Learning Class-relevant Features and Class-irrelevant Features via a Hybrid third-order RBM

Heng Luo
Shanghai
Jiao Tong University
hengluo@sjtu.edu

Ruimin Shen
Shanghai
Jiao Tong University
rmshen@sjtu.edu

Changyong Niu
Zhengzhou University
iecyniu@zzu.edu.cn

Carsten Ullrich
Shanghai
Jiao Tong University
ullrich_c@sjtu.edu.cn

Abstract

Restricted Boltzmann Machines are commonly used in unsupervised learning to extract features from training data. Since these features are learned for regenerating training data a classifier based on them has to be trained. If only a few of the learned features are discriminative other non-discriminative features will distract the classifier during the training process and thus waste computing resources for testing. In this paper, we present a hybrid third-order Restricted Boltzmann Machine in which class-relevant features (for recognizing) and class-irrelevant features (for generating only) are learned simultaneously. As the classification task uses only the class-relevant features, the test itself becomes very fast. We show that class-irrelevant features help class-relevant features to focus on the recognition task and introduce useful regularization effects to reduce the norms of class-relevant features. Thus there is no need to use weight-decay for the parameters of this model. Experiments on the MNIST, NORB and Caltech101 Silhouettes datasets show very promising results.

1 INTRODUCTION

Restricted Boltzmann Machines (RBM) (Hinton and Sejnowski, 1986) (Freund and Haussler, 1992) have become increasingly popular because of their excellent ability in feature extraction and been successfully applied in various application domains. Usually RBM are trained in an unsupervised manner by the Contrastive Divergence (CD) algorithm (Hinton, 2002). After training, each hidden unit in the RBM functions as a feature detector. Since these features capture most of the statistical structure in the observed data they can be used to construct sophisticated classifiers with excellent performance in several challenging classification tasks (Hinton and Salakhutdinov, 2006; Salakhutdinov and Hinton, 2009).

However, these features learned by RBMs in such an unsupervised manner may not be totally appropriate for classification tasks. Firstly, in some difficult classification tasks, observed data from different classes share a significant amount of features, thus left a few of the learned generative features being discriminative. In consequence a classifier based on all of the learned features can be very inefficient. Secondly, although features with larger norms can generate data better, the norms of these generative features could be too large for recognition tasks (see details in section 2). Using a strong weight-decay strategy might reduce the capacity of RBMs and restrict RBMs from learning regularities in observed data. On the other hand, using a weak weight-decay strategy may result in features with large norms, which have a negative impact on recognition tasks. Thus training RBMs need a carefully designed weight-decay strategy (Swersky, 2010).

To address these two issues, we try to seek compact discriminative features with relatively small norms. In this paper, we present a hybrid 3-order Restricted Boltzmann Machine, in which the hidden units are divided into two parts, class-relevant and class-irrelevant. Both class-relevant and class-irrelevant hidden units are trained to model the joint distribution of observed data, x and labels, y. The conditional distribution, $P(y|x)$ is only composed of the class-relevant hidden units. We also present a variant of the Contrastive Divergence algorithm for training this hybrid 3-order RBM. By using this CD variant, class-

Appearing in Proceedings of the 14^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2011, Fort Lauderdale, FL, USA. Volume 15 of JMLR: W&CP 15.

irrelevant hidden units can learn the shared features (which are probably unsuitable for recognition tasks) faster than class-relevant units. Class-relevant units are thus restricted from learning those features. Class-irrelevant units can also help class-relevant units to generate better negative data and thereby limit the growth of the class-relevant feature norms. Thus there is no need to use weight-decay for the parameters in the hybrid 3-order RBM. Empirically we find that by using only a few class-relevant units the classifier can achieve fairly good classification accuracies.

In the next section we give an introduction to RBMs and Contrastive Divergence. Then we explain why good generative features might not be good for recognition tasks. In section 3, we introduce the hybrid 3-order RBM. In section 4, the training algorithm based on Contrastive Divergence is presented. We also give an interpretation of the algorithm from the perspective of energy-based models. Section 5 discusses important related work. The experimental results based on the MNIST, NORB and Caltech101 Silhouettes datasets are shown in section 6.

2 RESTRICTED BOLZMANN MACHINES AND CONTRASTIVE DIVERGENCE

A Restricted Boltzmann Machine is a two layer neural network with one visible layer representing observed data and one hidden layer as feature detectors. Connections only exist between the visible layer and the hidden layer. Here we assume that both the visible and hidden units of the RBM are binary. The models below can be easily generalized to other types of units (Welling et al., 2005). The energy function of a RBM is defined as

$$E(x,h) = -\sum_{i,j} x_i h_j w_{ij} \tag{1}$$

where x_i and h_j denote the states of the i^{th} visible unit and the j^{th} hidden unit, while w_{ij} represents the strength of the connection between them. For simplicity, we omit the biases of the visible and hidden units.

Based on the energy function, we can define the joint distribution of (x,h),

$$P(x,h) = \frac{\exp(-E(x,h))}{Z} \tag{2}$$

where $Z = \sum_{x,h} \exp(-E(x,h))$.

Given the state of the visible units, the activation probability of the hidden unit is

$$\begin{aligned} P(h_j = 1|x) &= \text{sigmoid}(x^T w_{.j}) \\ &= \frac{1}{1+\exp(-x^T w_{.j})} \end{aligned} \tag{3}$$

where $w_{.j}$ denotes the j^{th} column of W. For $x^T w_{.j} = \cos(\alpha)\|w_{.j}\|_2\|x\|_2$ (α is the angle of the vector $w_{.j}$ and x), the activation probability can be interpreted as the similarity between x and the feature $w_{.j}$ in data space. As we can assume that all data has the same length, the activation probability is completely dependent on $\|w_{.j}\|_2$ and α. More specifically, when α is below a specific threshold (x is close enough to the direction of $w_{.j}$), x activates the hidden unit firmly. Increasing $\|w_{.j}\|_2$ increases the threshold. In consequence additional training data activates the hidden unit with a very high probability because of the larger threshold. Thus its feature will be less discriminative.

The activation probabilities of the visible units are similar, $P(x_i = 1|h) = \text{sigmoid}(w_{i.}h)$.

The marginal distribution over the visible units actually is a model of products of experts (Hinton, 2002),

$$P(x) = \frac{\prod_j (1+\exp(x^T w_{.j}))}{Z} \tag{4}$$

From equation 4 we can deduce that each expert will contribute probabilities according to the similarity between its feature and the data vector x.

The objective of generative training of a RBM is to model the marginal distribution of the visible units $P(x)$. To do this, we need to compute the gradient of the training data likelihood,

$$\begin{aligned} \frac{\partial \log P(x^{(n)})}{\partial \theta} &= -\Big\langle \frac{\partial E(x^{(n)},h)}{\partial \theta} \Big\rangle_{P(h|x^{(n)})} \\ &\quad + \Big\langle \frac{\partial E(x,h)}{\partial \theta} \Big\rangle_{P(x,h)} \end{aligned} \tag{5}$$

where $< . >_P$ is the expectation with respect to the distribution P. Hinton (2002) shows that we can get very good approximations to the second term when running the Gibbs sampler only k-step, initialized from the training data. Named Contrastive Divergence (CD), the algorithm updates the feature of the j^{th} hidden unit after seeing the training data $x^{(n)}$,

$$\Delta w_{.j} = P(h_j = 1|x^{(n)}) \cdot x^{(n)} - P(h_j = 1|x^{(n)-}) \cdot x^{(n)-} \tag{6}$$

$$x^{(n)-} = \text{sigmoid}(\sum_j \hat{h_j} w_{.j}) \tag{7}$$

where $\hat{h_j}$ is sampled from $P(h_j = 1|x^{(n)})$.

From equation 6 it can be seen that there is little learning when the reconstruction are very good ($x^{(n)-}$ is very similar to $x^{(n)}$). In other words, the CD algorithm aims at learning such features which can be used to regenerate training data well.

Unfortunately, the learned generative features may not be appropriate for a recognition task for two reasons. Firstly, only some of the generative features are discriminative. Lots of the generative features are local and learned from the parts of a training data that have high reconstruction errors (Hinton, 2002, section 12.2). These parts with reconstruction errors may be located anywhere in the training data. Some of them are relevant to a recognition task and some are not. For example, when a RBM is trained to model handwritten digits, some of features resemble different strokes and some of them look like "point" filters. The latter is probably shared by images from different classes and thus less discriminative than the former. Secondly, the generative features' norms may be too large to be suitable for a recognition task. To achieve a good reconstruction some specific visible units need to receive large (small) enough inputs to be turned on (off).There are two ways to ensure large (small) inputs, namely allowing some elements of features to become big (small) or using a large number of hidden units. Both ways have their own shortcoming. The former makes the feature norms too large to be discriminative. Although the latter keeps the feature norms small, the classifier trained on such a large number of features need a much longer time for training and testing.

In order to obtain discriminative features while at the same time to restrict the number of features as less as possible, we propose to divide hidden units into two parts, namely class-relevant hidden units and class-irrelevant hidden units. Class-relevant units learn from training data with the same class. Class-irrelevant hidden units, like hidden units in regular RBMs trained in an unsupervised manner, learn from all of the training data. These two kinds of units together compose a hybrid 3-order RBM.

3 HYBRID THIRD-ORDER RBM FOR CLASSIFICATION

We use a third-order Restricted Boltzmann machine (Sejnowski, 1986; Nair and Hinton, 2009) to model the joint distribution of observed data and labels. Furthermore, class-irrelevant units are introduced to restrict the class-relevant units to model the features shared by data from different classes. This model can be seen as a hybrid 3-order RBM and is illustrated in Figure 1. The energy function is

$$E(x,y,g,h) = -\sum_{i,j'} x_i g_{j'} w^0_{ij'} - \sum_{i,j,k} x_i h^k_j w^k_{ij} y_k \qquad (8)$$

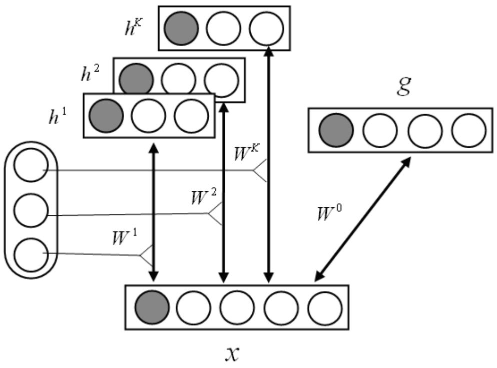

Figure 1: Third-order Restricted Boltzmann Machine with class-irrelevant units. The black circles are for implementing biases.

where h^k_j is the j^{th} hidden unit of class k, and $g_{j'}$ denotes the j'^{th} class-irrelevant hidden unit. Class labels y are K-dimensional binary vectors with 1-of-K activation. It can be seen that only parts of the parameters are related to the class labels. Again we omit the biases of the visible and hidden units. Based on the energy function, we define the joint distribution of (x, y) as,

$$P(x,y) = \frac{\sum_{h,g} \exp -E(x,y,g,h)}{Z} \qquad (9)$$

where Z is the normalization factor.

The activation probabilities of the hidden and visible units are straightforward

$$P(g_{j'} = 1|x) = \text{sigmoid}(x^T w^0_{.j'}) \qquad (10)$$

$$P(h_j = 1|x, y_k = 1) = \text{sigmoid}(x^T w^k_{.j}) \qquad (11)$$

$$P(x_i = 1|g, h, y_k = 1) = \text{sigmoid}(w^k_{i.} h^k + w^0_{i.} g) \qquad (12)$$

From equation 12 it can be seen that both class-relevant units and class-irrelevant units are used for generating data.

Because of the normalization factor, the joint distribution $P(x, y)$ is intractable. But the conditional distribution $P(y_k = 1|x)$ is tractable:

$$\begin{aligned} P(y_k = 1|x) &= \frac{\sum_h P(x, y_k = 1, h)}{\sum_{l=1}^K \sum_{\widetilde{h}} P(x, y_l = 1, \widetilde{h})} \\ &= \frac{\prod_j (1 + \exp(x^T w^k_{.j}))}{\sum_{l=1}^K \prod_j (1 + \exp(x^T w^l_{.j}))} \end{aligned} \qquad (13)$$

where the class-irrelevant features are canceled out. Equation 13 requires $O(N_V N_{RH} K)$ computation, where N_V, N_{RH} and K are the number of visible units, the number of class-relevant hidden units of each class and the number of classes, respectively. To ensure that testing takes place in an efficient manner, the number of class-relevant hidden units has to be confined, especially when the input data x is in a high dimensional space and/or the number of classes is large.

Equation 13 shows that this classifier is a model of products of experts. When x is similar to the feature of one specific hidden unit of class k, this unit will contribute probabilities on class k. As discussed in section 2, the features with large norms are less discriminative since the value of $x^T w^k_{\cdot j}$ could be dominated by the norm $\|w^k_{\cdot j}\|_2$. Furthermore, if some of the class-relevant features are shared by two or more different classes' data the model could be distracted. In the next section, we introduce a variant of the CD algorithm to train the hybrid 3-order RBM.

4 LEARNING ALGORITHM

Given the training data $\{(x^{(1)}, y^{(1)}), ..., (x^{(N)}, y^{(N)})\}$, the hybrid RBM is trained by maximizing the log likelihood $L = \sum_{n=1}^{N} \log P(x^{(n)}, y^{(n)})$. There is a need to compute the following gradient,

$$\frac{\partial L}{\partial \theta} = -\sum_{n=1}^{N} < \frac{\partial E(x^{(n)}, y^{(n)}, g, h)}{\partial \theta} >_{P(g,h|x^{(n)},y^{(n)})} + N < \frac{\partial E(x, y, g, h)}{\partial \theta} >_{P(x,y,g,h)} \tag{14}$$

The first term of equation 14 is tractable and can be computed effectively. We use a variant of the CD algorithm to approximate the second term. Given the training data $(x^{(n)}, y^{(n)})$, we sample $(\hat{g}^{(n)}, \hat{h}^{(n)})$ from $P(g|x^{(n)})$ and $P(h|x^{(n)}, y^{(n)})$. Then we take two steps to reconstruct the negative sample $(x^{(n)-}, y^{(n)-})$. Firstly, we fix $y^{(n)}$ and sample $x^{(n)-}$ from $P(x|y^{(n)}, \hat{g}^{(n)}, \hat{h}^{(n)})$. Secondly, we sample $y^{(n)-}$ from $P(y|x^{(n)-})$. We summary this process in Algorithm 1. To speed up the training process, we can divide the training data into mini-batches. Here N_b is the size of a mini-batch.

Based on Algorithm 1, the update of j'^{th} class-irrelevant unit's features is

$$\Delta w^0_{\cdot j'} = \frac{\lambda}{N_b} \sum_n P(g_{j'} = 1|x^{(n)}) \cdot x^{(n)}) - P(g_{j'} = 1|x^{(n)-}) \cdot x^{(n)-} \tag{15}$$

which is independent of the label, $y^{(n)}$.

The update of the j^{th} hidden unit's feature of class k

Algorithm 1

```
Input: training data $\{(x^{(1)}, y^{(2)}), ..., (x^{(N_b)}, y^{(N_b)})\}$,
learning rate $\lambda$ and $\theta \in (W^0, W^1, ..., W^K)$
for $n = 1$ to $N_b$ do
    $g^{(n)} = P(g|x^{(n)})$
    $h^{(n)} = P(h|x^{(n)}, y^{(n)})$
    Sample $\hat{g}^{(n)}$ from $P(g|x^{(n)})$.
    Sample $\hat{h}^{(n)}$ from $P(h|x^{(n)}, y^{(n)})$.
    Sample $x^{(n)-}$ from $P(x|\hat{g}^{(n)}, \hat{h}^{(n)}, y^{(n)})$.
    Sample $y^{(n)-}$ from $P(y|x^{(n)-})$.
    $g^{(n)-} = P(g|x^{(n)-})$
    $h^{(n)-} = P(h|x^{(n)-}, y^{(n)-})$
end for
$\theta = \theta - \frac{\lambda}{N_b} \sum_{n=1}^{N_b} \frac{\partial}{\partial \theta} E(x^{(n)}, y^{(n)}, g^{(n)}, h^{(n)}) - \frac{\partial}{\partial \theta} E(x^{(n)-}, y^{(n)-}, g^{(n)-}, h^{(n)-})$
```

is

$$\Delta w^k_{\cdot j} = \frac{\lambda}{N_b} \sum_{\{n|y^{(n)}_k=1\}} P(h^k_j = 1|x^{(n)}, y^{(n)}_k = 1) \cdot x^{(n)} - \sum_{\{n'|y^{(n')-}_k=1\}} P(h^k_j = 1|x^{(n')-}, y^{(n')-}_k = 1) \cdot x^{(n')-} \tag{16}$$

Assuming that the label of $x^{(n)}$ is l ($y^{(n)}_l = 1$), the corresponding negative sample is

$$x^{(n)-}_i = \text{sigmoid}(w^l_{i\cdot} \hat{h}^{(n)} + w^0_{i\cdot} \hat{g}^{(n)}) \tag{17}$$

From equation 15 and 16, both class-relevant and class-irrelevant hidden units learn from the parts of a training data that have high reconstruction errors. Both of them have a chance to learn features shared by data from different classes. However, these non-discriminative features probably appear more frequently in the training data than those features which are unique for one specific class's data. Class-relevant units thus learn the non-discriminative features more slowly than class-irrelevant units since in each learning epoch one specific class-relevant unit only learn from its own class's data. In other words, class-irrelevant units receive much more learning signals of these non-discriminative features than class-relevant units do and thus model them well before class-relevant units can. While the training process is going on, the reconstruction errors located in the non-discriminative parts are reduced quickly by the class-irrelevant units. In consequence the class-relevant units are restricted from learning in those parts.

From equation 17, by adopting a large number of class-irrelevant units, we can easily make the visible units receive large (small) enough inputs to achieve a good reconstruction and keep all of the hidden units' norms relatively small. On the other hand, since those class-irrelevant units do not appear in the classifier we still can keep the parameters of the classifier at a reasonable scale to ensure efficient testing.

Next we interpret our algorithm from an energy model's perspective. Without considering class-irrelevant units, class-relevant units can be represented as the following energy model

$$\text{Energy}(x, y) = -\sum_{k=1}^{K} y_k \sum_j \log(1 + \exp(x^T w^k_{\cdot j})) \quad (18)$$

We expect that the energy surfaces decided by equation 18 give low energies to areas around the training data set and high energies to all other areas. Since our final objective is to learn discriminative features for a recognition task, the energy surfaces need to have a local structure to achieve the above expectation. To obtain such a local structure, Ranzato et al. (2007) proposed to pull energies down on the training set and pull energies up on some contrastive samples. Formally, given training data pair $(x^{(n)}, y^{(n)})$ we need find the following contrastive sample

$$(x^*, y^*) = \text{argmin}_{\{x \in N(x^{(n)}), y\}} \text{Energy}(x, y) \quad (19)$$

where $N(x^{(n)})$ is the neighborhood of $x^{(n)}$.

Our learning algorithm actually performs this intuition. Equation 16 is the gradient of the following objective function

$$\max_W \sum_n \text{Energy}(x^{(n)-}, y^{(n)-}) - \text{Energy}(x^{(n)}, y^{(n)}) \quad (20)$$

Equation 16 is aimed at decreasing energies of the training data and increasing energies of the corresponding negative data. The negative data can be seen as a stochastic approximation to the contrastive sample in equation 19 and is generated in two steps (See Algorithm 1). These two steps are designed to make the negative data on average coming from the low energy areas which are near the training data. Furthermore, class-relevant and class-irrelevant units implicitly defined the neighborhood of equation 19. Better reconstruction imply smaller neighborhood. During the learning process, the class-irrelevant hidden units help the class-relevant units reconstructing the negative sample increasingly better (meaning $x^{(n)-}$ being increasingly similar to $x^{(n)}$). In other words, the neighborhood of $x^{(n)}$ in equation 19 is narrowed more quickly by using both the class-irrelevant and the class-relevant units than by using the latter alone. Thus the class-irrelevant units can help the class-relevant units focusing on the increasingly narrowed local area of the training data and finally speed-up the convergence of the learning process.

Since all weights are randomly initialized before the training process starts, the differences between the energy functions given the training data may be huge. That means before learning the classifier has already had a big bias towards the training data. Following (Nair and Hinton, 2009), we introduced a temperature T to the distribution $P(y|x)$ eliminating the bias and we can start learning with a vague conditional distribution.

$$P(y_k = 1|x) = \frac{\exp(-\text{Energy}(x, y_k = 1)/T)}{\sum_{l=1}^{K} \exp(-\text{Energy}(x, y_l = 1)/T)} \quad (21)$$

In algorithm 1 we use equation 21 as the conditional distribution. With the learning process going on the confidence in the learned conditional distribution increases and then gradually anneals the temperature.

5 RELATED WORK

Nair and Hinton (2010) present a third-order Restricted Boltzmann Machine as a top-layer for Deep Belief Nets applied for 3D object recognition. Our work is different from their work in two ways. Firstly, we introduce class-irrelevant units to make class-relevant features discriminative and restrict the growth of class-relevant features' norms. In (Nair and Hinton, 2010), there are no class-irrelevant units. This results in a significant larger number of hidden units in Nair's model than ours (see the next section for details). Secondly, the learning algorithms are different in the two models. In Nairs model given training data (x, y), the activation probabilities of hidden units of class y in Nair's model are computed first. $\hat{h}$ is sampled from $P(h|x, y)$. Then the negative class label y^- is sampled from $P(y|\hat{h})$. At last x^- is sampled from $P(x|\hat{h}, y^-)$. This inference process implies that the states of hidden units contain discriminative information. This also causes a large number of hidden units in their model.

Salakhutdinov and Hinton (2007) used a deep neural network to learn a nonlinear embedding. After pretraining the network, the codes of top layer are split into class-relevant and class-irrelevant parts. During the fine-tuning process only the class-relevant codes contribute to the Neighborhood Component Analysis (NCA) objective function. In essence the features of the deep network are found by unsupervised pretraining. The class-relevant codes are used to select discriminative features. However in our model there is no unsupervised pretraining process and all class-relevant features are learned directly from training data. Furthermore, in (Salakhutdinov and Hinton 2007) weight-decay for parameters is necessary during the pretraining process.

6 EXPERIMENTS

We applied our algorithm to the MNIST[1], NORB[2] and Caltech101 Silhouettes (28×28)[3] datasets.

Our model is trained by minimizing the negative log-likelihood $L = -\sum_{n=1}^{N} \log P(x^{(n)}, y^{(n)})$, which can be divided into two parts $-\sum_{n=1}^{N} \log P(y^{(n)}|x^{(n)})$ and $-\sum_{n=1}^{N} \log P(x^{(n)})$. While the training process is going on, the increase of the log likelihood is dominated by the second part. Therefore we stop the training process as soon as the errors on the training set do not decrease during 10 iterations.

6.1 The MNIST Experiments

The MNIST dataset consists of 60,000 training and 10,000 test images of ten handwritten digits (from 0 to 9). We performed model selection on a subset of MNIST, which contains 10,000 training and 1000 test images randomly selected from the 60,000 training images. In all experiments on MNIST, the learning rate and initial temperature were set to 0.0001 and 10. We configured the model to consist of 100 class-relevant units for each class and 400 class-irrelevant units (R100IR400). We compare this model to a second one called R140IR0, which is a third-order RBM with 140 class-relevant hidden units for each class and without class-irrelevant units. Then R140R0 has the same number of trainable parameters with R100IR400. As discussed above, since there are no class-irrelevant units the weights of R140IR0 need to be regularized by weight-decay. The parameter of weight-decay is selected by the subset of MNIST and set to 0.0002. Following (Larochelle and Bengio, 2008), we use stochastic gradient descent $(N_b = 1)$ to train these models.

Due to space reasons, we show only 100 class-irrelevant features in figure 2 and class 0's features of R100IR400 in figure 3.

As show in figure 2, some class-irrelevant features are quite local and detect an on-center off-surround structure (for example the one in column 1, row 1 and column 4, row 3). Those class-irrelevant features are obviously shared by data with different classes.

In contrast, all of the class-relevant features in figure 3 look similar to different shapes of handwritten zeros. To generate a picture of handwriting, the class-relevant features may focus on the shape of a specific class's digits and the class-irrelevant features may decide the strokes' thickness and the background of the picture.

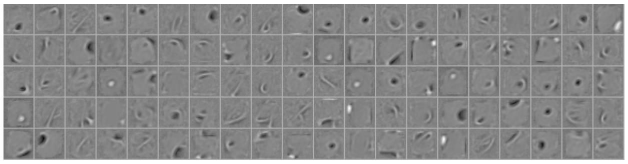

Figure 2: Random selected class-irrelevant features in R100IR400.

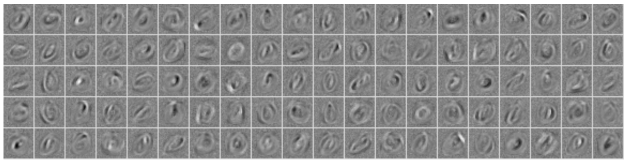

Figure 3: Features of class 0 in R100IR400.

We show 100 class-relevant features of class 0 from R140IR0 in figure 4. We can see the background of R140IR0's features are neater than those of R100IR400. The background of R100IR400's class-relevant features resemble meaningless noise. This means that with the help of class-irrelevant units the class-relevant units in R100IR400 avoid to learn the parts of the training data which do not contain discriminative information.

The results of our models and other state-of-the-art algorithms on MNIST are given in Table 1. R100IR400 achieves a similar error rate as Sparse HDRBM (Larochelle and Bengio, 2008), and outperforms regular RBM (Larochelle and Bengio, 2008), R140IR0, Support Vector Machine (SVM) (Decoste and Schölkopf, 2002) and Deep Belief Nets (DBN) (Hinton et al., 2006). Note that R100IR400 is trained generatively. It is very likely that R100IR400 does not make full use of the discriminative capacity. We have fine-tuned R100IR400 discriminatively by minimizing the conditional log-likelihood $(-\sum_{n=1}^{N} \log P(y^{(n)}|x^{(n)}))$ after the above generative training process. We randomly selected 10,000

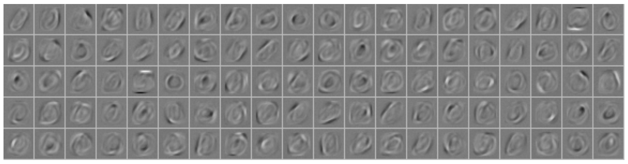

Figure 4: Features Of class 0 In R140IR0.

[1] http://yann.lecun.com/exdb/mnist/

[2] http://www.cs.nyu.edu/∼ylclab/data/norb-v1.0/

[3] http://www.cs.ubc.ca/∼bmarlin/data/

Table 1: Comparison of the classification performances on the MNIST dataset.

Model	Parameters for test	Error
RBM	4,770,794	3.39%
R140IR0	1,099,784	1.88%
SVM	-	1.4%
DBN	1,665,794	1.25%
Sparse HDRBM	2,385,794	1.16%
R100IR400	785,784	1.14%
DBM	904,294	0.95%

smaples in MNIST training data set as the validation set to decide the number of sweeps for fine-tuning. We fine-tuned the model by using the method of conjugate gradients[4] on mini-batches containing 1000 samples. Three line searches were performed for each mini-batch in each epoch. After fine-tuning, the test error of R100IR400 is **1.04%**, which is better than the result of Sparse HDRBM and worse than the best published result on the permutation-invariant version of the MNIST task achieved by DBM. Training R100IR400 in Matlab 2010b on an Intel Xeon 2.4GHz machine takes 38 hours. However, training a DBM[5] on the same machine takes almost 3 days.

In our approach, since the class-irrelevant units restrict the class-relevant units to model features shared by data from different classes, by using few class-relevant units the classifier have achieved fairly good classification accuracies. From Table 1, compared with other methods, our models used for classification have the fewest parameters, which amounts to about half of the parameters of DBN and one third of the parameters of Sparse HDRBM.

Furthermore, to quantitatively illustrate the regularization effects introduced by class-irrelevant units, we show the average norms of the class-relevant and class-irrelevant units during the learning process in figure 5.

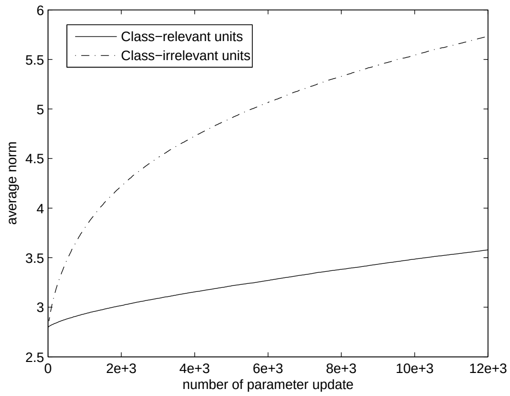

Figure 5: The average norms of the class-relevant and class-irrelevant units.

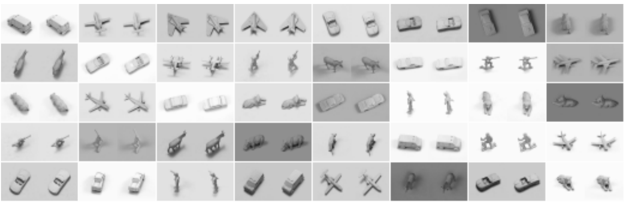

Figure 6: Some examples in the NORB dataset.

6.2 The NORB Experiments

The NORB dataset contains stereo image pairs of 50 different 3D toy objects with 10 objects in each of five generic classes (four-legged animals, humans, airplanes, trucks and cars). Each object is captured from different viewpoints and under various lighting conditions. The training set contains 24,300 stereo image pairs of 25 objects, 5 per class, while the test set contains 24,300 stereo pairs of the remaining, different 25 objects. The goal is to classify each previously unseen object into its generic class. Some examples in NORB are shown in figure 6.

The NORB dataset is much more challenging than MNIST because of the fact that the only useful and discriminative features are the shapes of the different objects while all the other parameters that affect the appearance are class-irrelevant.

To model raw pixel data of the NORB dataset, we need to use Gaussian units. However, learning a RBM with Gaussian units is very slow[6]. To speed up the training process, we divided the training data into mini-batches, and updated the weights after each mini-batch. We set the size of mini-batches $N_b = 100$ at the first 40 sweeps through the training set and then change the size to $N_b = 1000$ for the following sweeps. Furthermore, we subsampled the images from 96×96 to 32×32. This results in 24,300 2048-dimensional vectors in the training set and an equal number of vectors in the test set. We rescale all training data to have zero mean and unit variance (all test data are rescaled by the means and variances of training data).

We performed model selection on the subset of NORB, which contains 5,000 training and 5,000 test images randomly selected from the 24,300 training images. In all experiments on NORB, the learning rate and initial temperature were set to 0.001 and 15. The best classification results on the subset of NORB are given

[4] We used Carl Rasmussen's "minimize" code available at www.kyb.tuebingen.mpg.de/bs/people/carl/code.

[5] We used Ruslan Salakhutdinov's codes available at http://www.mit.edu/∼rsalakhu/DBM.html.

[6] The details of using Gaussian units in RBM can be found in (Taylor et al. 2006).

by 100 class-relevant hidden units for each class and 6000 class-irrelevant hidden units (R100IR6000).

It came to us as a surprise that the best classification results on the subset of NORB were achieved by using only 100 class-relevant hidden units for each class. That might be due to the fact that the NORB are more challenging than MNIST because there are more class-irrelevant features in NORB. This model (R100IR6000) achieves a very good error rate of **9.7%** on the test set. Similar to the procedure for the MNIST experiments, we discriminatively fine-tuned R100IR6000. We randomly selected 4300 image pairs in NORB training data set as the validation set to decide about the number of sweeps for fine-tuning. After fine-tuning, the test error rate of R100IR6000 is **8.9%**. Previous result were 11.6% achieved by SVM's (Bengio and LeCun 2007), 18.4% achieved by the K-nearest neighbours (LeCun et al. 2004) and 10.8% achieved by Deep Boltzmann Machine (Salakhutdinov and Hinton, 2009). Furthermore, the SVM contains about 6,800 support vectors and the DBM is a 3-layer Boltzmann Machine (the bottom layer as a preprocessed layer to convert NORB data from real vectors to binary vectors) with 12,000 hidden units in total. Our model for classification task only contains 500 hidden units. This results in a very fast test speed: testing all 24,300 image pairs takes only 9 seconds in Matlab 2010b on an Intel 2.4Ghz processor. In addition, the learning process of R100IR6000 stopped after trained for 56 sweeps through the training set, which takes about 12 hours. This is very fast considering that in order to obtain an good RBM with Gaussian units on NORB requires 500 training sweeps (Salakhutdinov and Hinton, 2009).

Nair and Hinton (2010) preprocessed images in NORB by using a "foveal" image representation. The foveal representation reduces the dimensionality of an original stereo-pair in NORB from 18432 to 8976. To enable a comparison with their method, we also preprocessed data in this way and trained a hybrid third-order RBM with 100 class-relevant hidden units for each class and 8000 class-irrelevant hidden units(R100IR8000). R100IR8000 achieves a error rate of **14.2%**. The result of Nair's model is 20.8%. In addition, Nair's model for classification has 179,548,976 parameters, a significantly greater amount than in our setting, 79,313,476 trainable parameters and only 4,407,796 parameters used for classification in R100IR8000.

6.3 The Caltech101 Silhouettes Experiments

Caltech101 Silhouettes is a binary image data set derived from CalTech101[7] by Marlin and et al. (2010).

Figure 7: Caltech101 Silhouettes data sample.

Some examples in Caltech101 Silhouettes are shown in figure 7. The data set includes 4100 examples with at least 20, and at most 100 examples from each class in the training set, 2264 examples in the validation set and 2307 examples in the test set. The Caltech101 Silhouettes data set is very different from MNIST and NORB. It contains a significant larger number of classes (101 in total) but much fewer samples for each class than MNIST and NORB. As discussed in section 3, it is very important to restrict the number of class-relevant units when the number of classes is large.

We perform model selection on the validation set. The learning rate and initial temperature were set to 0.1 and 15. The best classification results on the validation set are given by 50 class-relevant hidden units for each class and 500 class-irrelevant hidden units (R50IR500). We trained this model (R50IR500) on a new training set composed by the original training set and the validation set. To speed up the training process, we divided the training data into mini-batches (the size of mini-batches $N_b = 100$). R50IR500 achieves an error rate of **28.6%** on the test set. Since there are much less examples in the training set of Caltech101 Silhouettes than MNIST and NORB, we do not further discriminatively fine-tune this model to avoid overfitting. The learning process of R50IR500 takes about 3 hours. Marlin's result on this data set is about 34.2%.

7 CONCLUSIONS

In this paper, we described a hybrid third-order RBM that learns class-relevant and class-irrelevant features simultaneously. Classification uses only the class-relevant features. It leads to a lightweight classifier.

Because in the training data the shared features appear more frequently than the discriminative features, class-irrelevant hidden units can learn the shared features faster than class-relevant units. We showed that the class-relevant features become discriminative and compact. In addition, class-irrelevant units introduce useful regularization effects to restrict the growth of norms of class-relevant features. Thus there is no need to use weight-decay for the parameters in this hybrid 3-order RBM.

[7] www.vision.caltech.edu/Image_Datasets/Caltech101

Acknowledgements

This work was partially supported by National Natural Science Foundation of China (61071154, 60901078 and 60873132) and Zhengzhou Science & Technology (10LJRC189).

References

Y. Bengio and Y. LeCun (2007). Scaling learning algorithms towards AI. *Large-Scale Kernel Machines.*

D. Decoste and B. Schölkopf (2002). Training invariant support vector machines. *Machine Learning*, 46(1/3):161.

Y. Freund and D. Haussler (1992). Unsupervised learning of distributions of binary vectors using 2-layer networks. In *Advances in Neural Information Processing Systems*, volume 4, pages 912-919.

G. E. Hinton (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1711-1800.

G. E. Hinton and R. R. Salakhutdinov (2006). Reducing the dimensionality of data with neural networks. *Science*, 313:504-507.

G. E. Hinton, S. Osindero and Y. Teh (2006). A fast learning algorithm for deep belief nets. *Neural Computation* 18:1527-1554.

G. E. Hinton and T. J. Sejnowski (1986). Learning and relearning in boltzmann machines. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition.* Volume 1: Foundations. MIT Press.

H. Larochelle and Y. Bengio (2008). Classification using discriminative restricted boltzmann machines. In *Proceedings of the 25th international conference on Machine learning.* ACM.

Y. LeCun, F. Huang, and L. Bottou (2004). Learning methods for generic object recognition with invariance to pose and lighting. In *CVPR (2)*, pages 97-104.

B. M. Marlin, K. Swersky, B. Chen and N. de Freitas (2010). Inductive principles for Restricted Boltzmann Machine learning. In *Proceedings of The Thirteenth International Conference on Artificial Intelligence and Statistics*, 9:509-516.

V. Nair and G. E. Hinton (2009). Implicit Mixtures of Restricted Boltzmann Machines. In *Advances in Neural Information Processing Systems 21*, MIT Press.

V. Nair and G. E. Hinton (2010). 3D Object Recognition with Deep Belief Nets. In *Advances in Neural Information Processing Systems 22*, MIT Press.

M. Ranzato, Y. Boureau, S. Chopra, and Y. LeCun (2007). A unified energy-based framework for unsupervised learning. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, Omnipress.

R. R. Salakhutdinov and G. Hinton (2007). Learning a nonlinear embedding by preserving class neighbourhood structure. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, Omnipress.

R. R. Salakhutdinov and G. Hinton (2009). Deep Boltzmann machines. In *Proceedings of The Twelfth International Conference on Artificial Intelligence and Statistics*, Vol. 5, pp. 448-455.

T. J. Sejnowski (1986). Higher-order Boltzmann machines. *AIP Conference Proceedings*, 151(1):398-403.

K. Swersky, B. Chen, B. Marlin, and N. de Freitas (2010). A Tutorial on Stochastic Approximation Algorithms for Training Restricted Boltzmann Machines and Deep Belief Nets. *Information Theory and Applications (ITA) Workshop*

G. W. Taylor, G. E. Hinton and S. Roweis (2006). Modeling human motion using binary latent variables. In *Advances in Neural Information Processing Systems*, MIT Press.

M. Welling, M. Rosen-Zvi and G. E. Hinton (2005). Exponential family harmoniums with an application to information retrieval. In *Advances in Neural Information Processing Systems 17*, MIT Press.