Neurocomputing 202 (2016) 84–90

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Encouraging orthogonality between weight vectors in pretrained deep neural networks

Karol Grzegorczyk, Marcin Kurdziel *, Piotr Iwo Wójcik

AGH University of Science and Technology, Faculty of Computer Science, Electronics and Telecommunications, Department of Computer Science, al. A. Mickiewicza 30, 30–059 Krakow, Poland

ARTICLE INFO

Article history:
Received 19 June 2015
Received in revised form
19 January 2016
Accepted 18 March 2016
Communicated by Dr. Deng Cheng
Available online 6 May 2016

Keywords:
Unsupervised pretraining
Contrastive Divergence
Orthogonalization

ABSTRACT

Deep neural networks have recently shown impressive performance in several machine learning tasks. An important approach to training deep networks, useful especially when labeled data is scarce, relies on unsupervised pretraining of hidden layers followed by supervised finetuning. One of the most widely used approaches to unsupervised pretraining is to train each layer with the Contrastive Divergence (CD) algorithm. In this work we present a modification to CD with the goal of learning more diverse sets of features in hidden layers. In particular, we extend the CD learning rule to penalize cosines of the angles between weight vectors, which in turn encourages orthogonality between the learned features. We demonstrate experimentally that this extension to CD improves performance of pretrained deep networks on image recognition and document retrieval tasks.

1. Introduction

For years neural networks have been attracting attention of researchers in both academia and industry. The main appeal of these models was the prospect of learning multiple layers of data features. Yet, for a long time training deep neural networks was unsuccessful and simpler machine learning algorithms, like support vector machines [1], were more useful in many practical applications. However, research advances from the last decade led to a resurgence of interest in neural networks. A key concept that contributed to this renaissance is training the networks in two phases, namely unsupervised pretraining and supervised finetuning. The first significant work in this direction was presented in [2], where Deep Belief Networks (DBNs), i.e., stacked Restricted Boltzmann Machines [3] (RBMs), were trained layer-by-layer in an unsupervised manner and then finetuned with error backpropagation [4]. The resultant deep models significantly outperformed state-of-the-art approaches on multiple machine learning tasks [5,2].

In recent years several methods have been proposed that improve the performance of networks with no pretraining. Most notably, these include dropout [6] and Hessian-free optimization [7]. Results reported in these two works show, however, that dropout and Hessian-free method also benefit from unsupervised pretraining, which further reduces the generalization error. This agrees with observation in [8] that unsupervised pretraining usually improves network performance. Another justification for unsupervised pretraining stems from the unbalance between available unlabeled and labeled data: data acquisition is relatively inexpensive compared to labeling. Therefore, unsupervised pretraining can typically incorporate much bigger training sets than supervised finetuning.

The aim of the work presented in this article is to improve pretraining of deep neural networks. Specifically, we propose an improved method for pretraining such networks with RBMs. Restricted Boltzmann Machine models the joint probability distribution of training observations and their latent features. These latent features are detected by weight vectors of neurons in the hidden layer. Ideally, each neuron in the hidden layer should represent a distinct latent feature. Otherwise, the effective number of the latent features decreases and, in effect, the hidden layer becomes less expressive. Our goal is therefore to explicitly encourage more diversity in latent features. To this end, we modify the Contrastive Divergence (CD) algorithm [9], i.e., the most widely used training method for RBMs, in a way that penalizes parallel components of the weight vectors. This extension to CD encourages the model to learn more orthogonal weight vectors. We show experimentally that the modified CD improves the performance of pretrained deep networks on image recognition and document retrieval task.

2. Related work

While significant effort has been recently put into improving the training of deep neural networks, relatively little attention has

* Corresponding author.
E-mail addresses: kgr@agh.edu.pl (K. Grzegorczyk), kurdziel@agh.edu.pl (M. Kurdziel), pwojcik@agh.edu.pl (P.I. Wójcik).

http://dx.doi.org/10.1016/j.neucom.2016.03.044
0925-2312/

been given to methods that encourage orthogonality of features learned by the hidden layers. To the best of our knowledge, explicit orthogonalization of weight vectors has thus far been applied in the context of shallow non-pretrained networks and for pretraining local receptive fields in convolutional neural networks.

Specifically, in [10] authors introduced an additional penalty term to the squared error cost function in backpropagation networks, forcing the weight vectors to fulfill the orthonormality constraint. They tested the proposed method on regression and prediction tasks and demonstrated an improvement in the generalization error. However, networks used therein are small by the current standards—authors employed the resilient backpropagation algorithm to train two-layer networks with between 2 and 10 units in a hidden layer. Another work where orthogonalization was used in small networks was presented in [11]. Note that both these works impose an orthogonality (or orthonormality) constraint not in CD but during error backpropagation.

More recently, in [12] authors applied local weights orthogonalization during unsupervised pretraining of tiled convolutional neural networks. Specifically, orthogonalization was employed with topographic independent component analysis and significantly improved the accuracy on two popular image classification benchmarks, namely NORB and CIFAR-10 datasets.

Another related work was presented in [13]. This article proposes a replacement for the CD gradient, with the goal of improving the stability of RBM training. While the proposed gradient does not include explicit orthogonalization of weight vectors, results show that it leads to more orthogonal hidden features than classical CD. However, in experimental evaluation RBM was used only as a one layer feature extractor and no results were reported for deep networks pretrained with the proposed gradient.

Unlike the above studies, in this paper we focus on pretrained non-convolutional deep neural networks. In particular, we investigate orthogonalization of weight vectors during unsupervised pretraining of multilayer perceptron networks and deep autoencoders.

3. Encouraging orthogonality between weight vectors in Restricted Boltzmann Machines

Restricted Boltzmann Machine is a generative neural network with neurons arranged in two layers that form a bipartite graph. Visible layer $\mathbf{v}$ models observations and hidden layer $\mathbf{h}$ models their latent features. As described in [14], the weights w_{ij} define an energy function $E(\mathbf{v},\mathbf{h})$ over configurations of visible and hidden units. The energy then translates into the joint probability $P(\mathbf{v},\mathbf{h})$ of visible and hidden states:

$$P(\mathbf{v},\mathbf{h}) = \frac{e^{-E(\mathbf{v},\mathbf{h})}}{Z}, \tag{1}$$

where Z is the partition function:

$$Z = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}. \tag{2}$$

The goal of the training is to fit the weights so that the marginal distribution $P(\mathbf{v})$ approximates the distribution of observations. The training gradient for the weight w_{ij} is then given by [9]:

$$\nabla_{ij} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{RBM}, \tag{3}$$

where $\langle v_i h_j \rangle_{data}$ is the expected product of visible and hidden activations when RBM observes a training example and $\langle v_i h_j \rangle_{RBM}$ is their expected product under the joint distribution $P(\mathbf{v},\mathbf{h})$ given by the current model parameters. While computation of the first of these terms is straightforward, the second term is intractable. The gradient is therefore approximated, typically with the Contrastive Divergence [9] algorithm. In CD the rule for updating the weights is

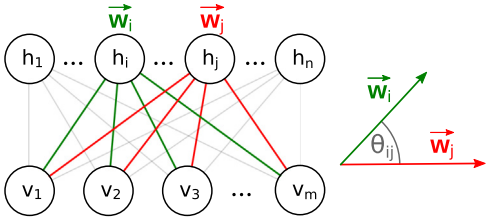

Fig. 1. Encouraging orthogonality between weight vectors in an RBM layer. Green and red connections correspond to weight vectors of the i-th and j-th latent feature, respectively. The goal of the orthogonalization procedure is to increase the orthogonality between all pairs of weight vectors. To this end, we penalize the cosines of the angles between weight vectors: $\cos\theta_{ij}$ (Eqs. (5), (6)). (For color version of this figure the reader is referred to the online version of this paper.)

$$\Delta\mathbf{W} = \varepsilon(\mathbf{v}^T\mathbf{h} - \mathbf{v}_R^T\mathbf{h}_R), \tag{4}$$

where ε is the learning rate, $\mathbf{v}$ is the training example and $\mathbf{h}$ is its corresponding hidden activation vector. Activations $\mathbf{v}_R$ and $\mathbf{h}_R$ are constructed by performing Gibbs sampling of visible and hidden layer, starting the chain from $\mathbf{h}$. When pretraining deep networks usually only one Gibbs step is performed, which is the so called CD-1 algorithm [9].

Columns of the weight matrix $\mathbf{W}$ represent latent features learned by the neurons in the hidden layer. Our goal is to learn a broad set of features in the hidden layer, and therefore we encourage weight vectors to be orthogonal to each other (Fig. 1). Let $\mathbf{w}_k$ denote the k-th column of $\mathbf{W}$, i.e., the k-th latent feature. We encourage orthogonality between latent features by introducing an additional term to the weight update rule, which penalizes parallel components of the weight vectors. Specifically, in an update to a weight vector $\mathbf{w}_k$ we penalize $\frac{1}{n-1}\sum_{j\neq k} o_{kj}\mathbf{w}_j$, where o_{kj} is a coefficient that reflects the degree of non-orthogonality between $\mathbf{w}_k$ and $\mathbf{w}_j$, and n is the number of hidden units. Thus, our weight update rule takes the form:

$$\Delta\mathbf{W} = \varepsilon\left(\mathbf{v}^T\mathbf{h} - \mathbf{v}_R^T\mathbf{h}_R - \frac{\lambda}{n-1}\mathbf{W}\mathbf{O}\right), \tag{5}$$

where we construct matrix $\mathbf{O}$ to reflect the non-orthogonality of weight vectors and introduce λ as the non-orthogonality cost. The non-orthogonality cost should apply only to pairs of different weight vectors, so we set the main diagonal of $\mathbf{O}$ to zeros. For the off-diagonal elements we initially considered three types of non-orthogonality coefficients o_{ij}, namely cosine of the angle between weight vectors:

$$o_{ij} = \frac{\mathbf{w}_i \cdot \mathbf{w}_j}{\|\mathbf{w}_i\|\,\|\mathbf{w}_j\|}, \quad i \neq j, \tag{6}$$

classical Gram–Schmidt orthogonalization:

$$o_{ij} = \frac{\mathbf{w}_i \cdot \mathbf{w}_j}{\mathbf{w}_i \cdot \mathbf{w}_i}, \quad i \neq j, \tag{7}$$

and the dot product between weight vectors:

$$o_{ij} = \mathbf{w}_i \cdot \mathbf{w}_j, \quad i \neq j. \tag{8}$$

However, of the three approaches presented above, penalizing cosine of the angle between weight vectors Eq. (6) consistently yielded better results in validation experiments than the other two approaches (see Section 4).

Orthogonalization of RBM weight vectors is not explicitly connected to modeling the probability distribution of observations. However, it is not unusual to include additional regularization terms in RBMs when they are used to initialize deep networks: a good example is the sparsity penalty introduced in [15] to improve discriminative performance of pretrained deep networks.