimental comparisons, ablation studies, and analyses are presented in Section 4. Section 5 concludes the paper.

2. Related Work

Semi-supervised deep neural networks. Many researches have shown the use of semi-supervised deep models to learn accurate classifiers. Most of existing approaches [13, 6] provide an effective method for semi-supervised learning by using unsupervised pre-training and fine-tuning separately. Recently, Kingma [18] *et al*. extended the semi-supervised learning problem as a specialised missing data imputation task which firstly learns a new latent representation using

a generative model and subsequently trains with a latent label variable. However, the disjoint training scheme of collaborating labeled data with unlabeled data may lead to sub-optimal solutions.

Instead of training on labeled and unlabeled data separately, some semi-supervised deep models [10, 15, 33] aim to train an accurate classifier using labeled and unlabeled data simultaneously. For example, Grandvalet and Bengio [10] introduced minimum entropy to regularize generative models, and the weighting of unlabeled data provides robustness to the violation of the cluster assumption. In [15, 33], the labeled examples are used as targets to learn a metric embedding using labeled data and then each unlabeled data is assigned to a distinct cluster based on embedded distance. The above methods share a common criterion of examples being more related to nearby examples than to examples farther away. Furthermore, some approaches [32, 25, 3] extended Generative Adversarial Networks (GANs) [9] to semi-supervised learning. Springenberg [32] used an objective function that trades off mutual information between observed examples and their predicted categorical class distribution, against robustness of the classifier to an adversarial generative model. Odena [25] extended GANs to the semi-supervised context by forcing the discriminator network to output class labels while Denton *et al*. [3] introduced a simple semi-supervised learning approach to generate whole images based on in-painting using an adversarial loss. Different from using a semi-supervised regularizer, the Ladder network [28, 26] is extended to simultaneously minimize the sum of supervised and unsupervised cost functions by backpropagation in a unified network. Besides, the method of [36] integrates discriminative and generative pathways and provides a unified approach to supervised, semi-supervised and unsupervised learning without relying on sampling during training.

Among the works that share similar idea to ours, a self-training scheme [23] is exploited to train a semi-supervised deep model. The main idea is to pick up the class which has the maximum predicted probability as the pseudo-label of unlabeled data every weights update to boost up the performance in a supervised fashion, alternately exploiting labeled and unlabeled data. Our approach differs from this method, as the proposed model can fit the increasing data by dynamically adding parameters.

Layer-wise training. To address the difficulty of training deep multi-layer neural networks, an approach [7] based on constructively adding layers in a supervised fashion has been explored with some success. Hinton *et al*. [13] proposed a greedy algorithm that can learn deep and directed belief networks one layer at a time to initialize a learning procedure. This algorithm was further studied by [1]. In [1], they confirmed the hypothesis that the greedy layer-wise unsupervised training strategy mostly helps the optimization, by initializing weights in a region near a good local minimum, giving rise to internal distributed representations that are high-level abstractions of the input, bringing better generalization.

Inspired by the greedy layer-wise unsupervised training, one would think if the layer-wise training scheme is beneficial to the training of convolutional neural networks. This topic was investigated partly in [30] and the result shows that the greedy layer-wise training strategy can be advantageous for training deep convolutional neural networks with small size datasets, yielding cleaner and more interpretable visual features, as well as improved accuracy. To train a very deep convolutional networks for large-scale image recognition, the approach [14] begins with training a shallow network, and then trains a deeper network initialized by the shallow one. These methods suggest that layer-wise training can simplify the optimization. Different from these methods, our DGL model aims to fit the increasing data by automatically growing layers.

3. Deep Growing Learning

Deep growing learning aims to learn an accurate classifier with a small amount of labeled data and a large amount of unlabeled data. With limited labeled data, it is difficult for deep neural networks to deal with overfitting due to a large number of parameters to be trained. To prevent deep networks from overfitting, we firstly train a shallow network with labeled data and subsequently feed the unlabeled data to pick up the confident ones as pseudo-labeled data, which is further used to train a deeper network.

3.1. Self-training

Self-training [34, 37], also known as self-teaching or bootstrapping, is one of techniques using both labeled and unlabeled data to improve learning. Given a set of labeled data L and unlabeled data U, self-training proceeds as follows: train a classifier C using L, and classify U with C; select a pseudo-labeled subset $U^{'}$ $(U^{'} \subset U)$ for which C has the highest confidence scores; add U' to L and remove U' from U. Repeat the process until the algorithm converges. Note that, C can be any classifier, *e.g*., SVM, random forest, boosting tree, and neural networks.

Self-training is based on an assumption that examples from the same class follow a coherent distribution. It can produce outstanding results because unlabeled examples provide a wealth of information to describe the details of the data structure and give a better sense of the class separation boundary. The drawback of the self-training is that misclassified instances could occur and reinforce itself.

One of the interesting things about self-training is that it can generate more and more pseudo-labeled data with the increase of the iterations. On the one hand, if one designs a classifier with an appropriate number of parameters to fit

the labeled data at the begin of self-training, the classifier may not have the capability to fit the increasing pseudo-labeled data. On the other hand, if one designs a classifier initialized with a large number of parameters, the overfitting problem could occur due to the limited labeled data. To remedy these problems, we propose a deep growing learning framework to fit the increasing pseudo-labeled data by dynamically adding parameters to the classifier.

3.2. Deep Growing Learning Framework

The proposed deep growing learning framework trains a shallow network and selects the confident prediction examples as the next iteration of the loop to train a deeper network. During the training, the pseudo-labeled data set is re-evaluated each K iterations. Specifically, our architecture is comprised of four sub-networks: growing sub-network, fixed sub-network, supervised sub-network, and selection sub-network (Fig. 1).

Growing sub-network. The growing sub-network consists of a cascade of "building blocks". Here, each building block may contain convolutional filtering, pooling, nonlinear activation, and batch normalization. As a start, the growing sub-network contains only one building block. This sub-network serves as a seed that can automatically grows a new building block when the network can not fit the labeled data and the increasing pseudo-labeled data. We copy the parameters of the previous growing sub-network to the new one and then fine-tune the network globally. The sub-network continues growing up before it goes into overfitting.

Fixed sub-network. The growing sub-network is followed by two fully connected layers, which are used to reduce the dimensionality and compute feature transformation. In practice, we add some other layers after each fully connected layer, *e.g.*, dropout and nonlinear activation.

Supervised sub-network. The supervised sub-network ("SoftmaxWithloss" layer) computes the multinomial logistic loss for a one-of-many classification task, passing real-valued prediction through a softmax to get a probability distribution over classes. This sub-network actually encodes the convolutional features into a one-hot vector using the stochastic gradient descent algorithm.

Selection sub-network. After renewing the network structure and updating the parameters by K-iteration training, all unlabeled examples are fed into this sub-network ("Softmax" layer) to predict their labels. Because of the limited classifying ability of the shallow classifier C, some prediction results are unreliable. To get confident predictions, the selection sub-network aims to compute the probability distribution over N classes and find out the maximal value by

$$l = \arg\max_i p(y = i|x, C), 0 \leq i \leq N-1, \qquad (1)$$

where $p(y = i|x, C)$ is the probability of the class label i given a point x and a classifier C. If the value $p(y = l|x, C)$ is greater than a threshold value α, then we regard (x, l) as a pseudo-labeled example. In this way, we can get a set of pseudo-labeled examples, which is subsequently used to update the pseudo-labeled data set $U^{'}$.

In our framework, growing learning can be implicitly explained as an EM-like algorithm, which alternates between automatically growing network and selecting reliable pseudo-labeled data. With the increasing pseudo-labeled data, we can train a deeper network, and we in turn select more pseudo-labeled data counting on the improved performance of the growing model. Errors of pseudo-labeled examples could occur but could be corrected due to re-evaluation of unlabeled data every K iterations. From the general trend, the network gradually boosts itself up by alternately iterating until the convergence.

3.3. Deep Growing Learing Algorithm Principle

As informally described in Section 3.2, the DGL algorithm makes use of three straightforward and intuitive assumptions: (i) The more pseudo-labeled data we have, the more accurate classifier we train; (ii) The more accurate classifier we have, the more pseudo-labeled data we select; (iii) Using enough data with noise, deeper networks outperform shallower ones. Actually, Assumption (i) and (ii) have been proved by the self-training theory [34, 37]. Many deep models [14, 12] also present an detailed interpretation of Assumption (iii).

We first consider a one-layer classifier $C^L_{net_1}$, which is trained over the limited labeled data set L. According to Eq. 1, we select a set of pseudo-labeled data $U^{'}$ from a set of unlabeled data U by

$$U^{'} = \{x|p(y = l|x, C^L_{net_1}) > \alpha, x \in U\} \qquad (2)$$

where $p(y = l|x, C^L_{net_1})$ represents the probability of the candidate pseudo-label given a point x and a one-layer classifier $C^L_{net_1}$. Using the set $L \bigcup U^{'}$, we then train a more accurate classifier $C^{L \bigcup U^{'}}_{net_1}$ based on Assumption (i). We re-select a new set of pseudo-labeled data $U^{''}$ by

$$U^{''} = \{x|p(y = l|x, C^{L \bigcup U^{'}}_{net_1}) > \alpha, x \in U\} \qquad (3)$$

Because the classifier $C^{L \bigcup U^{'}}_{net_1}$ is better than $C^L_{net_1}$, we have $|U^{''}| > |U^{'}|$ based on Assumption (ii), where $|\cdot|$ represents the size of a set. We thus have

$$U^{'''} = \{x|p(y = l|x, C^{L \bigcup U^{''}}_{net_1}) > \alpha, x \in U\} \qquad (4)$$

In this way, we repeat this process until the performance of this classifier does not improve. At this time, we obtain the set of pseudo-labeled U^o and the one-layer classifi-

er $C_{net_1}^{L \bigcup U^o}$. Using the data set $L \bigcup U^o$, the classifier automatically grows a new layer, denoted by $C_{net_2}^{L \bigcup U^o}$, which is better than $C_{net_1}^{L \bigcup U^o}$ based on Assumption (iii). The training process then enters a loop according to Eq. 2, 3, 4. In this way, the DGL model boosts itself up to automatically fit the increasing data. During training, the test error of the validation set firstly goes down and then goes up like a bowl-shaped curve, corresponding to under-fitting and overfitting, respectively. It is easy to find optimal point according to each recorded evaluation error. When the error starts to go up, we stop the growth of DGL. The details of our model are shown in Algorithm 1.

Algorithm 1: Deep growing learning algorithm

Input: labeled data set L and unlabeled data set U
Output: The network parameters Net_o

```
Initialize a shallow net Net_1;
Initialize a pseudo-labeled data set U' with L;
i ← 1;
while Net_i does not go into overfitting do
    while Net_i does not go into underfitting do
        for j = 1 : K do
            m/2 random samples from L;
            m/2 random samples from U';
            Feed m samples into the network;
            Update parameters;
        end
        Select pseudo-labeled examples U''
        according to Eq. 1;
        Update U' ← U'';
    end
    Grow a new layer;
    Copy parameters Net_i to Net_{i+1};
    Fine-tune Net_{i+1} globally;
    i ← i + 1;
end
Net_o ← Net_{i-1};//Net_i has went into overfitting
```

4. Experiments

In this section, we apply our growing learning method to two benchmark datasets for evaluation, *i.e.*, MNIST and CIFAR-10, with regard to digital recognition and image classification, respectively. In both datasets, we compare the state-of-the-art methods with our model, showing the effectiveness of growing learning. We also present ablation studies to reveal the benefits of each main component of our method, *i.e.*, the growing sub-network and the selection sub-network. Besides, we study the influence of the number of labeled examples and the effect of threshold α value. Finally, we further investigate how growing learning addresses the overfitting problem.

Experimental setting. In our framework, we use mini-batch learning in our experiments to reduce the memory requirements. In both datasets, we randomly select a batch of samples from the original training set, where half of samples are from labeled data and the other are from pseudo-labeled data. The initial parameters of the convolutional and full layers are set by two zero-mean Gaussian distributions, whose standard deviations are 0.01 and 0.1, respectively. We use a momentum of 0.95 and adopt batch normalization (BN) right after each convolution and before activation. We train a shallow convolutional network from scratch and do not need any pre-training on extra datasets. We use SGD with a mini-batch size of 100. The learning rate starts from 0.001 and is divided by 10 each 200,000 iterations. The other specific settings for different datasets are included in the following sub-sections.

In practice, we empirically predefine an ultimate deep network architecture and building blocks for a specific recognition task. After initialization, the shallow network grows to the ultimate deep network by adding building blocks one after one. In this paper, we predefine two deep networks for MNIST and CIFAR-10, respectively. The details of the networks are shown in Table 1. The building block of the neural network designed on CIFAR-10 consists of two convolutional layers because CIFAR-10 is more challenging than MNIST.

4.1. Evaluations on the MNIST Dataset

The MNIST database of handwritten digits [22] consists of handwritten digit images, 28×28 in size, organized into 10 classes (0 to 9) with 60,000 training and 10,000 test samples. The 60,000 labeled examples are randomly split into a 10,000-sample validation set and 50,000 samples as the training set without overlap.

Following precious works [28], we randomly choose 100 labeled data from the training set as labeled set. The number of examples for different classes is balanced (10 for each class). The validation set is used for evaluating the model structure and hyperparameters. All the samples except labeled set are used to predict and select pseudo-labeled data, which does not utilize the actual labels. We repeat each training 10 times, varying the random seed that is used for the splits. For the growing neural network, we use a 2-layer convolutional network, where each layer is followed by a ReLU non-linear layer and a Max Pooling layer. The network grows up layer after layer during training. The details of the network are described in Table 1. In our experiment, all the images are cropped to the size of 24×24 at the center with a small random perturbation. We set the threshold value $\alpha = 0.99$.

We compare our approach with a broad range of exist-

MNIST	CIFAR-10
Input: 24×24 monochrome	24×24 RGB
Conv-ReLU $(16, 5 \times 5, 1 \times 1, 1 \times 1)$ Max-Pooling $(2 \times 2, 2 \times 2, 0 \times 0)$	Conv-ReLU-BN $(32, 3 \times 3, 1 \times 1, 1 \times 1)$ Conv-ReLU-BN $(32, 3 \times 3, 1 \times 1, 1 \times 1)$ Max-Pooling $(2 \times 2, 2 \times 2, 0 \times 0)$
	Conv-ReLU-BN $(64, 3 \times 3, 1 \times 1, 1 \times 1)$ Conv-ReLU-BN $(64, 3 \times 3, 1 \times 1, 1 \times 1)$ Max-Pooling $(2 \times 2, 2 \times 2, 0 \times 0)$
Conv-ReLU $(16, 5 \times 5, 1 \times 1, 1 \times 1)$ Max-Pooling $(2 \times 2, 2 \times 2, 0 \times 0)$	Conv-ReLU-BN $(128, 3 \times 3, 1 \times 1, 1 \times 1)$ Conv-ReLU-BN $(128, 3 \times 3, 1 \times 1, 1 \times 1)$ Max-Pooling $(2 \times 2, 2 \times 2, 0 \times 0)$
	Conv-ReLU-BN $(128, 2 \times 2, 1 \times 1, 1 \times 1)$ Conv-ReLU-BN $(128, 2 \times 2, 1 \times 1, 1 \times 1)$ Max-Pooling $(2 \times 2, 1 \times 1, 0 \times 0)$
FC-ReLU-Dropout (1024)	FC-ReLU-Dropout (64)
FC (10)	FC (10)

Table 1: The employed network architectures for MNIST and CIFAR-10, respectively. The convolutional layers, pooling layers and fully connected layers are denoted by $(featuremaps, kernel, stride, pad)$, $(kernel, stride, pad)$, and $(channel)$, respectively.

ing methods, as shown in Table 2. In particular, the competing methods can be grouped into two categories. The first group includes the methods using handcrafted features, such as Support Vector Machines on the labeled set (SVM) [29], transductive SVM (TSVM) [29], and AtlasRBF [27]. The experimental results on MNIST clearly demonstrate the effectiveness of our model against the other classifiers without counting on deep feature learning, even though our deep model only exploits limited labeled data.

The second group of comparing methods includes thirteen representative semi-supervised deep learning models: Neural Networks (NN) [29], Neural Networks with Embedding regularizer (EmbedNN) [33], Convolutional Neural Network (CNN) [29], Convolutional Neural Network for Embedding (EmbedCNN) [33], Contractive Auto-Encoders (CAE) [29], Manifold Tangent Classifier (MTC) [29], Pseudo-Label (+PL) [23], Pseudo-Label using unsupervised pretraining with DAE (+PL+DAE) [23], SWWAE [36], Deep Generative Networks (DGN) [18], Virtual Adversarial [24], Ladder network [28], and CatGAN [32]. It is encouraging to see that our approach significantly outperforms the competing methods. Among the comparing deep learning models, NN and CNN easily go into overfitting and yield steadily worse test error using the labeled set for training only. EmbedNN, EmbedCNN, and CatGAN improve the accuracy due to the semi-supervised regularization, but not as much. This is because the semi-supervised embeddings have not yet been learned at the beginning of the training, and are thus inefficient for the unlabeled examples to be clustered, leading to an error accumulation problem. DGN fails to provide a unified mechanism to unsupervised and supervised learning, which may degrade the performance. Our DGL model also outperforms SWWAE and Ladder networks, which integrate discriminative and generative pathways to provide a unified approach for unsupervised and supervised learning. The main reason may be the simple integration of two different learning tasks (*i.e.*, the former one for image reconstructions while the latter one for high-level classification). Besides, it is difficult for +PL and +PL+DAE to address the overfitting problem because of the fixed-depth networks.

4.2. Evaluations on the CIFAR-10 Dataset

CIFAR-10 consists of 60,000 color images in 10 classes, with 6,000 images per class. There are 50,000 training images and 10,000 test images. The 50,000 labeled examples are randomly split into a 10,000-sample validation set and 40,000 samples as the training set without overlap. The classes include airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks.

For CIFAR-10, we randomly choose 4,000 labeled data from the training set as the labeled set (400 for each class). The validation set is used for evaluating the model structure and hyperparameters. All the samples except the selected labeled set are used to be predict and select pseudo-labeled data. For the growing neural network, we use a convolutional network with 4 building blocks, each of which consists of two convolutional layers. The network grows block after block during training. The details of the network are described in Table 1. All the images are also cropped to the size of 24×24 at the center with a small random perturbation. We set the threshold value $\alpha = 0.98$. The other

Model	Test error %
SVM [29]	23.44
TSVM [29]	16.81
AtlasRBF [27]	8.10($\pm$0.95)
NN [29]	25.81
CNN [29]	22.98
EmbedNN [33]	16.86
EmbedCNN [33]	7.75
CAE [29]	13.47
MTC [29]	12.03
+PL [23]	16.15
+PL+DAE [23]	10.49
SWWAE [36]	9.17
DGN [18]	3.33($\pm$0.14)
Virtual Adversarial [24]	2.12
Ladder network [28]	0.89($\pm$0.50)
CatGAN [32]	1.39($\pm$0.28)
DGL(Ours)	**0.76**($\pm$0.37)

Table 2: Comparison results on MNIST using 100 labeled examples.

Model	Test error %
Spike-and-Slab Sparse Coding [8]	31.9
View-Invariant k-means [16]	27.4($\pm$0.7)
Exemplar-CNN [5]	23.4($\pm$0.2)
Ladder network [28]	20.04($\pm$0.47)
CatGan [32]	19.58($\pm$0.58)
ImprovedGan [31]	18.63($\pm$2.32)
DGL(Ours)	**17.56**($\pm$0.31)

Table 3: Comparison results on CIFAR-10 using 4,000 labeled examples.

settings are the same as that of MNIST.

We also conduct two kinds of comparison experiments, as shown in Table 3. First, we compare our DGL method with the handcraft feature methods, *e.g.*, Spike-and-Slab Sparse Coding [8] and View-Invariant k-means [16]. Then, we compare our DGL method with some other semi-supervised deep methods, *e.g.*, Exemplar-CNN [5], Ladder network [28], CatGan [32], and ImprovedGan [31]. We can see that the proposed method outperforms all the previously reported results.

4.3. Ablation Studies and Further Analysis

In this section, we conduct several experiments to validate the effects of different modules in our framework, which are evaluated on CIFAR-10.

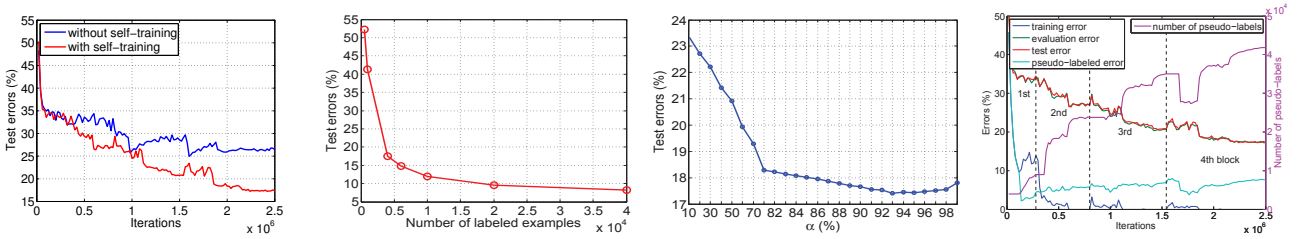
Effect of growing learning. To show the benefit of our deep growing learning, we set a baseline model by removing the selection sub-network and replacing the growing sub-network with a fixed four-block network sub-network. The dataset varies the size of the labeled data from 4,000 to 40,000. We ensure that all classes are balanced when doing this, *i.e.*, each class has the same number of labeled examples. Note that, only the labeled examples are exploited in the baseline experiment. Our DGL model uses the same network as the predefined network and grows building blocks one after one. As can be observed in Table 4, our DGL model improves the performance from 35.38% to 17.56%, 24.05% to 12.77%, 16.70% to 10.41%, and 12.39% to 8.07%. And the superiority of our approach against them should be attributed to the deployment of both unlabeled data and the effective growing learning method for semi-supervised learning.

Model	4000	10000	20000	40000
baseline	35.38	24.05	16.70	12.39
DGL(Ours)	**17.56**	**12.77**	**10.41**	**8.07**

Table 4: Comparison between DGL and a baseline on CIFAR-10 using 4,000, 10,000, 20,000, and 40,000 labeled examples, respectively.

Effect of growing sub-network. To show the benefit of the growing sub-network, we conduct an ablation study by isolating this sub-network, *i.e.*, replacing the growing sub-network by four fixed-depth networks, respectively. In this setting, we still use self-training scheme to train these models for fair comparisons. That is, we select the pseudo-labeled data each K iterations. Specifically, we design four fixed-depth convolutional neural networks, i.e., one-block network, two-block network, three-block network, and four-block network, as shown in Table 1. The results are reported in Table 5. It is obvious that without the growing learning the performance drops by 12.71%, 9.42%, 12.60%, and 15.87%, respectively. We can see that the shallower networks (*e.g.*, one-block network) can fit the labeled data, but fail to fit the increasing pseudo-labeled data. Conversely, deeper networks (*e.g.*, four-block network) can fit the increasing pseudo-labeled data, but easily go into overfitting when training on limited labeled data. The two-block network may gain a slight improvement but is still in a dilemma. The above results clearly demonstrate the effectiveness of utilizing growing learning for semi-supervised learning.

Effect of selection sub-network. To show the benefit of the selection sub-network, we conduct an ablation study by isolating this sub-network. To achieve this, we remove the selection sub-network, and thus we only use labeled data to our growing learning model. Note that, we still keep the growing sub-network. In this case, our growing learning model is degraded as [30]. It is observed that the performance drops by 8.96% when removing the self-training



(a) With/without self-training. (b) Effect of the number of labels. (c) Performance w.r.t. α. (d) Several kinds of errors.

Figure 2: Results of the ablation studies demonstrating the effectiveness of our models.

Model	Test error %
one-block network	30.27
two-block network	26.98
three-block network	30.16
four-block network	33.43
DGL(Ours)	**17.56**

Table 5: Effect of the growing sub-network on CIFAR-10 using 4,000 labeled examples.

scheme, as shown in Fig. 2 (a). From another point of view, the unlabeled examples provide more information to describe the details of the data structure. With the network growing up, the performance of our classifier improves and thus the labeled examples propagate their labels to the unlabeled data region gradually. The growth of the pseudo-labeled data in turn helps the network grow up. At the end, an effective decision boundary forms.

Effect of the number of labeled examples. To study the effect of the number of labeled examples, we conduct a controlled experiment. The only difference from the experimental setting above is the size of the labeled examples, which varies from 5,00 to 40,000. The results are shown in Fig. 2 (b). We can see that the performance improves significantly from using 500 to 4,000 labeled samples, but gains a slight increase from 4,000 to 40,000. This is because the classifier training over few labeled examples is nearly degraded as unsupervised learning, which hardly learns robust feature representations and a discriminant similarity measure. When we have enough labeled examples (*e.g.*, 4,000), we can train a competitive classifier by utilizing a large amount of unlabeled examples, compared with supervised learning.

Effect of threshold α value. To investigate the impact of α value on the performance, we conduct a sensitivity analysis experiment on CIFAR-10 dataset. Fig. 2 (c) shows the robustness of our model with respect to α value. When α is in the range of 86-99%, the proposed DGL model achieves less than 18% test error.

How growing learning addresses the overfitting problem. To further study how growing learning addresses the overfitting problem, we conduct an experiment to investigate the relationships between training loss, evaluation error, test error, the number of the pseudo-labeled examples, pseudo-labeled error, and the number of the building blocks. Note that, training error is only the error of the labeled data, without considering unlabeled one. The results are shown in Fig. 2 (d). It is demonstrated that 1) although the training error is reduced to zero, the testing/evaluation error is still very large. This means that the model easily goes into overfitting if we only train the model on the labeled data; 2) To alleviate the overfitting problem, pseudo-labeled data is exploited. In the case of keeping the error of the pseudo-labeled data (serves as a pseudo training error), the test/evaluation error drops significantly. 3) As the network grows up, the number of pseudo-labeled data increases, which actually in turn helps train a more accurate (deeper) classifier. The results, therefore, validate the effectiveness of growing learning for addressing overfitting.

5. Conclusions

In this work, we presented a deep growing learning framework where the deep network alternately iterates between automatically growing convolutional layers and selecting reliable pseudo-labeled data for training, so as to address the overfitting problem for semi-supervised learning on deep neural network model. We integrated growing learning into the self-training scheme and achieved competitive results with state-of-the-art on both MNIST and CIFAR-10 in the semi-supervised regime. Further exploration is needed to extend the proposed DGL to online learning, where data becomes available in a sequential order, gradually increasing from a small set to a large one, *e.g.*, video data.

Acknowledgments

This project was supported by the National Natural Science Foundation of China (U1611461, 61573387, 61672544).

References

[1] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In *NIPS*, 2007.

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009.

[3] E. Denton, S. Gross, and R. Fergus. Semi-supervised learning with context-conditional generative adversarial networks. *arXiv:161.06430*, 2016.

[4] S. Ding, L. Lin, G. Wang, and H. Chao. Deep feature learning with relative distance comparison for person re-identification. *Pattern Recognition*, 48(10):2993–3003, 2015.

[5] A. Dosovitskiy, P. Fischer, J. T. Springenberg, M. Riedmiller, and T. Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *PAMI*, 38(9):1734–1747, 2016.

[6] D. Erhan, Y. Bengio, A. Courville, P. A. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *JMLR*, 11(3):625–660, 2010.

[7] S. Fahlman and C. Lebiere. The cascade-correlation learning architecture. In *NIPS*, pages 524–532, 1990.

[8] I. J. Goodfellow, A. Courville, and Y. Bengio. Large-scale feature learning with spike-and-slab sparse coding. *arXiv:1606.01583*, 2012.

[9] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. In *NIPS*, pages 2672–2680, 2014.

[10] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *NIPS*, volume 17, pages 529–536, 2004.

[11] A. W. Grossman, J. D. Churchill, K. E. Bates, J. A. Kleim, and W. T. Greenough. A brain adaptation view of plasticity: is synaptic plasticity an overly limited concept? *Progress in brain research*, 138:91–108, 2002.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[13] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2013.

[14] G. E. Hinton, S. Osindero, and Y.-W. Teh. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[15] E. Hoffer and N. Ailon. Semi-supervised deep learning by metric embedding. *arXiv:1611.01449*, 2016.

[16] K. Y. Hui. Direct modeling of complex invariances for visual object features. In *ICML*, pages 485–488, 2013.

[17] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv:1408.5093*, 2014.

[18] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling. Semi-supervised learning with deep generative models. In *NIPS*, pages 3581–3589, 2014.

[19] J. A. Kleim and T. A. Jones. Principles of experience-dependent neural plasticity: implications for rehabilitation after brain damage. *Journal of speech, language, and hearing research*, 51(1):S225–S239, 2008.

[20] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images, 2009. Tech Report.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012.

[22] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[23] D.-H. Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks." in workshop on challenges in representation learning. In *ICML Representation Learning Workshop*, volume 3, page 2, 2013.

[24] T. Miyato, S. Maeda, M. Koyama, K. Nakae, and S. Ishii. Distributional smoothing by virtual adversarial examples. *stat*, 1050:2, 2015.

[25] A. Odena. Semi-supervised learning with generative adversarial networks. *arXiv:1606.01583*, 2016.

[26] M. Pezeshki, L. Fan, P. Brakel, A. Courville, and Y. Bengio. Deconstructing the ladder network architecture. *arXiv:1511.06430*, 2015.

[27] N. Pitelis, C. Russell, and L. Agapito. Semi-supervised learning using an unsupervised atlas. In *ECML*, volume LNCS 8725, pages 565–580, 2014.

[28] A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko. Semi-supervised learning with ladder networks. In *NIPS*, pages 3546–3554, 2015.

[29] S. Rifai, Y. Dauphin, P. Vincent, Y. Bengio, and X. Muller. The manifold tangent classifier. In *NIPS*, pages 2294–2302, 2011.

[30] D. Rueda-Plata, R. R., and F. G. Supervised greedy layer-wise training for deep convolutional networks with small datasets. In *Computational Collective Intelligence*, volume 9329, 2015.

[31] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *NIPS*, pages 2226–2234, 2016.

[32] J. T. Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. In *ICLR*, 2016.

[33] J. Weston, F. Ratle, and H. Mobahi. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*, pages 639–655. Springer, 2012.

[34] D. Yarowsky. Unsupervised word sense disambiguation rivalling supervised methods. In *Annual Meeting of the Association for Computational Linguistics*, pages 189–196, 1995.

[35] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833, 2014.

[36] J. Zhao, M. Mathieu, R. Goroshin, and Y. Lecun. Stacked what-where auto-encoders. *arXiv:1506.02351*, 2015.

[37] Y. Zhou, M. Kantarcioglu, and B. Thuraisingham. Self-training with selection-by-rejection. In *ICDM*, pages 795–803, 2012.