Improving Deep Representation Learning with Complex and Multimodal Data

by

Kihyuk Sohn

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering:Systems)
in the University of Michigan
2015

Doctoral Committee:

Assistant Professor Honglak Lee, Chair
Professor Alfred O. Hero III
Professor Benjamin Kuipers
Associate Professor Clayton D. Scott

To the Lord who save me from the dark and let all these happen.

ACKNOWLEDGEMENTS

First of all, I would like to express my sincere gratitude to my advisor, Professor Honglak Lee, for his guidance, encouragement and continuous support throughout my graduate studies. Luckily for me, I was admitted as his first Ph.D. student when he started his career as an assistant professor at the University of Michigan, and had more than enough opportunities to discuss, execute, and publish our work over the years. All my achievements have benefited from his knowledge, insight and inspiration. I would like to extend my appreciation to other members of my dissertation committee, Professor Alfred O. Hero III, Professor Benjamin Kuipers, and Professor Clayton D. Scott for their motivating suggestions and valuable comments. My thanks also extend to many other faculty members and staffs in the Department of Electrical Engineering and Computer Science for their teaching, advice and kindness.

My sincere appreciation is extended to Professor Erik G. Learned-Miller of the University of Massachusetts Amherst for his invaluable advice and support, which played a crucial role in motivating the second part of the Ph.D. thesis. Many thanks to my colleagues, Scott Reed, Yuting Zhang, Wenling Shang, Ruben Villegas, Junhyuk Oh, Xinchen Yan, Ye Liu, Guanyu Zhou, Chansoo Lee (from UM machine learning group), and Andrew Kae (from UMass), for their valuable discussions and warm friendship. To my friends that I have met in Ann Arbor, thanks for the memories and great times we shared over the years. Specifically, I would like to thank Sungjoon Park, Donghwan Kim, Dae Yon Jung, Kyunghoon Lee and the members of "Invaders", Jongjin Park, Taehyung Kim, Yelin Kim, who shared my 7 years of journey at Michigan from begin-

ning to end. I am also thankful to Gyouho Kim, Suyoung Bang, and the members of our football club “A.K. United” for our team workouts and other moments we shared. I am truly indebted to the members of Korean Presbyterian Church of Ann Arbor, including Pastor Jae Joong Hwang, executive board members, Hyungmin Baek, Hanna Song, Myungjoon Choi, and the “For God” worship team and the “Hesed” choir team, who helped enriching my life in God.

I would like to express my deepest gratitude to my family for their countless love and support. My parents have always been a source of comfort and hope in difficult times. I also thank my sister, brother-in-law, and my lovely nephew, Onew Yang, for being there. Your presence has been great pleasure and encouragement.

Lastly, I gratefully acknowledge the financial support from the National Science Foundation (contract No. 1247414), Office of Naval Research (contract No. N00014-13-1-0762) and Samsung Digital Media and Communication Lab, and the donation of GPUs from NVIDIA.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

LIST OF APPENDICES

Appendix

ABSTRACT

Improving Deep Representation Learning with Complex and Multimodal Data

by

Kihyuk Sohn

Chair: Honglak Lee

Representation learning has emerged as a way to learn meaningful representation from data and made a breakthrough in many applications including visual object recognition, speech recognition, and text understanding. However, learning representation from complex high-dimensional sensory data is challenging since there exist many irrelevant factors of variation (e.g., data transformation, random noise). On the other hand, to build an end-to-end prediction system for structured output variables, one needs to incorporate probabilistic inference to properly model a mapping from single input to possible configurations of output variables. This thesis addresses limitations of current representation learning in two parts.

The first part discusses efficient learning algorithms of invariant representation based on restricted Boltzmann machines (RBMs). Pointing out the difficulty of learning, we develop an efficient initialization method for sparse and convolutional RBMs. On top of that, we develop variants of RBM that learn representations invariant to data transformations such as translation, rotation, or scale variation by pooling the filter responses of input data after a transformation, or to irrelevant patterns such as random or structured noise, by jointly performing feature selection and feature learning. We

demonstrate improved performance on visual object recognition and weakly supervised foreground object segmentation.

The second part discusses conditional graphical models and learning frameworks for structured output variables using deep generative models as prior. For example, we combine the best properties of the CRF and the RBM to enforce both local and global (e.g., object shape) consistencies for visual object segmentation. Furthermore, we develop a deep conditional generative model of structured output variables, which is an end-to-end system trainable by backpropagation. We demonstrate the importance of global prior and probabilistic inference for visual object segmentation. Second, we develop a novel multimodal learning framework by casting the problem into structured output representation learning problems, where the output is one data modality to be predicted from the other modalities, and vice versa. We explain as to how our method could be more effective than maximum likelihood learning and demonstrate the state-of-the-art performance on visual-text and visual-only recognition tasks.

CHAPTER I

Introduction

1.1 Motivation

In recent years, representation learning algorithms (e.g., clustering [2, 32, 74, 140], sparse coding [15, 147, 143, 101, 81], restricted Boltzmann machine (RBM) [117], autoencoders [9, 91, 73, 6], and deep learning [46, 111, 84, 11, 78, 68]) have emerged as a way to learn useful features from unlabeled and labeled data.

Representation learning algorithms can be classified into two categories, supervised and unsupervised learning, depending on the use of supervision during the training. In unsupervised learning, the goal is to learn features that capture underlying structures (e.g., statistical dependencies such as co-occurrence) in data, and features that are learned complement or sometimes outperform manually designed domain-specific features (e.g., SIFT [88], HOG [22] in computer vision, MFCC [23] in speech processing). Furthermore, these methods make minimal assumptions about the data, and they have been successfully applied to many tasks in different domains, including visual recognition [68], speech processing [103], and text understanding [34]. However, learning representation from complex unlabeled sensory data is still a very challenging problem due to many reasons; first, raw input data is usually very noisy and highly variable, and does not provide useful information for the target task. Second, there are many factors of variation that are not necessarily relevant to the target tasks, such

as low-level domain-specific transformation (e.g., pixel-level translation or rotation) or external sources of variation (e.g., lighting condition). To achieve a good recognition performance, it is important to learn robust feature representations that are *invariant* to such kinds of irrelevant intrinsic or extrinsic factors of variation. Finally, it is often the case that the unsupervised learning algorithms with high expressive power (e.g., RBM) are difficult to train and thus require much of an expert's knowledge and efforts to train.

On the other side of the story, the supervised representation learning algorithms have made a significant progress in recent years. Specifically, the convolutional neural network (CNN) has shown impressive performance on large-scale visual recognition tasks [68]. Behind the scenes, there are several ingredients that contributed to make a breakthrough: 1) powerful GPUs that can train very deep (convolutional) neural networks with a reasonable time cost, 2) huge number of labeled training examples such as the ImageNet database [24], 3) stochastic gradient descent with advanced optimization techniques (e.g., adaptive learning rate schedule algorithms [128, 26], rectified linear units [149, 97], dropout [123]). Although deep neural networks have been so successful for simple recognition tasks, not much has been shown yet for complex structured output prediction problems. Unlike simple recognition problems, the distribution of structured outputs have multiple modes, i.e., there could be several possible outcomes that can be derived from the same input, and deep neural networks, which are extremely powerful function approximators, may not be the optimal for modeling complex outputs. Similar challenge can often be found in multimodal joint representation learning problems, where we have input data from multiple channels during the training. The promise of multimodal representation learning is that the performance improvement is guaranteed over the single data modality counterpart. However, it becomes non-trivial when we have a missing data modality for testing, and it is important to learn a generative model that has an ability to predict or reason about a missing data modality conditioned on

the observation.

In this thesis, we aim to solve the following research questions to build a robust and intelligent agent that can effectively learn representations from complex and multiple heterogeneous data sources:

1. How to make the learning procedure of the highly expressive representation learning methods a *black-box* by avoiding extensive hyperparameter search?

2. How to learn representations that are invariant to intrinsic data transformation from complex sensory data?

3. How to learn representations that are robust to irrelevant input patterns or random noise?

4. How to develop a generic supervised learning algorithm for structured output prediction that incorporates long-range, higher-order interactions among output variables?

5. How to learn a better joint representation of multiple heterogeneous data that can reason about a missing data modality?

6. How to develop an end-to-end system for structured output representation learning and prediction, and multimodal representation learning with deep convolutional neural networks?

1.2 Organization of the Thesis

This thesis is organized in 8 chapters including the introduction (Chapter I), and the conclusion and future work (Chapter VIII). The main chapters (II – VII) are divided into 2 parts, 3 chapters each. The first part (Chapter II, III, IV) discusses on efficient learning algorithms of invariant feature representations from complex sensory data. We

change gears in the second part (Chapter V, VI, VII) towards learning representations of structured output or multimodal data using (a combination of) conditional generative objectives. We briefly state the problem, our approach and contributions for each chapter.

Chapter II. Efficient Learning of Sparse, Distributed, Convolutional Feature Representations for Object Recognition.

Informative feature representations are important for achieving state-of-the-art performance in machine learning tasks. The RBM has been successfully applied to automatically learn useful patterns from large amount of unlabeled data. Although it has a great potential due to its rich expressive power and capability to build a deep network, the difficulty of training RBMs has been a barrier to their wide use. In this chapter, we address this difficulty by showing the connections between mixture models and RBMs and deriving an efficient training method for RBMs from these connections. Along with this efficient training, we evaluate the importance of convolutional training that can capture a larger spatial context with less redundancy, as compared to non-convolutional training. Overall, our method achieves state-of-the-art performance on visual object recognition benchmarks.

Chapter III. Learning Invariant Representations with Local Transformations.

The difficulty of developing representation learning algorithms that are robust to data transformations (e.g., scale, rotation, or translation) has been a challenge in many applications (e.g., object recognition problems). In this chapter, we address the problem of learning transformation invariant features by introducing the transformation matrices into the energy function of the RBMs. The proposed transformation-invariant RBMs not only learn the diverse patterns by explicitly transforming the weight matrix, but

they also achieve the invariance of the representation via probabilistic max pooling of hidden units over the set of transformations. We evaluate our algorithm on several benchmark on visual recognition, such as the variations of MNIST, or CIFAR-10 and STL-10, as well as the customized digit datasets with significant transformations, and show competitive classification performance to the state-of-the-art. Besides the image data, we apply our method to phone classification task on the TIMIT database to show the wide applicability of our proposed algorithms to other domains, also achieving state-of-the-art performance.

Chapter IV. Learning and Selecting Features Jointly with Point-wise Gated Boltzmann Machines.

Learning useful high-level features is still challenging when the data contains a significant amount of irrelevant patterns. In this chapter, we propose a *point-wise gated Boltzmann machine*, a unified generative model that combines feature learning and feature selection. Our model performs not only feature selection on learned high-level features (i.e., hidden units), but also *dynamic feature selection* on raw features (i.e., visible units) through a gating mechanism. For each example, the model can adaptively focus on a variable subset of visible nodes corresponding to the task-relevant patterns, while ignoring visible units corresponding to the task-irrelevant patterns. In experiments, our method achieves improved performance over state-of-the-art in several visual recognition benchmarks.

Chapter V. Augmenting CRFs with Boltzmann Machine Priors for Structured Output Prediction.

CRFs provide powerful tools for building models to label image segments. They are particularly well-suited to model local interactions among adjacent regions (e.g., superpixels). However, CRFs are limited in dealing with complex, global (long-range)

interactions between regions. Complementary to this, RBMs can be used to model global shapes produced by segmentation models. In this chapter, we present a new model that combines these two network types to build a state-of-the-art region labeler. Although the CRF is a good baseline labeler, we show how an RBM can be added to the architecture to provide a *global* shape bias that complements the local modeling provided by the CRF. We demonstrate the labeling performance for the parts of complex face images from the Labeled Faces in the Wild data set. This hybrid model produces results that are both quantitatively and qualitatively better than the CRF alone. In addition, we demonstrate that the hidden units in the RBM portion of our model can be interpreted as face attributes that have been learned without any attribute-level supervision.

Chapter VI. Improved Multimodal Deep Learning with Variation of Information.

It is important to capture high-level associations between multiple data modalities with a compact set of latent variables, and deep learning has been successfully applied to this problem of multimodal representation learning. Nonetheless, there still remains an important question how to learn a good association between multiple data modalities, in particular, to reason about the missing data modalities in the testing time. In this chapter, we propose a novel multimodal representation learning objective that explicitly aims this goal. Instead of maximum likelihood learning, we train the networks to minimize the *variation of information*, an information theoretic measure that computes the information distance between data modalities. In experiments, we demonstrate the state-of-the-art visual-textual and visual recognition performance on MIR-Flickr database and PASCAL VOC 2007 database.

Chapter VII. Learning to Predict Structured Outputs using Stochastic Convolutional Networks.

To build an end-to-end system for structured output prediction one needs to incorporate probabilistic inference, as it may not be a simple many-to-one function approximation problem (e.g., recognition and classification), but could be a task of mapping input to many possible outputs. In this chapter, we propose a stochastic convolutional neural networks with Gaussian latent variables for structured output prediction and representation learning. In light of recent development in variational inference and learning of directed graphical models [62, 107, 63], we propose a conditional variational auto-encoder (condVAE). We demonstrate the importance of stochastic neurons in modeling the distribution with multiple major modes using the toy example of MNIST database. In addition, we demonstrate the effectiveness of our proposed model on several image segmentation and region labeling database.

1.3 List of Publications

Here, we enumerate the list of publications relevant to each chapter:

[1] **Efficient Learning of Sparse, Distributed, Convolutional Feature Representations for Object Recognition**. **Kihyuk Sohn**, Dae Yon Jung, Honglak Lee, and Alfred Hero III. In *Proceedings of the International Conference on Computer Vision*, 2011. (Chapter II)

[2] **Learning Invariant Representations with Local Transformations**. **Kihyuk Sohn** and Honglak Lee. In *Proceedings of the International Conference on Machine Learning*, 2012. (Chapter III)

[3] **Learning and Selecting Features Jointly with Point-wise Gated Boltzmann Machines**. **Kihyuk Sohn**, Guanyu Zhou, Chansoo Lee, and Honglak Lee. In *Proceedings of the International Conference on Machine Learning*, 2013.

(Chapter IV)

[4] **Augmenting CRFs with Boltzmann Machine Shape Priors for Image Labeling**. **Kihyuk Sohn***, Andrew Kae*, Honglak Lee, and Erik Learned-Miller. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013 (* indicates equal contribution) (Chapter V)

[5] **Improved Multimodal Deep Learning with Variation of Information**. **Kihyuk Sohn**, Wenling Shang, and Honglak Lee. In *Advances in Neural Information Processing Systems*, 2014 (Chapter VI)

There are few more publications that I have published during my Ph.D. years:

[6] **Online Incremental Feature Learning with Denoising Autoencoders**. Guanyu Zhou, **Kihyuk Sohn**, and Honglak Lee. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2012.

[7] **Learning to Disentangle Factors of Variation with Manifold Interaction**. Scott Reed, **Kihyuk Sohn**, Yuting Zhang and Honglak Lee. In *Proceedings of the International Conference on Machine Learning*, 2015.

[8] **Improving Object Detection with Deep Convolutional Networks via Bayesian Optimization and Structured Prediction**. Yuting Zhang, **Kihyuk Sohn**, Ruben Villegas, Gang Pan and Honglak Lee. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015

For reproducible research, the code will be made available in my personal website: `https://sites.google.com/site/kihyuksml/`.

CHAPTER II

Efficient Learning of Sparse, Distributed, Convolutional Feature Representations

2.1 Introduction

Object recognition poses a significant challenge due to the high pixel-level variability of objects in images. Therefore, having higher-level, informative image features is a necessary component for achieving state-of-the-art performance in object classification and detection. In the last decades, many efforts have been made to develop feature representations that can provide useful low-level information from images [88, 22]. However, these feature representations are often hand-designed and require significant amounts of domain knowledge and human labor.

Therefore, there has been much interest in developing unsupervised and supervised feature learning algorithms for image representations that address these difficulties. Notable successes include clustering [2, 32, 74, 140], sparse coding [15, 147, 143], and deep learning methods [46, 9, 91, 60]. These methods are nonlinear encoding algorithms that provide new image representations from inputs. For instance, unsupervised learning algorithms (e.g., sparse coding [101]) can learn representations for low-level descriptors (e.g., SIFT, HOG) and provide discriminative features for visual recognition [143]. From another perspective, these methods can be viewed as generative models with la-

tent variables that learn salient structures and patterns from inputs. In this view, the posterior probabilities of the latent variables can be used as features for discriminative tasks.

Although recently developed models provide powerful feature representations for visual recognition, some of these models are difficult to train, which has been a barrier to their wide use in many applications. For example, while the RBM has rich expressive power and capability to build a deep network, it is difficult to train due to its intractable partition function and the need to tune many hyperparameters through expensive cross-validation.

In this section, we investigate *black-box* training of RBMs. The main idea of our approach is to examine theoretical links among the unsupervised learning algorithms and take advantage of simple models to train more complicated models. We provide a theoretical analysis showing the equivalence between GMM and Gaussian RBM under specific constraints. This link has far-reaching implications on existing algorithms. For example, sparse RBMs [82] can be viewed as an approximation to a relaxation of clustering algorithms, and thus can provide richer image representations than clustering methods. Using these equivalence and implications, we enhance the training of RBMs by utilizing Kmeans as a way of initializing the model parameters of RBMs. This allows for faster training and greater classification performance. We evaluate clustering methods and sparse RBMs on standard computer vision benchmarks, showing that sparse RBMs outperform clustering algorithms by allowing *distributed and less sparse* encoding.

Furthermore, we provide a simple connection between CRBM and non-convolutional RBMs. For example, the CRBM becomes equivalent to its non-convolutional counterpart when the convolution filter size is 1 (i.e., no spatial context). Not surprisingly, the CRBM thus can capture larger spatial contexts and reduce the redundancy of feature representations.

Based on our efficient training method, we systematically evaluate the performance of CRBMs on standard object recognition benchmarks, such as Caltech 101 and Caltech 256. We also provide an analysis of hyperparameters, such as target sparsity and convolutional filter size, to demonstrate the effectiveness of sparse, distributed, convolutional feature learning. Overall, our approach leads to enhanced feature representations that outperform other learning-based encoding methods [143, 135] and achieve state-of-the-art performance.

The main contributions of this section are as follows:

- We provide a theoretical analysis showing an equivalence between mixture models and RBMs with specific constraints. We further show that sparse RBMs can be viewed as an approximation to a relaxation of such mixture models.
- Using these connections, we propose an efficient training method for sparse RBMs and CRBMs. To the best of our knowledge, this is the first work showing that the RBM can be trained with almost no hyperparameter tuning to provide classification performance similar to or significantly better than GMM.
- We evaluate the importance of convolutional training that can capture larger spatial contexts with less redundancy (compared to non-convolutional training). Specifically, we learn a feature representation based on SIFT and CRBM. In the experiments, we show that such convolutional training provides a much better representation than its non-convolutional counterparts.
- Overall, our method achieves state-of-the-art performance on both Caltech 101 and 256 datasets using a single type of feature.

2.2 Related Work

Recently, researchers have tried to improve image features for object classification via unsupervised learning. A common unsupervised feature learning framework for classification is as follows: (1) densely extract image descriptors (e.g., SIFT [88] or

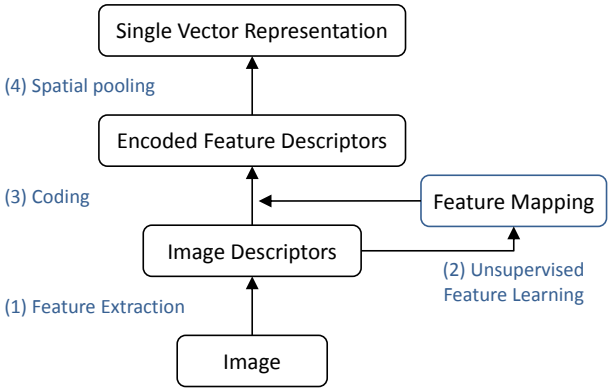


Figure 2.1: Pipeline for constructing features in object recognition.

HOG [22]); (2) train a feature mapping using an unsupervised learning algorithm; (3) once the feature mapping is learned, encode the descriptors to obtain "mid-level features" [15, 143]; (4) pool the single vector from multi-scaled sub-regions (e.g., spatial pyramid matching [74]) that characterize the entire image. The representations are then provided as inputs for linear or nonlinear classifiers (e.g., support vector machines). This pipeline is shown as a diagram in Figure 2.1.

Indeed, advanced encoding algorithms for image descriptors can provide significant improvements in object recognition. For example, Yang et al. [143] applied patch-based (non-convolutional) sparse coding on densely extracted SIFT descriptors to obtain sparse feature representations (ScSPM). Similarly, Wang et al. [135] proposed LLC based on locality. Boureau et al. [15] also used sparse coding, but they considered macrofeatures, which encode neighboring low-level descriptors to incorporate the spatial information. While these models are *not* trained convolutionally, we use the CRBM [83] as an unsupervised learning algorithm on top of the SIFT descriptors. Our feature representation is robust to translation variations of images and effectively captures the larger spatial context, as shown in the experiments.

Convolutional extensions of unsupervised learning algorithms, such as sparse coding and RBMs, have been successful in developing powerful image representations. For example, [148] and [60] developed algorithms for convolutional sparse coding, which

approximately solves the L_1-regularized optimization problem to minimize the reconstruction error between the data and the higher layer features convolved with the filters. Our approach is different from these methods in that we used CRBM instead of convolutional sparse coding. Further, we verified the advantage of convolutional training through the experimental comparison between convolutional and non-convolutional training.

Compared to sparse coding, the RBM can compute posterior probabilities in a feed-forward way, which is usually orders of magnitude faster. This computational efficiency provides a significant advantage over sparse coding since it scales up to a much larger number of codes. Furthermore, CRBMs are amenable to GPU computation resulting in another order of magnitude speedup.

2.3 Preliminaries

2.3.1 Restricted Boltzmann machines

The restricted Boltzmann machine is a bipartite, undirected graphical model with visible (observed) units and hidden (latent) units. The RBM can be understood as an MRF with latent factors that explains the input visible data using binary latent variables. The RBM consists of visible data $\mathbf{v}$ of dimension L that can take real values or binary values, and stochastic binary variables $\mathbf{h}$ of dimension K. The parameters of the model are the weight matrix $\mathbf{W} \in \mathbb{R}^{L \times K}$ that defines a potential between visible input variables and stochastic binary variables, the biases $\mathbf{c} \in \mathbb{R}^L$ for visible units, and the biases $\mathbf{b} \in \mathbb{R}^K$ for hidden units.

When the visible units are real-valued, the model is called the Gaussian RBM, and

its joint probability distribution can be defined as follows:

$$P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \exp(-E(\mathbf{v}, \mathbf{h})), \tag{2.1}$$

$$E(\mathbf{v}, \mathbf{h}) = \frac{1}{2\sigma^2} \sum_i (v_i - c_i)^2 - \frac{1}{\sigma} \sum_{i,j} v_i W_{ij} h_j - \sum_j b_j h_j. \tag{2.2}$$

where $Z = \int_{\mathbf{v}} \sum_{\mathbf{h}} \exp\left(-E(\mathbf{v}, \mathbf{h})\right)$ is a normalization constant. The conditional distribution of this model can be written as follows:

$$\begin{aligned} P(h_j = 1|\mathbf{v}) &= \frac{\exp\left(\frac{1}{\sigma} \sum_i W_{ij} v_i + b_j\right)}{\sum_{h_j \in \{0,1\}} \exp\left(\frac{1}{\sigma} \sum_i v_i W_{ij} h_j + h_j b_j\right)} \\ &= \frac{\exp\left(\frac{1}{\sigma} \sum_i W_{ij} v_i + b_j\right)}{1 + \exp\left(\frac{1}{\sigma} \sum_i W_{ij} v_i + b_j\right)} = \text{sigm}(\frac{1}{\sigma} \sum_i W_{ij} v_i + b_j), \end{aligned} \tag{2.3}$$

$$\begin{aligned} P(v_i|\mathbf{h}) &\propto \exp\left(\frac{1}{2\sigma^2}(v_i - c_i)^2 - \frac{1}{\sigma} \sum_j v_i W_{ij} h_j\right) \\ &= \exp\left(\frac{1}{2\sigma^2}(v_i^2 + c_i^2 - 2 v_i c_i - 2\sigma \sum_j v_i W_{ij} h_j)\right) \\ &\propto \exp\left(\frac{1}{2\sigma^2}(v_i - \sigma \sum_j W_{ij} h_j - c_i)^2\right) = \mathcal{N}(v_i; \sigma \sum_j W_{ij} h_j + c_i, \sigma^2). \end{aligned} \tag{2.4}$$

where $\text{sigm}(s) = \frac{1}{1+\exp(-s)}$ is the sigmoid function, and $\mathcal{N}(\cdot; \cdot, \cdot)$ is a Gaussian distribution. Here, the variables in a layer (given the other layers) are conditionally independent, and thus we can perform block Gibbs sampling in parallel.

The RBM can be trained using sampling-based approximate maximum-likelihood, e.g., contrastive divergence (CD) approximation [44]. After training the RBM, the posterior (Equation (2.3)) of the hidden units (given input data) can be used as feature representations for classification tasks.

2.3.2 Convolutional RBMs

The Gaussian restricted Boltzmann machine is defined for input data in the form of vectors and does not model spatial context effectively. Thus, to make the RBMs

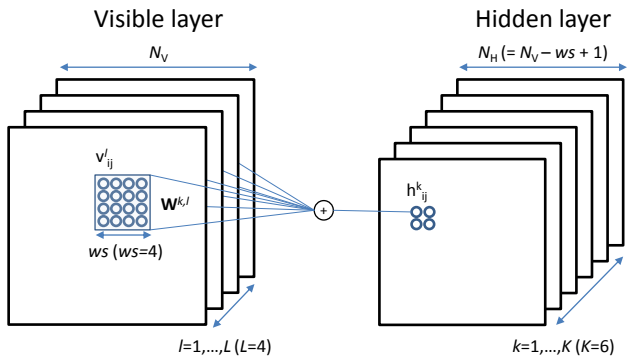


Figure 2.2: Illustration of CRBM. N_V and N_H refer to the size of visible and hidden layer, and ws to the size of convolution filter. The convolutional filter for the l-th channel (of size $ws \times ws$) corresponding to k-th hidden group is denoted as $\mathbf{W}^{k,l}$.

scalable to more realistic and larger images, [84] proposed the convolutional restricted Boltzmann machine (CRBM). The CRBM can be viewed as a convolutional learning algorithm that can detect salient patterns in unlabeled (image) data. A schematic description of the Gaussian CRBM is provided in Figure 2.2, whose energy function is defined as follows: $E(\mathbf{v}, \mathbf{h}) =$

$$\frac{1}{2\sigma^2}\sum_{l=1}^{L}\sum_{i,j}(v_{i,j}^l - c_l)^2 - \frac{1}{\sigma}\sum_{k=1}^{K}\sum_{l=1}^{L}\sum_{i,j,r,s} h_{i,j}^k W_{r,s}^{k,l} v_{i+r-1,j+s-1}^l - \sum_{k=1}^{K} b_k \sum_{i,j} h_{i,j}^k \tag{2.5}$$

$$= \frac{1}{2\sigma^2}\sum_{l=1}^{L}\sum_{i,j}(v_{i,j}^l - c_l)^2 - \sum_{l=1}^{L}\sum_{i,j} v_{i,j}^l \left(\sum_{k=1}^{K}\frac{1}{\sigma}(\mathbf{W}^{k,l} * \mathbf{h}^k)_{i,j}\right) - \sum_{k=1}^{K} b_k \sum_{i,j} h_{i,j}^k \tag{2.6}$$

$$= \frac{1}{2\sigma^2}\sum_{l=1}^{L}\sum_{i,j}(v_{i,j}^l - c_l)^2 - \sum_{k=1}^{K}\sum_{i,j} h_{i,j}^k \left(\sum_{l=1}^{L}\frac{1}{\sigma}(\widetilde{\mathbf{W}}^{k,l} * \mathbf{v}^l)_{i,j} + b_k\right), \tag{2.7}$$

where $\mathbf{v} \in \mathbb{R}^{N_V \times N_V \times L}$ denotes the visible nodes with L channels,[1,2] and $\mathbf{h} \in \mathbb{R}^{N_H \times N_H \times K}$ denotes the hidden nodes with K groups. The visible nodes and hidden nodes are related by the 4-d weight matrix $\mathbf{W} \in \mathbb{R}^{ws \times ws \times L \times K}$. More precisely, $\mathbf{W}^{k,l} \in \mathbb{R}^{ws \times ws}$ represents the connection between the units in k-th hidden group and l-th visible channel, and it is shared among the hidden units in the k-th group across all spatial locations. We define

[1]For the simplicity of presentation, we assume that input images are "square" shaped; however, the algorithm is applicable to images of arbitrary aspect ratios.

[2]For example, L will be 128 when we use dense SIFT as an input.

$\widetilde{\mathbf{W}}^{k,l}$ as the 2-d filter matrix $\mathbf{W}^{k,l}$ flipped vertically and horizontally, i.e., in Matlab notation, $\widetilde{\mathbf{W}}^{k,l} = fliplr(flipud(\mathbf{W}^{k,l}))$. The visible units in the l-th channel share the bias c_l, and the hidden units in the k-th hidden group share the bias b_k.

The conditional probability of the CRBM can be written as follows:

$$P(h^k_{i,j} = 1|\mathbf{v}) = \text{sigm}\left(\sum_l \frac{1}{\sigma}(\widetilde{\mathbf{W}}^{k,l} * \mathbf{v}^l)_{i,j} + b_k\right), \tag{2.8}$$

$$P(\mathbf{v}^l|\mathbf{h}) = \mathcal{N}\left(\mathbf{v}^l; \sigma \sum_k \mathbf{W}^{k,l} * \mathbf{h}^k + c_l, \sigma^2\mathbf{I}\right). \tag{2.9}$$

The convolutional RBM can be trained like the standard RBM using CD. Since the CRBM is highly overcomplete, sparsity regularization [82] is used to encourage the hidden units to have sparse activations.

2.4 Efficient Training of RBMs

Although the RBM has shown promise in computer vision problems, it is not yet as commonly used as other unsupervised learning algorithms primarily because of its difficulty in training. To address this issue, we provide a novel training algorithm by exploiting the relationship between clustering methods and RBMs.

2.4.1 Equivalence between mixture models and RBMs with a softmax constraint

In this section, we show that a Gaussian RBM with softmax hidden units can be converted into a GMM, and vice versa. This connection between mixture models and RBMs with a softmax constraint completes the chain of links between Kmeans, GMMs, Gaussian-softmax RBMs, sparse RBMs, and CRBMs. This chain of links will motivate an efficient training method for sparse RBMs and CRBMs.

Gaussian mixture model is a directed graphical model where the likelihood of visible units is expressed as a convex combination of Gaussian distributions. The likelihood of a GMM with $K+1$ Gaussians can be written as follows:

$$P(\mathbf{v}) = \sum_{k=0}^{K} \pi_k \mathcal{N}(\mathbf{v}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{2.10}$$

For the rest of this section, we denote the GMM with shared spherical covariance as GMM($\boldsymbol{\mu}_k, \sigma^2\mathbf{I}$), when $\boldsymbol{\Sigma}_k = \sigma^2\mathbf{I}$ for all $k \in \{0, 1, \ldots, K\}$. For the GMM with arbitrary positive definite covariance matrices, we will use the shorthand notation GMM($\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$).

Gaussian-softmax RBM is a Gaussian RBM with a constraint that at most one hidden unit can be activated for each input, i.e., $\sum_j h_j \leq 1$. The energy function of the Gaussian-softmax RBM can be written in a vectorized form as follows:

$$E(\mathbf{v}, \mathbf{h}) = \frac{1}{2\sigma^2}\|\mathbf{v} - \mathbf{c}\|^2 - \frac{1}{\sigma}\mathbf{v}^T\mathbf{W}\mathbf{h} - \mathbf{b}^T\mathbf{h} \text{ subject to } \sum_j h_j \leq 1 \tag{2.11}$$

The conditional probabilities can be computed as follows:

$$P(\mathbf{v}|\mathbf{h}) = \mathcal{N}(\mathbf{v}; \sigma\mathbf{W}\mathbf{h} + \mathbf{c}, \sigma^2\mathbf{I}) \tag{2.12}$$

$$P(h_j = 1|\mathbf{v}) = \frac{\exp(\frac{1}{\sigma}\mathbf{w}_j^T\mathbf{v} + b_j)}{1 + \sum_{j'} \exp(\frac{1}{\sigma}\mathbf{w}_{j'}^T\mathbf{v} + b_{j'})}, \tag{2.13}$$

where $\mathbf{w}_j$ is the j-th column of the $\mathbf{W}$ matrix, often denoted as a "basis" vector for the j-th hidden unit. In this model, there are $K+1$ possible configurations (i.e., all hidden units are 0, or only one hidden unit h_j is 1 for some j).

Equivalence between GMMs and Gaussian-softmax RBMs. The conditional probability of visible units given the hidden unit activations of Gaussian-softmax RBM follows a Gaussian distribution, as seen in Equation (2.12). From this perspective, the Gaussian-softmax RBM can be viewed as a mixture of Gaussians whose mean

components correspond to possible hidden unit configurations.[3] In this section, we show an explicit equivalence between these two models by formulating the conversion equations between GMM($\boldsymbol{\mu}_k, \sigma^2\mathbf{I}$) with $K+1$ Gaussian components and the Gaussian-softmax RBM with K hidden units.

Proposition II.1. *The mixture of $K+1$ Gaussians with shared spherical covariance of $\sigma^2\mathbf{I}$ is equivalent to the Gaussian-softmax RBM with K hidden units.*

Proof. We prove by constructing the following conversions.

(1) From Gaussian-softmax RBM to GMM($\boldsymbol{\mu}_k, \sigma^2\mathbf{I}$): We begin by the decomposition using a chain rule:

$$P(\mathbf{v}, \mathbf{h}) = P(\mathbf{v}|\mathbf{h})P(\mathbf{h}),$$

where

$$P(\mathbf{h}) = \frac{1}{Z}\int d\mathbf{v}\exp(-E(\mathbf{v}, \mathbf{h})).$$

Since there are only a finite number of hidden unit configurations, we can explicitly enumerate the prior probabilities:

$$P(h_j = 1) = \frac{\int d\mathbf{v}\exp(-E(\mathbf{v}, h_j = 1))}{\sum_{j'}\int d\mathbf{v}\exp(-E(\mathbf{v}, h_{j'} = 1))}$$

If we define $\tilde{\pi}_j = \int d\mathbf{v}\exp(-E(\mathbf{v}, h_j = 1))$, then we have $P(h_j = 1) = \frac{\tilde{\pi}_j}{\sum_{j'}\tilde{\pi}_{j'}} \triangleq \pi_j$. In fact, $\tilde{\pi}_j$ can be analytically calculated as follows:

$$\begin{aligned}
\tilde{\pi}_j &= \int d\mathbf{v}\exp(-E(\mathbf{v}, h_j = 1)) \\
&= \int d\mathbf{v}\exp(-\frac{1}{2\sigma^2}\|\mathbf{v} - \mathbf{c}\|^2 + \frac{1}{\sigma}\mathbf{v}^T\mathbf{w}_j + b_j) \\
&= (\sqrt{2\pi}\sigma)^L\exp(b_j + \frac{1}{2}\|\mathbf{w}_j\|^2 + \frac{1}{\sigma}\mathbf{c}^T\mathbf{w}_j)
\end{aligned}$$

[3] In fact, the Gaussian RBM (without any constraints) can be viewed as a mixture of Gaussians with an exponential number of components. However, it is nontrivial to use this notion itself to develop a useful algorithm.

Using this definition, we can show the following equality:

$$P(\mathbf{v}) = \sum_j \pi_j \mathcal{N}(\mathbf{v}; \sigma \mathbf{w}_j + \mathbf{c}, \sigma^2 \mathbf{I}).$$

(2) From GMM($\boldsymbol{\mu}_k, \sigma^2\mathbf{I}$) to Gaussian-softmax RBM: We will also show this by construction. Suppose we have the following GMM with $K+1$ components and the shared spherical covariance $\sigma^2\mathbf{I}$:

$$P(\mathbf{v}) = \sum_{j=0}^{K} \pi_j \mathcal{N}(\mathbf{v}; \boldsymbol{\mu}_j, \sigma^2 \mathbf{I}). \tag{2.14}$$

We can convert from this GMM($\boldsymbol{\mu}_k, \sigma^2\mathbf{I}$) to a Gaussian-softmax RBM using the following transformations:

$$\begin{aligned} \mathbf{c} &= \boldsymbol{\mu}_0 \\ \mathbf{w}_j &= \frac{1}{\sigma}(\boldsymbol{\mu}_j - \mathbf{c}), j = 1, ..., K \\ b_j &= \log \frac{\pi_j}{\pi_0} - \frac{1}{2}\|\mathbf{w}_j\|^2 - \frac{1}{\sigma}\mathbf{w}_j^T \mathbf{c}. \end{aligned} \tag{2.15}$$

It is easy to see that the conditional distribution $P(\mathbf{v}|h_j = 1)$ can be formulated as a Gaussian distribution with mean $\boldsymbol{\mu}_j = \sigma \mathbf{w}_j + \mathbf{c}$, which is identical to that of the Gaussian-softmax RBM. Further, we can recover the posterior probability of hidden units given the visible units as follows:

$$\begin{aligned} P(h_j = 1|\mathbf{v}) &= \frac{\pi_j \exp(-\frac{1}{2\sigma^2}\|\mathbf{v} - \sigma\mathbf{w}_j - \mathbf{c}\|^2)}{\sum_{j'=0}^{K} \pi_{j'} \exp(-\frac{1}{2\sigma^2}\|\mathbf{v} - \sigma\mathbf{w}_{j'} - \mathbf{c}\|^2)} \\ &= \frac{\exp(\frac{1}{\sigma}\mathbf{w}_j^T \mathbf{v} + b_j)}{1 + \sum_{j'=1}^{K} \exp(\frac{1}{\sigma}\mathbf{w}_{j'}^T \mathbf{v} + b_{j'})} \end{aligned}$$

Therefore, a GMM can be converted to Gaussian RBM with a softmax constraint. □

Similarly, the GMM with shared diagonal covariance is equivalent to the Gaussian-softmax RBM with a slightly more general energy function, where each visible unit v_i

has its own noise parameter σ_i, as stated below.

Corollary II.2. *The mixture of $K+1$ Gaussians with a shared diagonal covariance matrix (with diagonal entries $\sigma_i^2, i = 1, ..., L$) is equivalent to the Gaussian-softmax RBM with the following energy function: $E(\mathbf{v}, \mathbf{h}) = \sum_i \frac{1}{2\sigma_i^2}(v_i - c_i)^2 - \sum_{i,j} \frac{1}{\sigma_i} v_i W_{ij} h_j - \sum_j b_j h_j$.*

Further, the equivalence between mixture models and RBMs can be shown for other settings. For example, the following corollaries can be derived from Proposition II.1.

Corollary II.3. *The binary RBM (i.e., when the visible units are binary) with a softmax constraint on hidden units and the mixture of Bernoulli models are equivalent.*

Corollary II.4. *GMM($\mathbf{0}, \mathbf{\Sigma}_k$) with arbitrary covariance matrices and the factored 3-way RBM [104] with a softmax constraint on hidden units are equivalent.*

We provide proofs for Corollary II.3 and II.4 in Appendix A.

Implication. Proposition II.1 has important ramifications. First, it is well known that K-means can be viewed as an approximation of a GMM with spherical covariance by letting $\sigma \to 0$ [12]. Compared to GMMs, the training of K-means is highly efficient; therefore, it is plausible to train K-means to provide an initialization of a GMM.[4] Then, the GMM is trained with expectation-maximization (EM) algorithm, and we convert it to an RBM with softmax units. As will be discussed, this provides an efficient initialization for training sparse RBMs and CRBMs.

2.4.2 Activation constrained RBMs, sparse RBMs, and convolutional RBMs

We extend the Gaussian-softmax RBM to more general Gaussian RBMs that allow at most $\alpha \geq 1$ hidden units to be active for a given input example. We call this model

[4] K-means learns cluster centroids and provides hard-assignment of training examples to the cluster centroids (i.e., each example is assigned to one centroid). This hard-assignment can be used to initialize GMM's parameters, such as π_k and σ, by running one M-step in the EM algorithm.

the *activation constrained* RBM, and its energy function is written as follows:

$$E(\mathbf{v}, \mathbf{h}) = \frac{1}{2\sigma^2}\|\mathbf{v} - \mathbf{c}\|^2 - \frac{1}{\sigma}\mathbf{v}^T\mathbf{W}\mathbf{h} - \mathbf{b}^T\mathbf{h} \text{ subject to } \sum_j h_j \leq \alpha \tag{2.16}$$

Note that the number of possible hidden configurations grows polynomial with α.[5] Therefore, such relaxation provides more expressive power than GMMs. However, there is a trade-off between the expressive power (or capacity), and the tractability of exact inference and maximum-likelihood training. For example, an exact EM algorithm will require polynomial time complexity of $O(K^\alpha)$, which may be computationally expensive.

To address such difficulties, we approximate the activation constrained RBM to the sparse RBM [82]. Specifically, the sparse RBM is a variant of the RBM trained with a regularizer that encourages the average activation to be low (i.e., with target sparsity p_0) in the hidden representations. By setting $p_0 = \alpha/K$, the sparse RBM can be regarded as an approximation to the activation constrained RBM with a constraint $\sum_j h_j \leq \alpha$. The inference and training of sparse RBMs is much more efficient as α increases.

We further observe that the CRBM is a generalization of the sparse RBM. Specifically, the two algorithms are equivalent when (1) the filter size of the CRBM is 1 (i.e., the convolution does not smooth the image); or (2) the filter size is the same as the image size (i.e., this is essentially equivalent to vectorizing the whole image, which is usually not interesting). For example, note that non-convolutional feature learning algorithms (e.g., sparse RBM) on SIFT descriptors would have a weight matrix of size $128 \times K$, which is equivalent to that of convolutional algorithms ($1 \times 1 \times 128 \times K$, i.e., no interactions between adjacent hidden units). In general, CRBMs can model a larger spatial context using $ws \times ws \times L$ as weights for each hidden unit.

Predictions. Based on the connections described, we make the following predictions:

[5] If $\alpha \to \infty$ or there is no such constraint, the model is equivalent to the Gaussian RBM that has an exponential number of hidden configurations.

Algorithm 1 Efficient training algorithm for sparse or convolutional RBMs

1: Train $K + 1$ centroids $\boldsymbol{\mu}_k$ via K-means.
2: Initialize GMM($\boldsymbol{\mu}_k, \sigma^2\mathbf{I}$) parameters from K-means.
3: Train GMM($\boldsymbol{\mu}_k, \sigma^2\mathbf{I}$) via EM.
4: Initialize the RBM parameters (see Proposition II.1).
5: Train sparse or convolutional RBMs (e.g., via CD).

- K-means, GMM, and Gaussian-softmax RBM (with the same K) should have similar expressive power and show similar classification performance.
- When $\alpha > 1$, sparse RBMs can give better classification performance than K-means or GMMs due to their increased expressive power.
- When convolutional filter size is larger than 1, CRBMs can give better classification performance than non-convolutional RBMs since they can learn spatial context more efficiently.

These predictions will be verified with our efficient training method in the following section.

2.4.3 Algorithm and implementation details

The overall procedure for training sparse or convolutional RBMs is shown in Algorithm 1. In addition, we used the following methods to select hyperparameters.

Setting the σ automatically. As in Equation (2.4), σ roughly controls the noise in the visible units. Typically, σ is fixed during training and treated as a hyperparameter that needs to be cross-validated. As an alternative, we used the following heuristic to automatically tune the σ value. Suppose that we are given a fixed set of hidden unit values $\hat{h}$, then we have the following conditional probability distribution for the Gaussian RBM:

$$P(\mathbf{v}|\mathbf{h}) = \mathcal{N}(\mathbf{v}; \sigma\mathbf{W}\hat{\mathbf{h}} + \mathbf{c}, \sigma^2 I) \tag{2.17}$$

If we apply the maximum likelihood estimation of σ given $\hat{\mathbf{h}}$ fixed, then σ should be the sample standard deviation of $\mathbf{v} - (\sigma \mathbf{W}\hat{\mathbf{h}} + \mathbf{c})$. Here, we use $\hat{\mathbf{h}}$ as the expectation of $\mathbf{h}$ given input $\mathbf{v}$. Thus, we update σ so that it becomes close to the reconstruction error of the training data.[6] The same method also applies to convolutional training.

Setting the L_2 regularization. When training RBMs, a hyperparameter for L_2 regularization (that penalizes high L_2 norm of W) typically has to be determined via cross validation. However, due to the connections between mixture models and RBMs discussed in section 2.4.1, setting the L_2 regularization is straightforward. Specifically, the clustering-based initialization justifies using the L_2 regularization hyperparameter obtained from clustering models, which are often very small. In our experiments, we used 0.0001 without tuning.

In the following section, we show the efficacy of our training algorithm and provide experimental evidence for the above predictions. From the experiments, we find that a combination of moderately sparse ($1 < \alpha \ll K$) representations with a moderate amount of spatial convolution ($1 < ws \ll N_V$) performs the best for object recognition. Further, we show that our feature representation achieves state-of-the-art performance.

2.5 Experiments and Discussions

In this section, we report classification results based on two datasets: Caltech 101 [32] and Caltech 256 [37]. In the experiments, we used SIFT as low-level descriptors, which were extracted densely from every 6 pixels with a patch size of 24. We resized the images to no larger than 300×300 pixels with a preserved aspect ratio for computational efficiency. After training the codebook, feature vectors were pooled from the 4×4, 2×2, and 1×1 subregions using max-pooling and then concatenated to single

[6] We define the reconstruction error as $\sqrt{\frac{1}{LM}\sum_{i=1}^{M} \|\mathbf{v}^{(i)} - (\sigma \mathbf{W}\hat{\mathbf{h}}^{(i)} + \mathbf{c})\|^2}$ for training examples $\{\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(M)}\}$.

training images	5	10	15	20	25	30
Lazebnik et al. [74]	-	-	56.4	-	-	64.6
Griffin et al. [37]	44.2	54.5	59.0	63.3	65.8	67.6
Yang et al. [143]	-	-	67.0	-	-	73.2
Wang et al. [135]	51.2	59.8	65.4	67.7	70.2	73.4
Boureau et al. [15]	-	-	-	-	-	75.7
K-means (K=4096)	47.6	58.1	63.4	66.6	69.1	70.9
GMM (K=4096)	50.2	60.3	65.3	68.6	70.8	72.2
sparse RBM (K=4096)	54.2	64.0	68.6	71.2	73.1	74.9
CRBM (K=2048)	56.5	66.4	70.7	73.5	75.4	77.4
CRBM (K=4096)	**56.7**	**66.7**	**71.3**	**74.2**	**76.2**	**77.8**

Table 2.1: Average test classification accuracy for Caltech 101.

feature vectors. We used linear SVM [30] for classifier on randomly selected training images (with a fixed number of images per class) and then evaluated the classification accuracy on the rest of the images. We performed 5-fold cross-validation to determine hyperparameters on each randomly selected training set and reported the test accuracy averaged over 10 trials.

2.5.1 Caltech 101

The Caltech 101 dataset [32] is composed of 9,144 images split into 101 object categories, such as vehicles, artifacts, and animals, as well as one background category with significant variances in shape. The number of images in each class varies from 31 to 800. For fair comparisons, we performed experiments as in other studies [32, 143, 135]. Specifically, for each trial, we randomly selected $5, 10, \ldots, 30$ images from each class, including the background class, and trained a linear classifier. The remaining images from each class were tested, and the average accuracy over the classes was reported.

We summarize the results from our proposed method and other existing methods in Table 2.1. Our algorithm clearly outperformed other state-of-the-art algorithms using a single type of feature. Specifically, our method breaks the record on the Caltech 101 dataset by 4.3% for 15 training images and 2.1% for 30 training images.

training images	15	30	45	60
Griffin et al. [37]	28.30	34.10	-	-
van Gemert et al. [129]	-	27.17	-	-
Yang et al. [143]	27.73	34.02	37.46	40.14
Wang et al. [135]	34.36	41.19	45.31	47.68
CRBM (K=4096)	**35.09**	**42.05**	**45.69**	**47.94**

Table 2.2: Average test classification accuracy for Caltech 256.

2.5.2 Caltech 256

We also tested our algorithm on a more challenging dataset. Caltech 256 dataset [37] is composed of 30,607 images split into 256 object categories with more variability and finer classifications, as well as one "clutter" class of random pictures. Each class contains at least 80 images; the objects in each image are more variant in size, location, pose, etc., than those of Caltech 101 dataset. We followed the standard experimental settings from the benchmarks [37, 143], and the overall classification accuracy was averaged over 10 random trials. The summary of the results is reported in Table 2.2. Our algorithm performed slightly better than the LLC [135] algorithm, with considerably large margins to many other methods on Caltech 256 dataset.

2.5.3 Analysis of hyperparameters

To provide a better understanding of our proposed algorithm, we give a detailed analysis of the hyperparameters: sparsity and convolutional filter size. We performed the control experiments on Caltech 101 dataset while fixing the number of bases to 1024. In most cases, we observed improvement as the number of hidden bases increased, which is consistent with what others have reported [21]. All results reported in this section are validation accuracy (5-fold cross validation on the training set).

Sparsity (α/K). The performance of sparse models, such as sparse coding and sparse RBMs, can vary significantly as a function of sparsity level. As we discussed in Sec-

tion 2.4, the sparse RBM can be more expressive than K-means or GMMs. While K-means and GMM have sparsity of $1/K$ on average (i.e., allow only one cluster to be active for a given input), the sparse RBM can control sparsity by setting the target sparsity value $p_0 = \alpha/K$.

In this experiment, we compared two settings for sparse RBM training—one by initializing from GMM as described in Section 2.4, and the other by initializing randomly (baseline). Figure 2.3 (left) shows the average validation accuracy as a function of sparsity (α/K). Compared to the K-means and GMM, the sparse RBM with random initialization performed very poorly in the low α regime (i.e., when its corresponding number of activation is roughly 1). However, by using an efficient training method described in Section 2.4, the sparse RBM performs as well as K-means and GMM when the target sparsity is close to $1/K$, and significantly outperforms K-means and GMM when the representation is less sparse. Overall, the effect of accurate initialization is striking, especially in the high sparsity regime, and the best validation accuracy (maximum over α) was improved by 2% for both 15 and 30 training images.

Convolution filter size. Convolutional learning is powerful since it captures the spatial correlation between neighboring descriptors (e.g., pixels or dense SIFT) more efficiently than non-convolutional learning. The size of the filter, however, should be selected carefully. For instance, it is difficult to capture enough spatial information with small size filters; on the other hand, overly large filter size can result in severe over-smoothing of small details in the image. Therefore, we investigate how the filter size affects the performance.

In this experiment, we fixed the number of bases ($K = 1024$) and sparsity ($\alpha = 4$) while varying the convolutional filter size from 1 to 5. As shown in Figure 2.3 (right), the filter size of 3 resulted in highest validation accuracy. We also observed improvements (up to 2%) using the clustering-based initialization method in the convolutional setting.

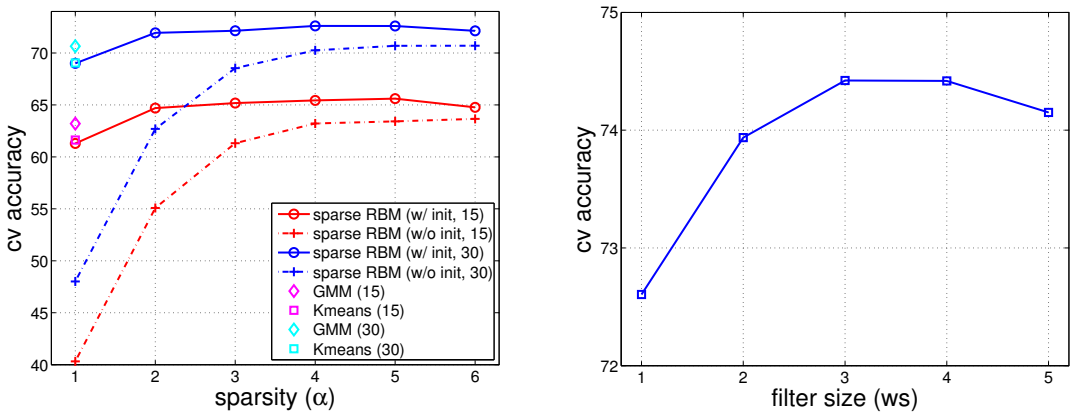


Figure 2.3: (Left) Average cross-validation accuracy on Caltech 101 dataset with 1024 bases using K-means, GMM, and sparse RBM with different sparsity values. The "sparse RBM (w/ init)" denotes the sparse RBM initialized from GMM as described in Section 2.4; the "sparse RBM (w/o init)" denotes the sparse RBM initialized randomly (baseline). Blue and cyan represent settings with 30 training images per class. Red and magenta represent settings with 15 training images per class. (Right) Average cross-validation accuracy on the Caltech 101 dataset with 1024 bases and different convolution filter sizes (ws).

Overall, our experimental analysis confirms that a moderately sparse ($1 < \alpha \ll K$) representation that is convolutionally trained with sufficient spatial context ($1 < ws \ll N_V$) outperforms both clustering methods (e.g., K-means and GMMs) and non-convolutional counterparts (e.g., non-convolutional sparse RBMs).

2.6 Conclusion

In this chapter, we proposed a mid-level feature extraction method using convolutional RBMs. Our key idea is to investigate an efficient training method for sparse RBMs and convolutional RBMs through the connections between mixture models and RBMs. In our experiments, we show efficacy of our training algorithm, as well as the benefit of learning sparse, distributed, convolutional feature representations. Overall, our method achieves state-of-the-art performance in object recognition benchmarks.

CHAPTER III

Learning Invariant Representations with Local Transformations

3.1 Introduction

In recent years, unsupervised feature learning algorithms have emerged as promising tools for learning representations from data [46, 9, 91]. In particular, it is an important problem to learn invariant representations that are robust to variability in high-dimensional data (e.g., images, speech, etc.) since they will enable machine learning systems to achieve good generalization performance while using a small number of labeled training examples. In this context, several feature learning algorithms have been proposed to learn invariant representations for specific transformations by using customized approaches. For example, convolutional feature learning methods [84, 60, 148] can achieve shift-invariance by exploiting convolution operators. As another example, the denoising autoencoder [132] can learn features that are robust to the input noise by trying to reconstruct the original data from the hidden representation of the perturbed data. However, learning invariant representations with respect to general types of transformations is still a challenging problem.

In this chapter, we present a novel framework of transformation-invariant feature learning. We focus on local transformations (e.g., small amounts of translation, ro-

tation, and scaling in images), which can be approximated as linear transformations, and incorporate linear transformation operators into the feature learning algorithms. For example, we present the transformation-invariant restricted Boltzmann machine, which is a generative model that represents input data as a combination of transformed weights. In this case, a transformation-invariant feature representation is obtained via probabilistic max pooling of the hidden units over the set of transformations. In addition, we show extensions of our transformation-invariant feature learning framework to other unsupervised feature learning algorithms, such as autoencoders or sparse coding.

In our experiments, we evaluate our method on the variations of the MNIST dataset and show that our algorithm can significantly outperform the baseline restricted Boltzmann machine when underlying transformations in the data are well-matched to those considered in the model. Furthermore, our method can learn features that are much more robust to the wide range of local transformations, which results in highly competitive performance in visual recognition tasks on CIFAR-10 [66] and STL-10 [21] datasets. In addition, our method also achieves state-of-the-art performance on phone classification tasks on the TIMIT dataset, which demonstrates wide applicability of our proposed algorithms to other domains.

3.2 Related Work

Researchers have made significant efforts to develop invariant feature representations. For example, the rotation- or scale-invariant descriptors, such as SIFT [88], have shown a great success in many computer vision applications. However, these image descriptors usually demand a domain-specific knowledge with a significant amount of hand-crafting.

As an alternative approach, several unsupervised learning algorithms have been proposed to learn robust feature representations automatically from the sensory data. As an example, the denoising autoencoder [132] can learn robust features by trying to

reconstruct the original data from the hidden representations of randomly perturbed data generated from the distortion processes, such as adding noise or multiplying zeros for randomly selected coordinates.

Among types of transformations relating to temporally or spatially correlated data, translation has been extensively studied in the context of unsupervised learning. Specifically, convolutional training [77, 60, 148, 84, 119] is one of the most popular methods that encourages shift-invariance during the feature learning. For example, the convolutional deep belief network (CDBN) [84], which is composed of multiple layers of convolutional restricted Boltzmann machines and probabilistic max pooling, can learn a representation invariant to local translation.

Besides translation, however, learning invariant features for other types of image transformations have not been extensively studied. In contemporary work of ours, Kivinen and Williams [64] proposed the transformation equivariant Boltzmann machine, which shares a similar mathematical formulation to our models in that both try to infer the best matching filters by transforming them using linear transformation matrices. However, while their model was motivated from the "global equivariance", the main purpose of our work is to learn locally-invariant features that can be useful in classification tasks. Thus, rather than considering an algebraic group of transformation matrices (e.g., "full rotations"), we focus on the variety of local transformations that include rotation, translation as well as scale variations. Furthermore, we effectively address the boundary effects that can be highly problematic in translation and scaling by forming a non-square T matrix, rather than zero-padding.[1] In addition, we present a general framework of transformation-invariant feature learning and show extensions based on the autoencoder and sparse coding. Overall, our argument is strongly supported by the state-of-the-art performance in image and audio classification tasks.

[1]We observed that learning with zero-padded squared transformation matrices showed significant boundary effect in its visualization of filters, and this often resulted in significantly worse classification performance.

Apart from learning transformation-invariant features, imposing structures such as topographic maps [56, 59] on the activation is another way of learning invariant representations. Compared to those algorithms, our method can be more compact and expressive since we factor out the patterns and their transformations and therefore we only need to learn a smaller number of features.

3.3 Learning Transformation-Invariant Features

3.3.1 Transformation-invariant RBM

In this section, we formulate a novel feature learning framework that can learn invariance to a set of linear transformations based on the RBM. We begin the section with describing the transformation operator.

The transformation operator is defined as a mapping $T : \mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$ that maps d_1-dimensional input vectors into d_2-dimensional output vectors. In our case, we assume a linear transformation matrix $T \in \mathbb{R}^{d_2 \times d_1}$, i.e., each coordinate of the output vector is represented as a linear combination of the input coordinates.

With this notation, we formulate the *transformation-invariant restricted Boltzmann machine* (TI-RBM) that can learn invariance to a set of transformations. Specifically, for a given set of transformation matrices T_s $(s = 1, \cdots, S)$, the energy function of TI-RBM is defined as follows:

$$E(\mathbf{v}, \mathbf{H}) = -\sum_{j=1}^{K}\sum_{s=1}^{S}(T_s\mathbf{v})^T\mathbf{w}_j h_{j,s} - \sum_{j=1}^{K}\sum_{s=1}^{S} b_{j,s}h_{j,s} - \mathbf{c}^T\mathbf{v} \tag{3.1}$$

$$\text{s.t. } \sum_{s=1}^{S} h_{j,s} \leq 1, h_{j,s} \in \{0,1\}, j = 1, \cdots, K, \tag{3.2}$$

where $\mathbf{v} \in \mathbb{R}^{d_1}$ are the visible units, and $\mathbf{w}_j \in \mathbb{R}^{d_2}$ are filters (or bases). The hidden units are represented as a matrix $\mathbf{H} \in \{0,1\}^{K \times S}$ with $h_{j,s}$ as its (j,s)-th entry. In addition, we denote $z_j = \sum_{s=1}^{S} h_{j,s}$, $z_j \in \{0,1\}$ as a pooled hidden unit over the transformations.

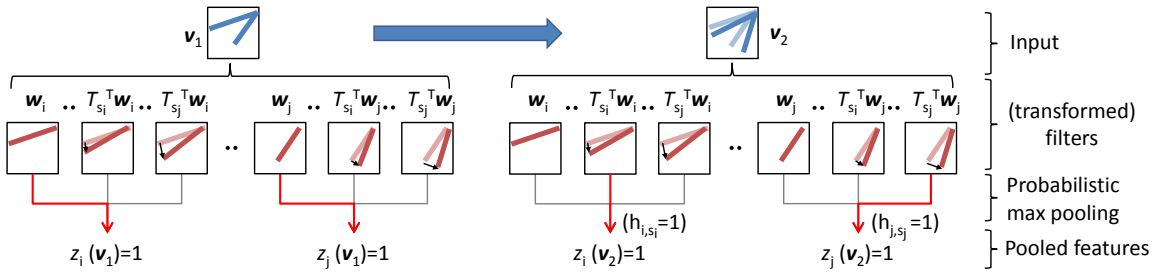


Figure 3.1: Feature encoding of TI-RBM. Shaded pattern inside the $\mathbf{v}_2$ reflects the $\mathbf{v}_1$, while the shaded patterns in transformed filters show the corresponding original filters $\mathbf{w}_i$ or $\mathbf{w}_j$. The filters selected via probabilistic max pooling across the set of transformations are depicted in red arrows (e.g., in the rightmost example, the hidden unit h_{j,s_j} corresponding to the transformation T_{s_j} and the filter $\mathbf{w}_j$ contributes to activate $z_j(\mathbf{v}_2)$.)

In Equation (3.2), we impose a softmax constraint on hidden units so that at most one unit is activated at each row of $\mathbf{H}$. This probabilistic max pooling[2] allows us to obtain a feature representation invariant to linear transformations. More precisely, suppose that the input $\mathbf{v}_1$ activates the filter vector $\mathbf{w}_j$. Given another input $\mathbf{v}_2$ that is a transformed version of $\mathbf{v}_1$, the TI-RBM will find a transformation matrix T_{s_j} so that the $\mathbf{v}_2$ activates the transformed filter vector $T_{s_j}^T\mathbf{w}_j$, i.e., $\mathbf{w}_j^T\mathbf{v}_1 \approx \mathbf{w}_j^T T_{s_j}\mathbf{v}_2$. Therefore, $\mathbf{v}_1$ and $\mathbf{v}_2$ both activate z_j after probabilistic max pooling. Figure 3.1 illustrates this idea.

Compared to the regular RBM, the TI-RBM can learn more diverse patterns, while keeping the number of model parameters small. Specifically, multiplying transformation matrix $T_s^T\mathbf{w}_j$ can be viewed as increasing the number of filters by the factor of S, but without significantly increasing the number of parameters due to parameter sharing. In addition, by pooling over local transformations, the filters can learn invariant representations (i.e., z_j's) to these transformations.

[2]A similar technique is used in convolutional deep belief networks [84], in which spatial probabilistic max pooling is applied over a small spatial region.

The conditional probabilities are computed as follows:

$$p(h_{j,s} = 1|\mathbf{v}) = \frac{\exp\left(\mathbf{w}_j^T T_s \mathbf{v} + b_{j,s}\right)}{1 + \sum_{s'} \exp\left(\mathbf{w}_j^T T_{s'} \mathbf{v} + b_{j,s'}\right)} \tag{3.3}$$

$$p(v_i = 1|\mathbf{h}) = \text{sigmoid}(\sum_{j,s} (T_s^T \mathbf{w}_j)_i h_{j,s} + c_i) \tag{3.4}$$

Similar to RBM training, we use stochastic gradient descent to train TI-RBM. The gradient of the log-likelihood is approximated via contrastive divergence by taking the gradient of the energy function (Equation (3.1)) with respect to the model parameters.

3.3.2 Sparse TI-RBM

The sparseness of the feature representation is often a desirable property. By following Lee et al. [82] approach, we can extend our model to sparse TI-RBM by adding the following regularizer for a given set of data $\{\mathbf{v}^{(1)}, \cdots, \mathbf{v}^{(N)}\}$ to the negative log-likelihood:

$$\mathcal{L}_{\text{sp}} = \sum_{j=1}^{K} \mathcal{D}\left(p, \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}[z_j^{(n)}|\mathbf{v}^{(n)}]\right) \tag{3.5}$$

where $\mathcal{D}$ is a distance function, p is the target sparsity. The expectation of pooled activation is written as

$$\mathbb{E}[z_j|\mathbf{v}] = \frac{\sum_s \exp\left(\mathbf{w}_j^T T_s \mathbf{v} + b_{j,s}\right)}{1 + \sum_s \exp\left(\mathbf{w}_j^T T_s \mathbf{v} + b_{j,s}\right)}. \tag{3.6}$$

Note that we regularize over the pooled hidden units z_j rather than individual hidden units $h_{j,s}$. In experiments, we used L2 distance for $\mathcal{D}(\cdot, \cdot)$, but one can also use KL divergence for the sparsity penalty.

3.3.3 Generating transformation matrices

In this section, we discuss how to design the transformation matrix T. For the ease of presentation, we assume 1-d transformations, but it can be extended to 2-d cases (e.g., image transformations) straightforwardly. Further, we assume the case of $d_1 = d_2 = d$.

As mentioned previously, $T \in \mathbb{R}^{d \times d}$ is a linear projection matrix from $\mathbf{x} \in \mathbb{R}^d$ to $\mathbf{y} \in \mathbb{R}^d$; i.e., each coordinate of $\mathbf{y}$ is constructed via linear combination of the coordinates in $\mathbf{x}$ with weight matrix T as follows:

$$y_i = \sum_{j=1}^{d} T_{ij} x_j, \forall i = 1, \cdots, d \tag{3.7}$$

For example, shifting by s can be defined as

$$T_{ij}(s) = \begin{cases} 1 & \text{if } i = j + s, \\ 0 & \text{otherwise} \end{cases}$$

For 2-d image transformations such as rotation and scaling, the contribution of input coordinates to each output coordinate is computed with bilinear interpolation. Since T's are pre-computed once and usually sparse, Equation (3.7) can be computed efficiently.

3.3.4 Extensions to other methods

We emphasize that our transformation-invariant feature learning framework is not limited to the energy-based probabilistic models, but can be extended to other unsupervised learning methods as well.

First, it can be readily adapted to autoencoders by defining the following softmax

encoding and sigmoid decoding functions:

$$f_{j,s}(\mathbf{v}) = \frac{\exp\left(\mathbf{w}_j^T T_s \mathbf{v} + b_{j,s}\right)}{1 + \sum_{s'} \exp\left(\mathbf{w}_j^T T_{s'} \mathbf{v} + b_{j,s'}\right)} \tag{3.8}$$

$$\hat{v}_i = \text{sigmoid}(\sum_{j,s} (T_s^T \mathbf{w}_j)_i f_{j,s} + c_i) \tag{3.9}$$

Following the idea of TI-RBM, we can also formulate the transformation-invariant sparse coding as follows:

$$\min_{\mathbf{W},\mathbf{H}} \sum_n \| \sum_{j=1}^{K} \sum_{s=1}^{S} T_s^T \mathbf{w}_j h_{j,s}^{(n)} - \mathbf{v}^{(n)} \|^2, \tag{3.10}$$

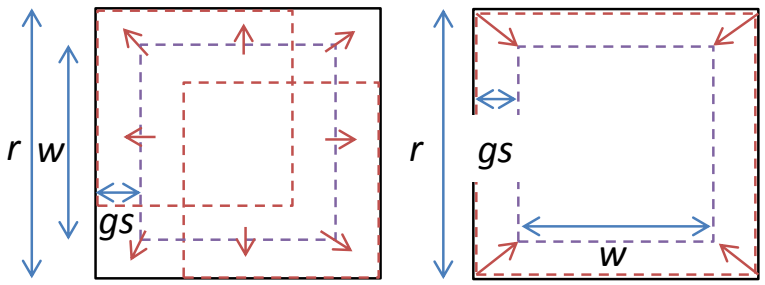
$$\text{s.t. } \|\mathbf{H}\|_0 \leq \gamma, \|\mathbf{H}(j,:)\|_0 \leq 1, \|\mathbf{w}_j\|_2 \leq 1, \tag{3.11}$$

where γ is a constant. The second constraint in (3.11) can be understood as an analogy to the softmax constraint in Equation (3.2) of TI-RBMs.

Similar to standard sparse coding, we can optimize the parameters by alternately optimizing $\mathbf{W}$ and $\mathbf{H}$ while fixing the other. Specifically, $\mathbf{H}$ can be (approximately) solved using Orthogonal Matching Pursuit, and therefore we refer this algorithm a transformation-invariant Orthogonal Matching Pursuit (TI-OMP).

3.4 Experiments

We begin by describing the notation. For images, we assume a receptive field size of $r \times r$ pixels (for input image patches) and a filter size of $w \times w$ pixels. We define gs to denote the number of pixels corresponding to the transformation (e.g., translation or scaling). For example, we translate the $w \times w$ filter across the $r \times r$ receptive field with a stride of gs pixels (Figure 3.2(a)), or scale down from $(r - l \cdot gs) \times (r - l \cdot gs)$ to $w \times w$ (where $0 \leq l \leq \lfloor (r-w)/gs \rfloor$) by sharing the same center for the filter and the receptive field (Figure 3.2(b)).



(a) translation (b) scaling

Figure 3.2: Translation and scale transformations on images.

For classification tasks, we used the posterior probability of the pooled hidden unit (Equation (3.6)) as a feature. Note that the dimension of the extracted feature vector for each image patch is K, not $K \times S$. Thus, we argue that the performance gain of the TI-RBM over the regular RBM comes from the better representation (i.e., transformation-invariant features), rather than from the classifier's use of higher-dimensional features.

3.4.1 Handwritten digit recognition with prior transformation information

First, we verified the performance of our algorithm on the variations of a handwritten digit dataset, assuming that the transformation information is given. From the MNIST variation datasets [73], we tested on "mnist-rot" (rotated digits, referred to as *rot*) and "mnist-rot-back-image" (rotated digits with background images, referred to as *rot-bgimg*). To further evaluate with different types of transformations, we created four additional datasets that contain scale and translation variations with and without random background (referred to as *scale*, *scale-bgrand*, *trans*, and *trans-bgrand*, respectively).[3] Some examples are shown in Figure 3.3.

For these datasets, we trained sparse TI-RBMs on the image patches of size 28×28

[3]We followed the generation process described in `http://www.iro.umontreal.ca/~lisa/twiki/bin/view.cgi/Public/MnistVariations` to create the customized scaled and translated digits. For example, we randomly selected the scale-level uniformly from 0.3 to 1 or the number of pixel shifts in horizontal and vertical directions without making any truncation of the foreground pixels. For the datasets with random background, we randomly added the uniform noise in $[0, 1]$ to the background pixels.

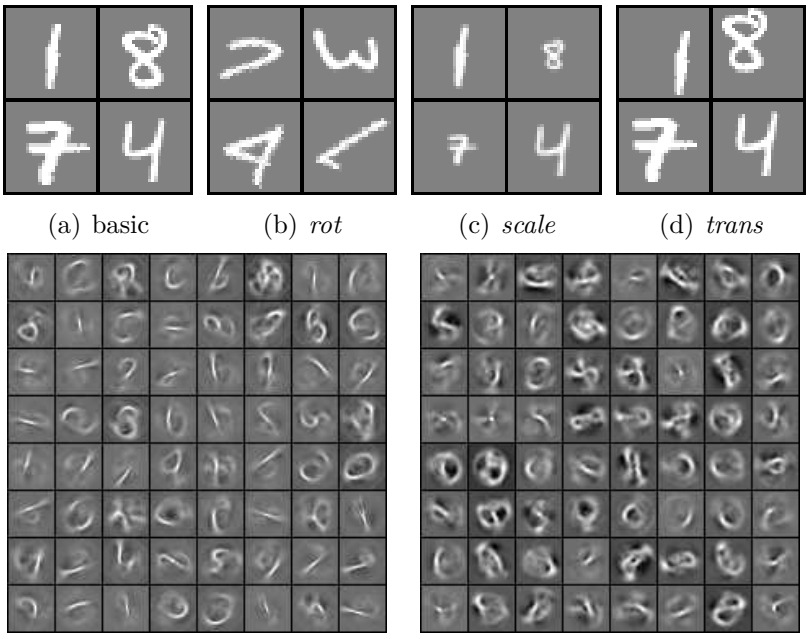
Dataset	RBM	TI-RBM	Transformation
rot	15.6%	**4.2**%	rotation
rot-bgimg	54.0%	**35.5**%	rotation
scale	6.1%	**5.5**%	scaling
scale-bgrand	32.1%	**23.7**%	scaling
trans	15.3%	**9.1**%	translation
trans-bgrand	57.3%	**43.7**%	translation

Table 3.1: Test classification error on MNIST transformation datasets. The best-performing methods for each dataset are shown in bold.

pixels with data-specific transformations. For example, we considered 16 equally-spaced rotations (i.e., the step size of $\frac{\pi}{8}$) for the *rot* and *rot-bgimg* datasets. Similarly, for the *scale* and *scale-bgrand* datasets, we generated scale-transformation matrices with $w = 20$ and $gs = 2$, which can map from $(28 - 2l) \times (28 - 2l)$ pixels to 20×20 pixels with $l \in \{0, ..., 4\}$. For the *trans* and *trans-bgrand* datasets, we set $w = 24$ and $gs = 2$ to have total nine translation matrices that cover the 28×28 regions using 24×24 pixels with a horizontal and vertical stride of 2 pixels. For classification, we trained $1,000$ filters for both sparse RBMs and sparse TI-RBMs and used a softmax classifier. We used 10,000 examples for the training set, 2,000 examples for the validation set, and 50,000 examples for the test set.

As reported in Table 3.1, our method (sparse TI-RBMs) consistently outperformed the baseline method (sparse RBMs) for all datasets. These results suggest that the TI-RBMs can learn better representations for the foreground objects by transforming the filters. It is worth noting that our error rates for the mnist-rot and mnist-rot-back-image datasets are also significantly lower than the best published results obtained with stacked denoising autoencoders [133] (9.53% and 43.75%, respectively).

For qualitative evaluation, we visualize the learned filters on the mnist-rot dataset trained with the sparse TI-RBM (Figure 3.3(e)) and the sparse RBM (Figure 3.3(f)), respectively. The filters learned from sparse TI-RBMs show much clearer pen-strokes than those learned from sparse RBMs, which partially explains the impressive classifi-



(a) basic (b) *rot* (c) *scale* (d) *trans*

(e) TI-RBM on mnist-rot (f) RBM on mnist-rot

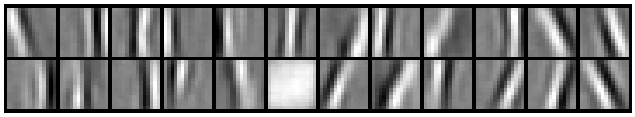
Figure 3.3: (top) Samples from the handwritten digit datasets with (a) no transformations, (b) rotation, (c) scaling, and (d) translation. (bottom) Learned filters from mnist-rot dataset with (e) the sparse TI-RBM and (f) the sparse RBM, respectively.

cation performance.

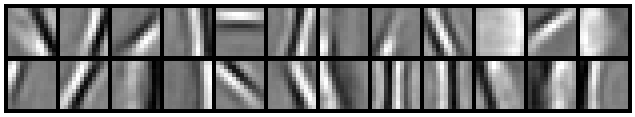
3.4.2 Learning invariant features from natural images

For handwritten digit recognition in the previous section, we assumed prior information on *global* transformations on the image (e.g., translation, rotation, and scale variations) for each dataset. This assumption enabled the proposed TI-RBMs to achieve significantly better classification performance than the baseline method, since the data-specific transformation information was encoded in the TI-RBM.

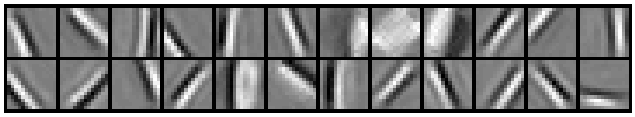
However, for natural images, it is not reasonable to assume such *global* transformations due to the complex image structures. In fact, recent literature [146, 84, 132] suggests that some level of invariance to *local* transformations (e.g., few pixel translation or coordinate-wise noise) leads to improved performance in classification. From this viewpoint, it makes more sense to learn representations with local receptive fields



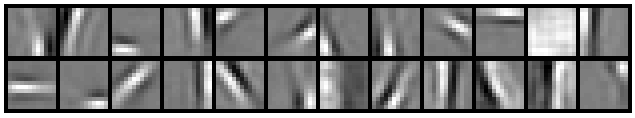
(a) sparse RBM (baseline)



(b) sparse TI-RBM with translation



(c) sparse TI-RBM with rotation



(d) sparse TI-RBM with scaling

Figure 3.4: Visualization of filters trained with RBM and TI-RBMs on natural images. We trained 24 filters and used nine translations with a step size of 1 pixel, five rotations with a step size of $\pi/8$ radian, and two-level scale transformations with a step size (gs) of 2 pixels, respectively.

that are invariant to *generic* image transformations (e.g., small amounts of translation, rotation, and scaling), which does not require data-specific prior information.

We visualize the learned TI-RBM filters in Figure 3.4, where we used the 14×14 natural image patches taken from the van Hateren dataset [130]. The baseline model (sparse RBM) learns many similar vertical edges (Figure 3.4(a)) that are shifted by a few pixels, whereas our methods can learn diverse patterns, including diagonal and horizontal edges, as shown in Figure 3.4(b), 3.4(c), and 3.4(d).

These results suggest that TI-RBMs can learn diverse sets of filters, which is reminiscent of the effects of convolutional training [60]. However, our model is much easier to train than convolutional models, and it can further handle generic transformations beyond translations.

Methods (Linear SVM)	Accuracy
sparse RBM† (baseline)	72.4%
sparse TI-RBM (scaling only)	76.8%
sparse TI-RBM (rotation only)	77.7%
sparse TI-RBM (translation only)	77.5%
sparse TI-RBM (combined)	78.8%
sparse TI-RBM (combined, K= 4,000)	80.1%
TI-OMP-1/T (combined)	80.7%
TI-OMP-1/T (combined, K= 4,000)	**82.2**%
VQ (K= 4,000)†	79.6%
OMP-1/T (K= 1,600)‡	79.4%
OMP-1/T (K= 6,000)‡	81.5%
convolutional DBN [67]	78.9%
deep NN [18]	80.5%
deep NN [20]	82.0%

Table 3.2: Test classification accuracy on CIFAR-10 dataset. 1,600 filters were used unless otherwise stated. The numbers with † and ‡ are from [21] and [19], respectively.

3.4.3 Object recognition

We evaluated on image classification tasks using two datasets. First, we tested on the widely used CIFAR-10 dataset [66], which is composed of 50,000 training and 10,000 testing examples with 10 categories. Rather than learning features from the whole image (32×32 pixels), we trained TI-RBMs on local image patches while keeping the RGB channels. As suggested by Coates et al. [21], we used the fixed filter size $w = 6$ and determined the receptive field size depending on the types of transformations.[4] Then, after unsupervised training with TI-RBM, we used the convolutional feature extraction scheme, also following Coates et al. [21]. Specifically, we computed the TI-RBM pooling-unit activations for each local $r \times r$ pixel patch that was densely extracted with a stride of 1 pixel, and averaged the patch-level activations in the same quadrant (with 2×2 regions). Eventually, this procedure yielded $4K$-dimensional feature vectors for each image, which were fed into an L2-regularized linear SVM. We performed 5-fold cross

[4]For example, we used $r = 6$ for rotations. For both scale variations or translations, we used $r = 8$ and $gs = 2$.

Methods (Linear SVM)	Acc.± std.
sparse RBM	$55.0 \pm 0.5\%$
sparse TI-RBM (scaling only)	$55.9 \pm 0.7\%$
sparse TI-RBM (rotation only)	$57.0 \pm 0.7\%$
sparse TI-RBM (translation only)	$57.8 \pm 0.5\%$
sparse TI-RBM (combined)	$58.7 \pm 0.5\%$
VQ (K= 1,600) [19]	$54.9 \pm 0.4\%$
SC (K= 1,600) [19]	$59.0 \pm 0.8\%$
deep NN [20]	$\mathbf{60.1} \pm 1.0\%$

Table 3.3: Test classification accuracy on STL-10. 1,600 filters were used for all experiments.

validation to determine the hyperparameter C.

For comparison to the baseline model, we separately evaluated the sparse TI-RBMs with a single type of transformation (translation, rotation, or scaling) using $K = 1,600$. As shown in Table 3.2, each single type of transformation in TI-RBMs brought a significant performance gain over the baseline sparse RBMs. The classification performance was further improved by combining different types of transformations into a single model.

In addition, we also report the classification results obtained using TI-OMP-1 (see Section 3.3.4) for unsupervised training. In this experiment, we used the following two-sided soft thresholding encoding function:

$$f_j = \max_s \left\{ \max(\mathbf{w}_j^T T_s \mathbf{v} - \alpha, 0) \right\}$$
$$f_{j+K} = \max_s \left\{ \max(-\mathbf{w}_j^T T_s \mathbf{v} - \alpha, 0) \right\},$$

where α is a constant threshold that was cross-validated. As a result, we observed about 1% improvement over the baseline method (OMP-1/T) using 1,600 filters, which supports the argument that our transformation-invariant feature learning framework can be effectively transferred to other unsupervised learning methods. Finally, by increasing the number of filters (K= 4,000), we obtained better results (82.2%) than

Methods (Linear SVM)	Accuracy
MFCC	68.2%
sparse RBM	76.3%
sparse TI-RBM	**77.6**%
sparse coding [98]	76.8%
sparse filtering [98]	75.7%

Table 3.4: Phone classification accuracy on the TIMIT core test set using linear SVMs.

the previously published results using single-layer models, as well as those using deep networks.

We also performed the object classification task on STL-10 dataset [21], which is more challenging due to the smaller number of labeled training examples (100 per class for each training fold). Since the original images are 96×96 pixels, we down-sampled the images into 32×32 pixels, while keeping the RGB channels. We followed the same unsupervised training and classification pipeline as we did for CIFAR-10. As reported in Table 3.3, there were consistent improvements in classification accuracy by incorporating the various transformations in learning algorithms. Finally, we achieved 58.7% accuracy using 1,600 filters, which is competitive to the best published single layer result (59.0%).

3.4.4 Phone classification

To show the broad applicability of our method to other data types, we report the 39-way phone classification accuracy on the TIMIT dataset. By following [98], we generated 39-dimensional MFCC features and used 11 contiguous frames of them as an input patch. For TI-RBMs, we applied three temporal translations with the stride of 1 frame.

First, we compared the classification accuracy using the linear SVM to evaluate the performance gain coming from the unsupervised learning algorithms, by following

Methods (RBF-kernel SVM)	Accuracy
MFCC (baseline)	80.0%
MFCC + TI-RBM (K= 256)	81.0%
MFCC + TI-RBM (K= 512)	**81.5**%
MFCC + SC [98]	80.1%
MFCC + SF [98]	80.5%
MFCC + CDBN [80]	80.3%
H-LMGMM [17]	81.3%

Table 3.5: Phone classification accuracy on the TIMIT core test set using RBF-kernel SVMs.

the experimental setup in [98].[5] As reported in Table 3.4, the TI-RBM showed an improvement over the sparse RBM, as well as the sparse coding and sparse filtering.

In the next setting, we used the RBF-kernel SVM [16] on the extracted features that are concatenated with MFCC features. We used default RBF kernel width for all experiments and performed cross-validation on the C values. As shown in Table 3.5, combining MFCC with TI-RBM features was the most effective and resulted in 1% improvement in classification accuracy over the baseline MFCC features. By increasing the number of TI-RBM features to 512, we were able to beat the best published results on the TIMIT phone classification tasks using hierarchical LM-GMM classifier [17].

3.5 Conclusion

In this chapter, we proposed novel feature learning algorithms that can achieve invariance to the set of pre-defined transformations. Our method can handle general transformations (e.g., translation, rotation, and scaling), and we experimentally showed that learning invariant features for such transformations leads to strong classification performance. In future work, we plan to work on learning transformations from the data and combine it with our algorithm. By automatically learning transformation matrices

[5]We used $K = 256$ with a two-sided encoding function by using the positive and negative weight matrices $[\mathbf{W}, -\mathbf{W}]$, as suggested by Ngiam et al. [98].

from the data, we will be able to learn more robust features, which will potentially lead to significantly better feature representations.

CHAPTER IV

Learning and Selecting Features Jointly with Point-wise Gated Boltzmann Machines

4.1 Introduction

One fundamental difficulty in building algorithms that can robustly learn from complex real-world data is to deal with significant noise and irrelevant patterns. In particular, let's consider a problem of *learning from scratch*, assuming the lack of useful raw features. Here, the challenge is how to learn a robust representation that can distinguish important (e.g., task-relevant) patterns from significant amounts of distracting (e.g., task-irrelevant) patterns.

For constructing useful features, unsupervised feature learning [46, 9, 91, 8] has emerged as a powerful tool in learning representations from unlabeled data. In many real-world problems, however, the data is not cleaned up and contains significant amounts of irrelevant sensory patterns. In other words, *not all patterns are equally important.*

In this case, the unsupervised learning methods may blindly represent the irrelevant patterns using the majority of the learned high-level features, and it becomes even more difficult to learn task-relevant higher-layer features (e.g., by stacking). Although there are ways to incorporate supervision (e.g., supervised fine-tuning), learning is still

challenging when the data contains lots of irrelevant patterns, as shown in [73].

To deal with such complex data, one may envision using feature selection. Indeed, feature selection [57, 145, 139, 39] is an effective method for distinguishing useful raw features from irrelevant raw features. However, feature selection may fail if there are no good raw features to start with.

To address this issue, we propose to combine feature learning and feature selection coherently in a unified framework. Intuitively speaking, given that unsupervised feature learning can find partially useful high-level abstractions, it may be easier to apply feature selection on learned high-level features to distinguish the task-relevant ones from the task-irrelevant ones. Then, the task-relevant high-level features can be used to trace back where such important patterns occur. This information can help the learning algorithm to focus on these task-relevant raw features (i.e., visible units corresponding to task-relevant patterns), while ignoring the rest.

In this section, we formulate a generative feature learning algorithm called the *point-wise gated Boltzmann machine* (PGBM). Our model performs feature selection not only on learned high-level features (i.e., hidden units), but also on raw features (i.e., visible units) through a gating mechanism using stochastic "switch units." The switch units allow our model to estimate where the task-relevant patterns occur, and make only those visible units to contribute to the final prediction through multiplicative interaction. The model ignores the task-irrelevant portion of the raw features, thus it performs *dynamic feature selection* (i.e., choosing a variable subset of raw features depending on semantic interpretation of the individual example).

We evaluate our models in two ways: 1) recognizing handwritten digits in the irrelevant background, and 2) localizing and classifying objects in the natural scenes. In the first experiment, our method shows strong performance in learning features and distinguishing task-relevant features from task-irrelevant features. In the second experiment, our model shows promising results in distinguishing foreground objects from

background scenes and localizing the object bounding boxes in a weakly-supervised way, which leads to an improved object recognition performance.

We summarize our main contributions as follows:

- We propose the PGBMs that jointly perform feature learning and feature selection in a unified framework.
- We propose the semi-supervised PGBM and show its effectiveness when given a small amount of labeled data and a large amount of unlabeled data.
- We show that the PGBM is an effective building block for constructing deep networks.
- We propose a convolutional extension of the PGBM. We further show that this can be used for weakly-supervised object localization. Using predicted bounding boxes of objects, we demonstrate state-of-the-art object recognition performance on the Caltech 101 dataset.
- We achieve a significant improvement over state-of-the-art on variations of the MNIST dataset.

4.2 Related Work

As mentioned in section 4.3.1, the PGBM can be viewed as an extension of the implicit mixture of RBM (imRBM) [96] that allows per-visible-unit switching. Although these two models look similar, the *per-visible-unit switching* property of the PGBM brings an important benefit over the imRBM because it allows the PGBM to represent data with multiple components, each of which focusing on different part of the raw features. In particular, the supervised PGBM represents the data with two groups of hidden units (one containing task-relevant hidden units and the other containing task-irrelevant hidden units). In contrast, the imRBM uses a single component to represent the data, and thus cannot distinguish between the relevant and irrelevant patterns when

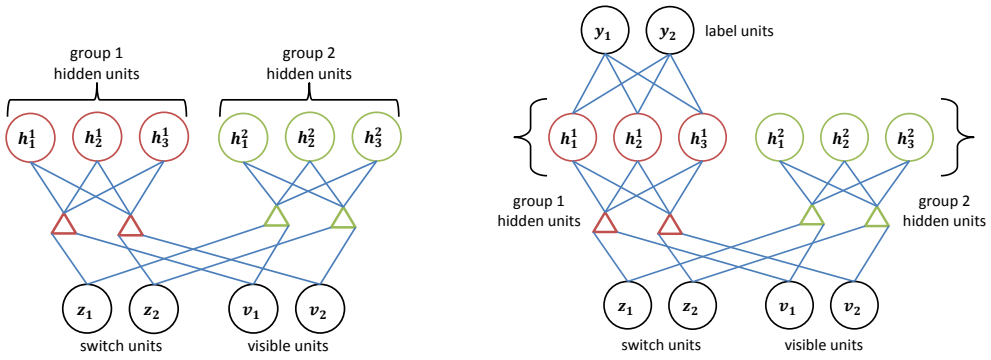
the data contains significant amounts of irrelevant patterns.

The discriminative RBM (discRBM) [72] is another model that can learn discriminative features using class labels. We argue that, however, the PGBM can be more robust to noisy data since it can prune out (or re-weigh) the irrelevant features dynamically for each data instance using switch unit activations, whereas the discRBM accumulates the contribution from noisy visible units with the fixed weights applied to all data instances. In section 4.4.1, we empirically show that the PGBM significantly outperforms both the imRBM and the discRBM in classifying handwritten digits in the presence of irrelevant background patterns.

Rifai et al. [109] proposed the contractive discriminative analysis (CDA). Similarly to the PGBM, the CDA has two groups of hidden units, one of which is connected to labels. The difference is that the CDA is a feed-forward neural network which can learn distinct features for each group with a contractive penalty term, while the PGBM is a probabilistic model that performs *generative* feature selection through a multiplicative interaction between visible, hidden, and switch units.

The robust Boltzmann machine (RoBM) [126] shares its motivation with our work, though there are several technical differences. First, the RoBM models each background noise unit with a unimodal Gaussian distribution, whereas the PGBM models the background visible units with more complicated multimodal distribution with a group of hidden units. Furthermore, the PGBM can directly learn from the noisy data with class labels, but the RoBM requires clean data to pre-train the GRBM.

In terms of energy function, the unsupervised PGBM can be viewed as having a similar formulation to the masked RBM [76, 43]. However, our main motivation is to perform joint feature selection at both low-level and high-level. Specifically, the difference becomes clearer when we use class labels in supervised PGBM that performs generative feature selection, as discussed in section 4.3.2.



(a) PGBM (b) supervised PGBM

Figure 4.1: Graphical model representation of the (a) PGBM and (b) supervised PGBM with two groups of hidden units. The Bernoulli switch unit z_i specifies which of the two components models the visible unit v_i. In other words, when $z_i = 1$, v_i is generated from the hidden units in the first group (shown in red); when $z_i = 2$, v_i is generated from the hidden units in the second group (shown in green).

4.3 Proposed Models

In this section, we propose the point-wise gated Boltzmann machine and its extensions. In Section 4.3.1, we describe the basic unsupervised PGBM that learns and groups features into semantically distinct components. In Section 4.3.2, we propose the supervised PGBM that uses class labels as a top-down feedback to partition the hidden units into the task-relevant and the task-irrelevant components. In Section 4.3.3, we propose the semi-supervised PGBM that uses unlabeled data as a regularizer when there are only a small number of labeled training examples. Furthermore, we construct a deep network using the PGBM as a building block, where we stack neural network layer(s) on top of the PGBM's task-relevant hidden units. Finally, we present the convolutional extension of the PGBM that can efficiently handle spatially correlated high-dimensional data.

4.3.1 Point-wise gated Boltzmann machines

When we deal with complex data, it is desirable for a learning algorithm to distinguish semantically distinct patterns. For example, an object recognition algorithm

may improve its performance if it can separate the foreground object patterns from the background clutters. To model this, we propose to represent each visible unit as a mixture model when conditioned on the hidden units, where each group of hidden units can generate the corresponding mixture component.

Before going into details, we describe the generative process of the PGBM as follows: (1) The hidden units are partitioned into *components*, each of which defines a distinct distribution over the visible units. (2) Conditioning on the hidden units, we sample the *switch units*. (3) The switch units determine which *component* generates the corresponding visible units. A schematic diagram is shown in Figure 4.1(a) as an undirected graphical model.

The PGBM with R mixture components has a multinomial switch unit, denoted $z_i \in \{1, \cdots, R\}$,[1] for each visible unit v_i. The PGBM imposes element-wise multiplicative interaction between the paired switch and visible units, as shown in Figure 4.1(a). Now, we define the energy function of the PGBM as follows:

$$E^{\mathrm{U}}(\mathbf{v}, \mathbf{z}, \mathbf{h}) = -\sum_{r=1}^{R}\sum_{i=1}^{D}\sum_{k=1}^{K_r}(z_i^r v_i)\, W_{ik}^r h_k^r - \sum_{r=1}^{R}\sum_{k=1}^{K_r} b_k^r h_k^r - \sum_{r=1}^{R}\sum_{i=1}^{D}(z_i^r v_i)\, c_i^r, \qquad (4.1)$$

$$\text{s.t.} \qquad \sum_{r=1}^{R} z_i^r = 1,\ i = 1, \cdots, D.$$

Here, $\mathbf{v}$, $\mathbf{z}^r$ and $\mathbf{h}$ are the visible, switch and hidden unit binary vectors, respectively, and the model parameters W_{ik}^r, b_k^r, c_i^r are the weights, hidden biases, and the visible biases of r-th component. The binary-valued switch unit z_i^r is activated (i.e. takes value 1) if and only if its paired visible unit v_i is assigned to the r-th component, and its conditional probability given hidden units follows a multinomial distribution over R

[1]For convenience, we also use the vector representation for switch unit in boldface, i.e., $\mathbf{z}_i = [z_i^1, \cdots, z_i^R]^T \in \{0,1\}^R$, where $\sum_{r=1}^{R} z_i^r = 1$, for each visible unit v_i.

categories. The energy function can be written in matrix form as follows:

$$E^{\mathrm{U}}(\mathbf{v}, \mathbf{z}, \mathbf{h}) = -\sum_{r=1}^{R}(\mathbf{z}^r \odot \mathbf{v})^T \mathbf{W}^r \mathbf{h}^r - \sum_{r=1}^{R}(\mathbf{b}^r)^T \mathbf{h}^r - \sum_{r=1}^{R}(\mathbf{c}^r)^T(\mathbf{z}^r \odot \mathbf{v}), \tag{4.2}$$

where the operator $\odot$ denotes element-wise multiplication, i.e., $(\mathbf{z}^r \odot \mathbf{v})_i = z_i^r v_i$.

The visible, hidden, and switch units are conditionally independent given the other two types of units, and the conditional probabilities can be written as follows:

$$P\left(h_k^r = 1 \mid \mathbf{z}, \mathbf{v}\right) = \sigma\left((\mathbf{z}^r \odot \mathbf{v})^T \mathbf{W}_{\cdot k}^r + b_k^r\right), \tag{4.3}$$

$$P\left(v_i = 1 \mid \mathbf{z}, \mathbf{h}\right) = \sigma\left(\sum_r z_i^r \left(\mathbf{W}_{i\cdot}^r \mathbf{h}^r + c_i^r\right)\right), \tag{4.4}$$

$$P\left(z_i^r = 1 \mid \mathbf{v}, \mathbf{h}\right) = \frac{\exp\left(v_i \left(\mathbf{W}_{i\cdot}^r \mathbf{h}^r + c_i^r\right)\right)}{\sum_s \exp\left(v_i \left(\mathbf{W}_{i\cdot}^s \mathbf{h}^s + c_i^s\right)\right)}, \tag{4.5}$$

where we use $\mathbf{W}_{i\cdot}$ to denote i-th row, and $\mathbf{W}_{\cdot k}$ to denote k-th column of $\mathbf{W}$.

It is important to note that, while inferring the hidden units, our model *gates* (or re-weighs) each visible unit v_i according to the corresponding switch units z_i^r (Equation 4.3). In other words, the point-wise multiplicative interaction between the switch and the visible units allows the hidden units in each component to *focus* on a specific part of the data, and this makes the hidden units in one component to be robust to the patterns learned by other components. Moreover, the top-down signal from the hidden units encourages assigning the same mixture component to semantically-related visible units during the switch unit inference, and therefore we can prune out the irrelevant raw features dynamically for each example.

It is worth noting that, when we tie all switch units (i.e., $\mathbf{z}_i = \mathbf{z}$ for all i), the PGBM becomes equivalent to the implicit mixture of restricted Boltzmann machine [96]. Furthermore, given that there is a multiplicative interaction between three types of variables, the PGBM can be understood in the context of higher-order Boltzmann machines [114, 93].

We train the PGBM with stochastic gradient descent using contrastive divergence. Since the exact inference is intractable due to the three-way interaction, we use mean-field or alternating Gibbs sampling (i.e., sample one type of variables given the other two types using Equations (4.3),(4.4), and (4.5)) for approximate inference.

4.3.2 Generative feature selection with supervised PGBMs

Although the PGBM can learn to group distinct features for each mixture component, it doesn't necessarily learn discriminative features automatically since the generative training is done in an unsupervised way. One way to make the PGBM implicitly perform feature selection (i.e., distinguish features into different groups based on their relevance to the task) is to provide a good initialization of the model parameters. For example, we can pre-train the regular RBM and divide the hidden units into two groups based on the score from the simple feature selection algorithms such as the t-test[2] to initialize the weight matrices of the PGBM. As we will discuss in Section 4.4, this approach improves classification performance of the PGBMs.

Furthermore, to make use of class labels during the generative training, we propose a *supervised PGBM* that only connects the hidden units in the task-relevant component(s) to the label units. The graphical model representation is shown in Figure 4.1(b). By transferring the label information to the raw features through the task-relevant hidden units, the supervised PGBM can perform *generative feature selection* both at the high-level (i.e., using only a subset of hidden units for classification) and the low-level (e.g., dynamically blocking the influence of the task-irrelevant visible units) in a unified way.

For simplicity, we present the supervised PGBM with two mixture components, where we assign the first component to be task-relevant. The energy function is defined

[2] `http://featureselection.asu.edu/software.php`

as follows:

$$E^{\mathrm{S}}(\mathbf{v}, \mathbf{z}, \mathbf{h}, \mathbf{y}) = E^{\mathrm{U}}(\mathbf{v}, \mathbf{z}, \mathbf{h}) - \mathbf{y}^T \mathbf{U} \mathbf{h}^1 - \mathbf{d}^T \mathbf{y} \tag{4.6}$$

subject to $z_i^1 + z_i^2 = 1,\ i = 1, \cdots, D$. The label vector $\mathbf{y} \in \{0, 1\}^L$ is in the 1-of-L representation. $\mathbf{U} \in \mathbb{R}^{L \times K_1}$ is the weight matrix between the task-relevant hidden units and the label units, and $\mathbf{d}$ is the label bias vector. The conditional probabilities can be written as follows:

$$P\left(h_k^1 = 1 \mid \mathbf{z}, \mathbf{v}, \mathbf{y}\right) = \sigma\left((\mathbf{z}^1 \odot \mathbf{v})^T \mathbf{W}_{\cdot k}^1 + b_k^1 + \mathbf{U}_{\cdot k}^T \mathbf{y}\right), \tag{4.7}$$

$$P\left(y_l = 1 \mid \mathbf{h}^1\right) = \frac{\exp\left(\mathbf{U}_{l\cdot} \mathbf{h}^1 + d_l\right)}{\sum_s \exp\left(\mathbf{U}_{s\cdot} \mathbf{h}^1 + d_s\right)}. \tag{4.8}$$

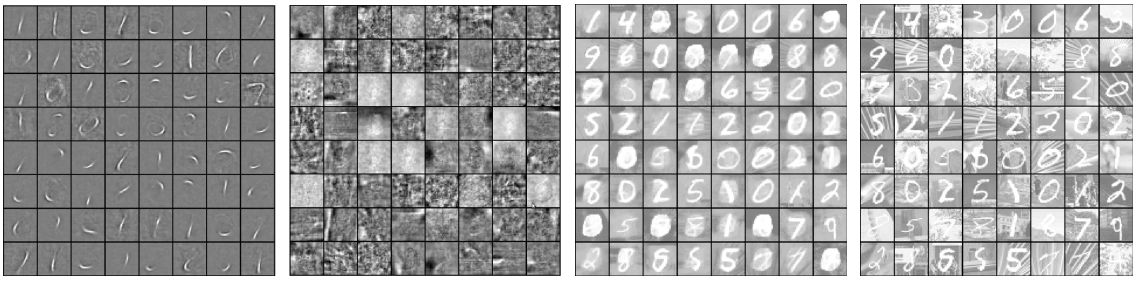
The conditional probabilities of the visible and switch units are the same as Equations (4.4) and (4.5). As we can see in Equation (4.7), the label information, together with the switch units, modulates the hidden unit activations in the first (task-relevant) component, and this in turn encourages the switch units z_i^1 to activate at the task-relevant visible units[3] during the iterative approximate inference.

We can train the supervised PGBM in generative criteria whose objective is to maximize the joint log-likelihood of the visible and the label units [72]. Similarly to that of PGBM, the inference can be done with alternating Gibbs sampling between Equations (4.4),(4.5),(4.7), and (4.8).

4.3.3 Variations of the model

Deep networks. The PGBM can be used as a building block of deep networks. For example, we can use it as a first layer block and stack neural networks on the hidden units of task-relevant components. Since the PGBM can select the task-relevant

[3]Note: In our model, we call that a visible unit (a raw feature) is "task-relevant" (or "task-irrelevant") if its switch unit for the task-relevant (or task-irrelevant) component is active, respectively.



(a) task-relevant filters (b) task-irrelevant filters (c) switch unit activation (d) original images

Figure 4.2: Visualization of (a, b) filters corresponding to two components learned from the PGBM, (c) activation of switch units, and (d) corresponding original images on *bg-image* dataset. Specifically, (a) represents the group of hidden units that activates for the foreground digits (task-relevant), and (b) represents the group of hidden units that activates for the background images (task-irrelevant). See text for details.

hidden units with supervision, the higher-layer networks can focus on the task-relevant information. In Section 4.4.1, we show that the two-layer model, where we stack a single-layer neural network on top of a PGBM's task-relevant component, was sufficient to outperform existing state-of-the-art classification performance on the variations of MNIST dataset with irrelevant backgrounds.

Semi-supervised PGBM. There are many classification tasks where we are given a large number of unlabeled examples in addition to only a few labeled training examples. For such scenario, it is important to include unlabeled examples during the training to generalize well to the unseen data, and thus avoid overfitting. Larochelle and Bengio [72] proposed the semi-supervised training of the discriminative restricted Boltzmann machine by combining the generative objective defined on the unlabeled examples with the discriminative objective. Similarly to their approach, the supervised PGBM can be trained in a semi-supervised learning framework. Specifically, we can use the input data log-likelihood defined on the unlabeled data as a regularizer.

The joint distribution of the supervised PGBM with two mixture components (i.e.,

$R = 2$) is written as follows:

$$P(\mathbf{v}, \mathbf{z}, \mathbf{h}, \mathbf{y}) = \frac{1}{Z} \exp\left(-E(\mathbf{v}, \mathbf{z}, \mathbf{h}, \mathbf{y})\right), \tag{4.9}$$

where

$$E(\mathbf{v}, \mathbf{z}, \mathbf{h}, \mathbf{y}) = -\sum_{r=1}^{R} (\mathbf{z}^r \odot \mathbf{v})^T \mathbf{W}^r \mathbf{h}^r - \sum_{r=1}^{R} (\mathbf{b}^r)^T \mathbf{h}^r - \sum_{r=1}^{R} (\mathbf{c}^r)^T (\mathbf{z}^r \odot \mathbf{v}) - \mathbf{y}^T \mathbf{U} \mathbf{h}^1 - \mathbf{d}^T \mathbf{y}$$

The semi-supervised PGBM is trained to minimize the following objective function:

$$\mathcal{L}_{semi} = \sum_{(\mathbf{v},\mathbf{y}) \in \mathcal{D}_\mathrm{l}} -\log P(\mathbf{v}, \mathbf{y}) + \alpha \sum_{\mathbf{v} \in \mathcal{D}_\mathrm{u}} -\log P(\mathbf{v}) \tag{4.10}$$

where $\mathcal{D}_\mathrm{l}$ and $\mathcal{D}_\mathrm{u}$ denote the set of labeled and unlabeled examples, respectively. The coefficient α controls the contribution of unlabeled data during the training.

We use CD to train the semi-supervised PGBM. The gradient of the first term can be approximated as described in Section 4.3.2. Similarly, the gradient of the second term with respect to the parameters $\Theta = \{\mathbf{W}^r, \mathbf{b}^r, \mathbf{c}^r, \mathbf{U}, \mathbf{d}\}$ can be computed as follows:

$$\frac{\partial \log P(\mathbf{v})}{\partial \theta} = \mathbb{E}_{P(\mathbf{z},\mathbf{h},\mathbf{y}|\mathbf{v})} \left[\frac{\partial}{\partial \theta} E(\mathbf{v}, \mathbf{z}, \mathbf{h}, \mathbf{y}) \right] - \mathbb{E}_{P(\mathbf{v},\mathbf{z},\mathbf{h},\mathbf{y})} \left[\frac{\partial}{\partial \theta} E(\mathbf{v}, \mathbf{z}, \mathbf{h}, \mathbf{y}) \right] \tag{4.11}$$

Since exact inference is intractable, we use alternating Gibbs sampling (or mean-field) with the following conditional distributions:

$$P\left(y_l = 1 \mid \mathbf{h}^1\right) = \frac{\exp\left(\mathbf{U}_{l\bullet}\mathbf{h}^1 + d_l\right)}{\sum_s \exp\left(\mathbf{U}_{s\bullet}\mathbf{h}^1 + d_s\right)}, \tag{4.12}$$

$$P\left(h_k^1 = 1 \mid \mathbf{v}, \mathbf{z}, \mathbf{y}\right) = \sigma\left((\mathbf{z}^1 \odot \mathbf{v})^T \mathbf{W}_{\bullet k}^1 + b_k^1 + \mathbf{U}_{\bullet k}^T \mathbf{y}\right), \tag{4.13}$$

$$P\left(h_k^2 = 1 \mid \mathbf{v}, \mathbf{z}\right) = \sigma\left((\mathbf{z}^2 \odot \mathbf{v})^T \mathbf{W}_{\bullet k}^2 + b_k^2\right), \tag{4.14}$$

$$P\left(z_i^r = 1 \mid \mathbf{v}, \mathbf{h}\right) = \frac{\exp\left(v_i\left(\mathbf{W}_{i\bullet}^r \mathbf{h}^r + c_i^r\right)\right)}{\sum_s \exp\left(v_i\left(\mathbf{W}_{i\bullet}^s \mathbf{h}^s + c_i^s\right)\right)}, \tag{4.15}$$

$$P\left(v_i = 1 \mid \mathbf{z}, \mathbf{h}\right) = \sigma\left(\sum_r z_i^r \left(\mathbf{W}_{i\bullet}^r \mathbf{h}^r + c_i^r\right)\right). \tag{4.16}$$

In the positive phase, we iterate over the Equations (4.12)∼(4.15). In the negative phase, we iterate over all five Equations (4.12)∼(4.16). For classification, we use the posterior of the first (i.e. task-relevant) component hidden units as an input to the linear SVM [30].

Convolutional PGBM. Convolutional models can be useful in representing spatially or temporally correlated data. The PGBM can be extended to a convolutional setting [84], where we share the filter weights over different locations in large images. Specifically, we introduce multinomial switch units $z_{m,n} \in \{1, \cdots, R\}$, $m, n = 1, \cdots, N_V$ for each input coordinate of the visible unit. Note that each switch unit is shared across all input channels. The energy function of the CPGBM with R components is written as follows: $E(\mathbf{v}, \mathbf{z}, \mathbf{h}) =$

$$-\sum_{r=1}^{R}\left[\sum_{l=1}^{L}\sum_{m,n} z_{m,n}^r v_{m,n}^l c_l^r - \sum_{k=1}^{K_r}\sum_{i,j} h_{i,j}^{r,k}\left(\sum_{l=1}^{L}(\widetilde{\mathbf{W}}^{r,k,l} * (\mathbf{z}^r \odot \mathbf{v}^l))_{i,j} + b_k^r\right)\right], \tag{4.17}$$

subject to $\sum_{r=1}^{R} z_{m,n}^r = 1$, $m, n = 1, \cdots, N_V$. The operator $\odot$ denotes an element-wise multiplication between two matrices, i.e., $(\mathbf{z} \odot \mathbf{v})_{m,n} = z_{m,n} v_{m,n}$. We can interpret the CPGBM's energy function described in Equation (4.17) as a sum of energy functions of R CRBM components whose visible units are filtered by the corresponding switch unit

activations, i.e., $\mathbf{v}^r = \mathbf{z}^r \odot \mathbf{v}$, as follows:

$$E(\mathbf{v}, \mathbf{z}, \mathbf{h}) = \sum_{r=1}^{R} E_{\text{CRBM}}(\mathbf{v}^r, \mathbf{h}^r; \mathbf{W}^r, \mathbf{b}^r, \mathbf{c}^r). \tag{4.18}$$

where $E_{\text{CRBM}}(\mathbf{v}, \mathbf{h})$ is defined in Equation (2.5).

Due to the three way interaction among visible, hidden and switch units, exact inference is intractable. Instead, we use alternating Gibbs sampling (or mean-field approximation) to compute the posterior distribution. The conditional probabilities of hidden, switch, and visible units given the other two types of variables can be written as follows:

$$P(h_{i,j}^{r,k} = 1|\mathbf{v}, \mathbf{z}) = \sigma\left(\sum_l (\widetilde{\mathbf{W}}^{r,k,l} * (\mathbf{z}^r \odot \mathbf{v}^l))_{i,j} + b_k^r\right) \tag{4.19}$$

$$P(z_{m,n}^r = 1|\mathbf{v}, \mathbf{h}) = \frac{\exp\left(\sum_l v_{m,n}^l (\sum_k (\mathbf{W}^{r,k,l} * \mathbf{h}^{r,k})_{m,n} + c_l^r)\right)}{\sum_s \exp\left(\sum_l v_{m,n}^l (\sum_k (\mathbf{W}^{s,k,l} * \mathbf{h}^{s,k})_{m,n} + c_l^s)\right)} \tag{4.20}$$

$$P(v_{m,n}^l = 1|\mathbf{h}, \mathbf{z}) = \sigma\left(\sum_r z_{m,n}^r \left[\sum_k (\mathbf{W}^{r,k,l} * \mathbf{h}^{r,k})_{m,n} + c_l^r\right]\right) \tag{4.21}$$

Similarly to the CRBM, the CPGBM can be trained using CD.

In Section 4.4.2, we present the convolutional PGBM with an application to the weakly supervised foreground object localization problem. Furthermore, by locating the bounding box at the foreground object accurately, we achieved state-of-the-art recognition performance in Caltech 101.

4.4 Experiments

4.4.1 Handwritten digit recognition with background noise

We evaluated the capability of the proposed models in learning task-relevant features from noisy data. We tested the single-layer PGBMs and their extensions on the

variations of MNIST dataset: *bg-rand*, *bg-image*, *rot-bg-image*, and *rot-bg-rand*.[4] The first two datasets use uniform noise or natural images as background patterns. The other two have rotated digits in front of the corresponding background patterns. We used the PGBM with two components of 500 hidden units, and initialized with the pre-trained RBM using the feature selection as described in Section 4.3.2. We used mean-field for approximate inference for all our experiments.[5]

In Figure 4.2, we visualize the filters and the switch unit activations for *bg-image*. The foreground filters capture the task-relevant patterns resembling pen strokes (Figure 4.2(a)), while the background filters (Figure 4.2(b)) capture task-irrelevant patterns in the natural images. Further, the switch unit activations (the posterior probabilities that the input pixel belongs to the *foreground component*, Figure 4.2(c)) are high (colored in white) for the foreground digit pixels, and low (colored in gray) for the background pixels. This suggests that our model can *dynamically* separate the task-relevant raw features from the task-irrelevant raw features for each example.

For quantitative evaluation, we show test classification errors in Table 4.1. For all experiments with our single-layer models, we used the "task-relevant" hidden unit activations as the input for the linear SVM [30]. The single-layer PGBM significantly outperformed the baseline RBM, imRBM, and discRBM.[6] We did a careful model selection to choose the best hyperparameters for each of the compared models. These results suggest that the point-wise mixture hypothesis is effective in learning task-relevant features from complex data containing irrelevant patterns.

Generative feature selection. As a control experiment, we compared our model to the two-step model which we call "RBM-FS," where we first trained the RBM and

[4]The first three datasets are generated by Larochelle et al. [73]. We generated *rot-bg-rand* following the procedure described in [73].

[5]We have tested mean-field and alternating Gibbs sampling with $10 \sim 25$ iterations, and they showed similar results.

[6]We used "hybrid" discriminative RBM whose objective is a weighted sum of the discriminative (conditional) and generative (joint) likelihood.

Algorithm	*bg-rand*	*bg-image*	*rot-bg-image*	*rot-bg-rand*
RBM	11.39	15.42	49.89	51.97
imRBM	10.46	16.35	51.03	51.02
discRBM	10.29	15.56	48.34	48.28
RBM-FS	11.42	15.20	49.65	51.69
PGBM	7.27	13.33	45.45	45.53
supervised PGBM	**6.87**	**12.85**	**44.67**	**43.47**

Algorithm	*bg-rand*	*bg-image*	*rot-bg-image*	*rot-bg-rand*
DBN-3 [132]	6.73	16.31	47.39	-
CAE-2 [108]	10.90	15.50	45.23	-
PGBM+ DN-1	**6.08**	**12.25**	**36.76**	**30.41**

Table 4.1: Test classification errors of (top) single-layer and (bottom) multi-layer models on MNIST variation datasets. We used 10,000/2,000/50,000 splits for train, validation and test sets, and report the test classification errors without retraining the model after hyperparameter search over the validation set. For all RBM variants including imRBM, discRBM, and PGBM, we used sparsity regularizer [82]. The best performers among the single-layer models and the deep network models are both in bold.

selected a subset of hidden units using feature selection. As we see in Table 4.1, the RBM-FS is only marginally better (or sometimes worse) than the baseline RBM. However, the PGBM significantly outperforms the RBM-FS, which demonstrates the benefit of the joint training.

Semi-supervised learning. The supervised PGBM can be trained in a semi-supervised way as described in Section 4.3.3. We used the same experimental setting as in [72], and provided labels for only 10 percent of training examples (100 labeled examples for each digit category). We summarize the classification errors of semi-supervised PGBM, supervised PGBM, RBM and RBM-FS in Table 4.2. The semi-supervised PGBM consistently performed the best for all datasets, showing that semi-supervised training is effective in utilizing a large number of unlabeled examples.

Deep networks. Finally, we constructed a two-layer deep network by stacking one layer of neural network with 1,000 hidden units on the task-relevant component of the

Algorithm	*bg-rand*	*bg-image*	*rot-bg-image*	*rot-bg-rand*
RBM	17.43 ± 0.36	23.71 ± 0.34	63.94 ± 0.50	63.17 ± 0.32
RBM-FS	17.15 ± 0.46	**20.22** ± 0.31	61.76 ± 0.43	62.02 ± 0.81
supervised PGBM	16.15 ± 0.70	21.04 ± 0.18	**59.39** ± 0.58	63.82 ± 0.68
semi-supervised PGBM	**11.98** ± 0.80	**20.32** ± 0.15	**59.19** ± 0.68	**58.57** ± 0.49

Table 4.2: The mean and the standard deviation of the test classification errors of semi-supervised PGBM, supervised PGBM, RBM, and RBM-FS. We repeated 5 times with randomly sampled 1,000 labeled training examples in addition to the remaining 9,000 unlabeled training examples. The best model and those within the standard deviation are in bold.

PGBM. We used softmax classifier for fine-tuning of the second layer neural network. Table 4.1 shows that our deep network (referred to as "PGBM+DN-1") outperforms the DBN-3 and the stacked contractive autoencoder by a large margin. In particular, the result of the DBN-3 on *bg-image* implies that adding more layers to the DBN does not necessarily improve the performance when there are significant amounts of irrelevant patterns in the data. In contrast, the PGBM can block the task-irrelevant information from propagating to the higher layers, and hence it is an effective building block for deep networks. Finally, we note that, to the best of our knowledge, the PGBM+DN-1 achieved state-of-the-art classification performance on all datasets except *rot-bg-image*, where the transformation-invariant RBM [118] achieved 35.5% error by incorporating the rotational invariance.

4.4.2 Weakly supervised object segmentation for object recognition

In this section, we extend our model to learn groups of task-relevant features (i.e., foreground patterns) from the images with higher resolution, and apply it to weakly supervised object segmentation.

Weakly supervised object segmentation. Lee et al. [84] showed that the convolutional deep belief network (CDBN) composed of multiple layers of convolutional RBM (CRBM) can learn hierarchical feature representations from large images. In particular,

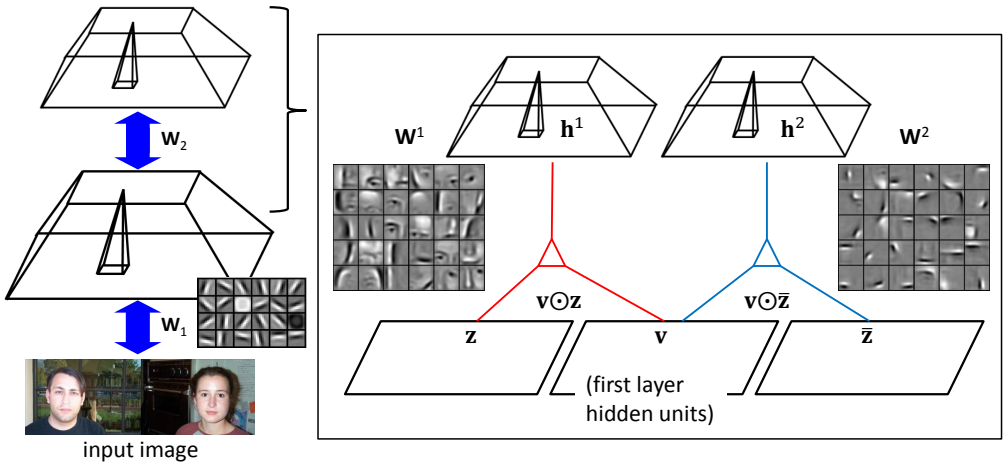
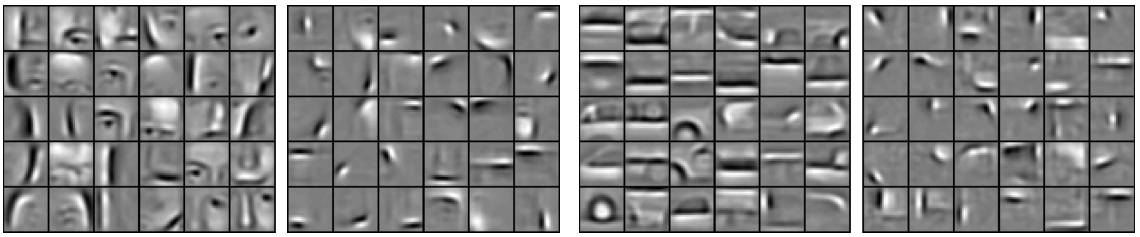


Figure 4.3: Architecture of the two-layer CPGDN model. We use the CRBM for the first layer, and the CPGBM with two mixture components for the second layer. We also visualize the filters for each mixture component learned from the "Face" category. In this figure, we use $\bar{z}$ for binary variable z to denote its complement, i.e., $\bar{z} = 1 - z$.

the first layer of the CDBN mostly learns generic edge filters, and the higher layers learn not only complex generic patterns, such as corners or contours, but also semantically meaningful features, such as object parts (e.g., eyes, nose, or wheels) in the second layer or whole objects (e.g., human face or car) in the third layer. To learn and group related features from large images, we propose the point-wise gated convolutional deep network (CPGDN), where we use the convolutional extension of the PGBM (CPGBM) as a building block.

Specifically, we construct the two-layer CPGDN by stacking the CPGBM on the first layer CRBM. This construction makes sense because the first layer features are mostly generic, and the class-specific features emerge in higher layers [84]. We train the CPGDN using greedy layer-wise training method, and perform feedforward inference in the first layer. We use mean-field in the second layer for approximate inference of switch and hidden units. In Figure 4.3, we show the architecture of two-layer CPGDN model that we used for weakly supervised object segmentation task.

We first trained a CPGDN with two mixture components only on the single class of images from Caltech 101 dataset [32]. For this experiment, we randomly initialized the



(a) Task-relevant (b) Task-irrelevant (c) Task-relevant (d) Task-irrelevant

Figure 4.4: Visualization of the second layer CPGBM features from "Faces" (a, b) and "Car side" (c, d) classes.

weights without pre-training. We visualize the second layer features trained on "Faces" and "Car side" classes in Figure 4.4. The CPGDN made a good distinction between the task-relevant patterns such as face parts and wheels, and the generic patterns. In Figure 4.5, we visualize the switch unit activation map, which shows that the switch units are selectively activated at the most informative region in each image. Interestingly, using this activation map, we can segment the object region from the background reasonably well, though our model is not specifically designed for image segmentation.

training images	15	30
Lazebnik et al. [74]	56.4%	64.6%
Griffin et al. [37]	59.0%	67.6%
Yang et al. [143]	67.0%	73.2%
Boureau et al. [15]	-	75.7%
Goh et al. [35]	71.1%	**78.9**%
RBM [119]	68.6%	74.9%
CRBM [119]	71.3%	77.8%
Our method + RBM	70.2%	76.8%
Our method + CRBM	**72.4**%	**78.9**%

Table 4.3: Test classification accuracy on Caltech 101.

Object recognition. Inspired by the CPGDN's ability to distinguish the foreground object from the background scene, we propose a novel object recognition pipeline on Caltech 101 dataset, where we first "crop" each image at the bounding box predicted using the switch unit activations of the CPGDN and perform classification using those

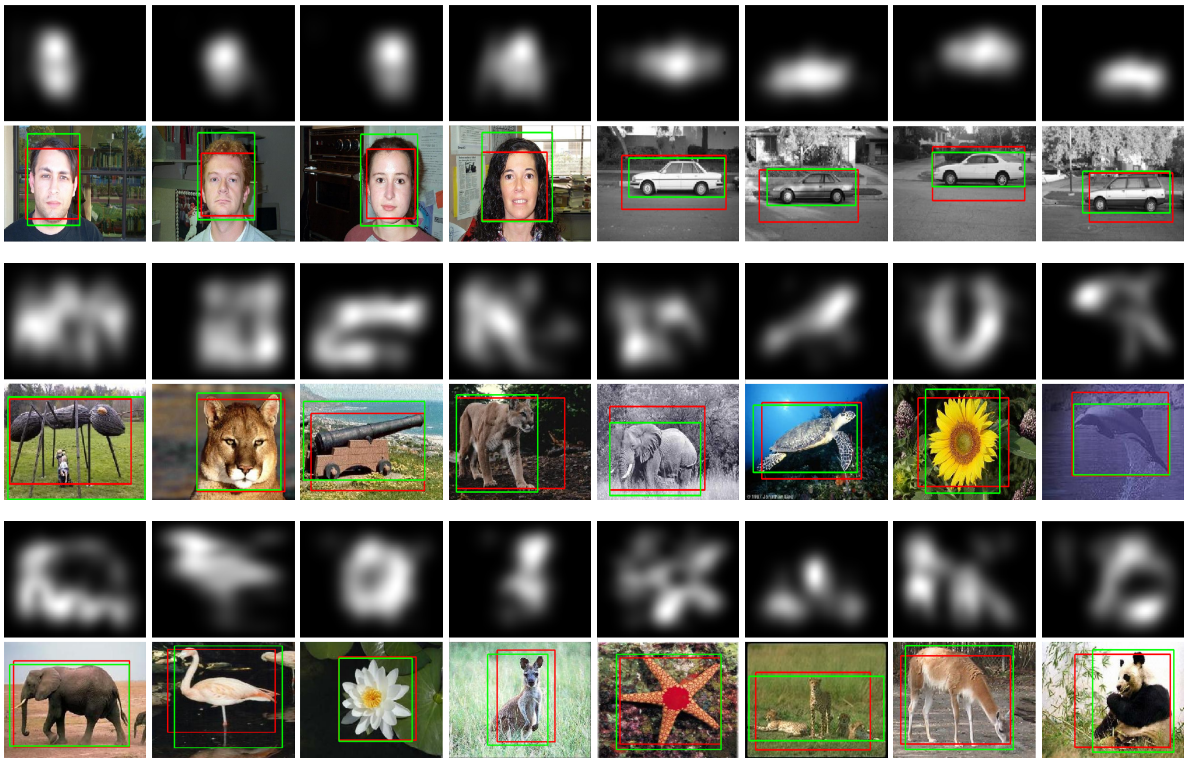


Figure 4.5: Visualization of the pairs of examples for switch unit activation map and the corresponding image below overlayed with the predicted (red) and the ground truth bounding boxes (green). The first row of examples are generated using the CPGDN trained only on either "Face" (left four examples) or "Car" (right four examples) classes. The second and third rows of examples are generated using the CPGDN trained on all categories of images from Caltech 101 dataset.

cropped images. Specifically, we used the CPGDN with two mixture components, each of which is composed of 100 hidden units. To train the model efficiently from many different classes of images, we pre-train a set of second layer CRBMs with a small number of hidden units (e.g., 30) for each class to capture more diverse and class-specific patterns, and perform feature selection on those CRBM features from all object categories to initialize the weights of the second layer CPGBM. Once we train the model, we compute the posterior of switch units arranged in 2d. To predict the bounding box, we compute the row-wise and column-wise cumulative sum of switch unit activations and select the region containing (5, 95) percentiles of the total activations as a bounding box. For classification, we followed the pipeline used in [119], which uses the Gaussian

(convolutional) RBMs with dense SIFT as input.

We first evaluated the bounding box detection accuracy. We declare that the bounding box prediction is correct when the average overlap ratio (the area of intersection divided by the union between the predicted and the ground truth bounding boxes) is greater than 0.5 [29]. We achieved average overlap ratio of 0.702 and detection accuracy of 88.3%.

Finally, we evaluated the classification accuracy using the cropped Caltech 101 dataset with CPGDN and summarize the results in Table 4.3. The object centered cropped images brought improvement in classification accuracies, such as 74.9% to 76.8% with RBM, and 77.8% to 78.9% with CRBM using 30 training images per class, respectively.[7] As a baseline, we also report the classification accuracy on the augmented dataset where we uniformly crop the center region across all the images with a fixed ratio. After cross-validating with different ratios, we obtain a worse classification accuracy of 75.8% with RBM using 30 training images per class. This suggests that the classification performance can be improved by localizing the object better than simply cropping the center region.

4.5 Conclusion

In this chapter, we proposed a point-wise gated Boltzmann machine that can effectively learn useful feature representations from data containing irrelevant patterns. Our methods achieve state-of-the-art classification performance on several datasets that contain irrelevant patterns. We believe our method holds promise in building a robust algorithm that can learn from large-scale, complex, sensory input data.

[7] We also performed the same experiment using different CPGDN model without pre-training. We obtained similar accuracy for the bounding box detection (0.697 for average overlap ratio, 90.2% for detection accuracy), but got slightly worse classification accuracy (76.4% with RBM using 30 training images per class).

CHAPTER V

Augmenting CRFs with Boltzmann Machine Priors for Structured Output Prediction

5.1 Introduction

Segmentation and region labeling are core techniques for the critical mid-level vision tasks of grouping and organizing image regions into coherent parts. Segmentation refers to the grouping of image pixels into parts without applying labels to those parts, and region labeling assigns specific category names to those parts. While many segmentation and region labeling algorithms have been used in general object recognition and scene analysis, they have played a surprisingly small role in the challenging problems of face recognition.

Recently, Huang et al. [50] identified the potential role of region labeling in face recognition, noting that a variety of high-level features, such as pose, hair length, and gender can often be inferred (by people) from the labeling of a face image into hair, skin and background regions. They further showed that simple learning algorithms could be used to predict high-level features, or *attributes* [69, 106], such as pose, from the labeling.

In this work, we address the problem of labeling face regions with hair, skin, and background labels as an intermediate step in modeling face structure. In region labeling

Figure 5.1: The left image shows a "funneled" or aligned LFW image. The center image shows the superpixel version of the image which is used as a basis for the labeling. The right image shows the ground truth labeling. Red represents hair, green represents skin, and the blue represents background.

applications, the conditional random field (CRF) [70] is effective at modeling region boundaries. For example, the CRF can make a correct transition between the hair and background labels when there is a clear difference between those regions. However, when a person's hair color is similar to that of the background, the CRF may have difficulty deciding where to draw the boundary between the regions. In such cases, a global shape constraint can be used to filter out unrealistic label configurations.

It has been shown that restricted Boltzmann machines (RBMs) [117] and their extension to deeper architectures such as deep Boltzmann machines (DBMs) [111] can be used to build effective generative models of object shape. Specifically, the recently proposed shape Boltzmann machine (ShapeBM) [28] showed impressive performance in generating novel but realistic object shapes while capturing both local and global elements of shape.

Motivated by these examples, we propose the *GLOC* (GLObal and LOCal) model, a strong model for image labeling problems, that combines the best properties of the CRF (that enforces *local consistency* between adjacent nodes) and the RBM (that models *global shape prior* of the object). The model balances three goals in seeking label assignments:

- The region labels should be consistent with the underlying image features.
- The region labels should respect image boundaries.
- The complete image labeling should be consistent with shape priors defined by

the segmentation training data.

In our GLOC model, the first two objectives are achieved primarily by the CRF part, and the third objective is addressed by the RBM part. For each new image, our model uses mean-field inference to find a good balance between the CRF and RBM potentials in setting the image labels and hidden node values.

We evaluate our proposed model on a face labeling task using the Labeled Faces in the Wild (LFW) data set. As shown in Section 5.5, our model brings significant improvements in labeling accuracy over the baseline methods, such as the CRF and the conditional RBM. These gains in numerical accuracy have a significant visual impact on the resulting labeling, often fixing errors that are small but obvious to any observer. In addition, we show in Section 5.5.2 that the hidden units in the GLOC model can be interpreted as face attributes, such as whether an individual has long hair or a beard, or faces left or right. These attributes can be useful in retrieving face images with similar structure and properties.

We summarize our main contributions as follows:

- We propose the GLOC model, a strong model for face labeling tasks, that combines the CRF and the RBM to achieve both local and global consistency.
- We present efficient inference and training algorithms for our model.
- We achieve significant improvements over the state-of-the-art in face labeling accuracy on subsets of the LFW data set. Our model also produces qualitatively better labeling than the baseline CRF models.
- We demonstrate that our model learns face attributes automatically without attribute labels.

5.2 Related Work

5.2.1 Face segmentation and labeling

Several authors have built systems for segmenting hair, skin, and other face parts [137, 136, 113, 85, 142, 50]. Because of the variety of hair styles, configurations, and amount of hair, the shape of a hair segmentation can be extremely variable. In our work, we treat facial hair as part of "hair" in general, hoping to develop hidden units corresponding to beards, sideburns, mustaches, and other hair parts, which further increases the complexity of the hair segments. Furthermore, we include skin of the neck as part of the "skin" segmentation when it is visible, which is different from other labeling regimes. For example, Wang et al. [136] limit the skin region to the face and include regions covered by beards, hats, and glasses as being skin, which simplifies their labeling problem.

Yacoob and Davis [142] build a hair color model and then adopt a region growing algorithm to modify the hair region. This method has difficulty when the hair color changes significantly from one region to another, especially for dark hair, and the work was targeted at images with controlled backgrounds. Lee et al. [85] used a mixture model to learn six distinct hair styles, and other mixture models to learn color distributions for hair, skin, and background.

Huang et al. [50] used a standard CRF trained on images from LFW to build a hair, skin, and background labeler. We have implemented their model as a baseline and report the performance. Scheffler et al. [113] learn color models for hair, skin, background and clothing. They also learn a spatial prior for each label. They combine this information with a Markov random field that enforces local label consistency.

Wang et al. [136] used a compositional exemplar-based model, focusing mostly on the problem of hair segmentation. Following their earlier work, Wang et al. [137] proposed a model that regularizes the output of a segmentation using parts. In addition, their

model builds a statistical model of *where* each part is used in the image and the co-occurrence probabilities between parts. Using these co-occurrences, they build a tree-structured model over parts to constrain the final segmentations. To our knowledge, this is the best-performing algorithm for hair, skin, and background labeling to date. In Section 5.5, we report the results on two sets of labeled data showing improvements over these best previous results.

5.2.2 Object shape modeling

There are several related works on using RBMs (or their deeper extensions) for shape modeling. He et al. [40] proposed multiscale CRFs to model both local and global label features using RBMs. Specifically, they used multiple RBMs at different scales to model the regional or global label fields (layers) separately, and combined those conditional distributions multiplicatively. Recent work by Eslami et al. [28] introduced the Shape Boltzmann machine (ShapeBM), a two-layer DBM with local connectivity in the first layer for local consistency and generalization (by weight sharing), and full connectivity in the second layer for modeling global shapes, as a strong model for object shapes. Subsequently, Eslami and Williams [27] proposed a generative model by combining the ShapeBM with an appearance model for parts-based object segmentation. Our model is similar at a high-level to these models in that we use RBMs for object shape modeling to solve image labeling problems. However, there are significant technical differences that distinguish our model from others. First, our model has an edge potential that enforces local consistency between adjacent superpixel labels. Second, we define our model on the superpixel graph using a virtual pooling technique, which is computationally much more efficient. Third, our model is discriminative and can use richer image features than [27] which used a simple pixel-level appearance model (based on RGB pixel values). Finally, we propose a model combined with an RBM to act as a shape prior, which makes the training much easier while showing significant improvement over the baseline models in

face labeling tasks. See 5.4.4 for more discussions.

5.3 Preliminaries

In this section, we briefly describe the CRF and RBM, followed by our proposed GLOC model. We present the models in the context of multi-class labeling.

Notation An image I is pre-segmented into $S^{(I)}$ superpixels, where $S^{(I)}$ can vary over different images. We denote $\mathcal{V}^{(I)} = \{1, \cdots, S^{(I)}\}$ as a set of superpixel nodes, and $\mathcal{E}^{(I)}$ as a set of edges connecting adjacent superpixels. We denote $\mathcal{X}^{(I)} = \{\mathcal{X}_{\mathcal{V}}^{(I)}, \mathcal{X}_{\mathcal{E}}^{(I)}\}$, where $\mathcal{X}_{\mathcal{V}}^{(I)}$ is a set of node features $\{\mathbf{x}_s^{\text{node}} \in \mathbb{R}^{D_n}, s \in \mathcal{V}\}$ and $\mathcal{X}_{\mathcal{E}}^{(I)}$ is a set of edge features $\{\mathbf{x}_{ij}^{\text{edge}} \in \mathbb{R}^{D_e}, (i,j) \in \mathcal{E}\}$. The set of label nodes are defined as $\mathcal{Y}^{(I)} = \{\mathbf{y}_s \in \{0,1\}^L, s \in \mathcal{V} : \sum_{l=1}^{L} y_{sl} = 1\}$. Here, D_n and D_e denote the dimensions of the node and edge features, respectively, and L denotes the number of categories for the labeling task. We frequently omit the superscripts "I", "node", or "edge" for clarity, but the meaning should be clear from the context.

5.3.1 Conditional Random Fields

The conditional random field [70] is a powerful model for structured output prediction (such as sequence prediction, text parsing, and image segmentation), and has been widely used in computer vision [41, 4, 13, 40]. The conditional distribution and the energy function can be defined as follows:

$$P_{\text{crf}}(\mathcal{Y}|\mathcal{X}) \propto \exp(-E_{\text{crf}}(\mathcal{Y}, \mathcal{X})), \tag{5.1}$$

$$E_{\text{crf}}(\mathcal{Y}, \mathcal{X}) = \underbrace{-\sum_{s \in \mathcal{V}} \sum_{l=1}^{L} \sum_{d=1}^{D_n} y_{sl} \Gamma_{ld} x_{sd}}_{=E_{\text{node}}(\mathcal{Y}, \mathcal{X}_{\mathcal{V}})} \underbrace{- \sum_{(i,j) \in \mathcal{E}} \sum_{l,l'=1}^{L} \sum_{e=1}^{D_e} y_{il} y_{jl'} \Psi_{ll'e} x_{ije}}_{E_{\text{edge}}(\mathcal{Y}, \mathcal{X}_{\mathcal{E}})}, \tag{5.2}$$

where $\Psi \in \mathbb{R}^{L \times L \times D_e}$ is a 3D tensor for the edge weights, and $\Gamma \in \mathbb{R}^{L \times D_n}$ are the node weights. The model parameters $\{\Gamma, \Psi\}$ are trained to maximize the conditional log-likelihood of the training data $\{\mathcal{Y}^{(m)}, \mathcal{X}^{(m)}\}_{m=1}^{M}$,

$$\max_{\Gamma, \Psi} \sum_{m=1}^{M} \log P_{\text{crf}}(\mathcal{Y}^{(m)} | \mathcal{X}^{(m)}).$$

We can use loopy belief propagation (LBP) [95] or mean-field approximation [112] for inference in conjunction with standard optimization methods such as LBFGS.[1]

5.3.2 Restricted Boltzmann machines with multinomial visible unit

The restricted Boltzmann machine [117] is a bipartite, undirected graphical model composed of visible and hidden layers. In our context, we assume R^2 multinomial visible units $\mathbf{y}_r \in \{0, 1\}^L$ and K binary hidden units $h_k \in \{0, 1\}$. The joint distribution can be defined as follows:

$$P_{\text{rbm}}(\mathcal{Y}, \mathbf{h}) \propto \exp(-E_{\text{rbm}}(\mathcal{Y}, \mathbf{h})), \tag{5.3}$$

$$E_{\text{rbm}}(\mathcal{Y}, \mathbf{h}) = -\sum_{r=1}^{R^2} \sum_{l=1}^{L} \sum_{k=1}^{K} y_{rl} W_{rlk} h_k - \sum_{k=1}^{K} b_k h_k - \sum_{r=1}^{R^2} \sum_{l=1}^{L} c_{rl} y_{rl}, \tag{5.4}$$

where $\mathbf{W} \in \mathbb{R}^{R^2 \times L \times K}$ is a 3D tensor specifying the connection weights between visible and hidden units, b_k is the hidden bias, and c_{rl} is the visible bias. The parameters $\Theta = \{\mathbf{W}, \mathbf{b}, \mathbf{C}\}$ are trained to maximize the log-likelihood of the training data $\{\mathcal{Y}^{(m)}\}_{m=1}^{M}$,

$$\max_{\Theta} \sum_{m=1}^{M} \log \left(\sum_{\mathbf{h}} P_{\text{rbm}}(\mathcal{Y}^{(m)}, \mathbf{h}) \right).$$

We train the model parameters using stochastic gradient descent. Although the exact gradient is intractable to compute, we can approximate it using CD [44].

[1] We used LBFGS in minFunc by Mark Schmidt: `http://www.di.ens.fr/~mschmidt/Software/minFunc.html`

5.4 The Proposed Model

To build a strong model for image labeling, both local consistency (adjacent nodes are likely to have similar labels) and global consistency (the overall shape of the object should look realistic) are desirable. On one hand, the CRF is powerful in modeling local consistency via edge potentials. On the other hand, the RBM is good at capturing global shape structure through the hidden units. We combine these two ideas in the GLOC model, which incorporates both local consistency (via CRF-like potentials) and global consistency (via RBM-like potentials). Specifically, we describe the conditional likelihood of labels set $\mathcal{Y}$ given the superpixel features $\mathcal{X}$ as follows:

$$P_{\text{gloc}}(\mathcal{Y}|\mathcal{X}) \propto \sum_{\mathbf{h}} \exp\left(-E_{\text{gloc}}(\mathcal{Y}, \mathcal{X}, \mathbf{h})\right), \tag{5.5}$$

$$E_{\text{gloc}}\left(\mathcal{Y}, \mathcal{X}, \mathbf{h}\right) = E_{\text{crf}}\left(\mathcal{Y}, \mathcal{X}\right) + E_{\text{rbm}}\left(\mathcal{Y}, \mathbf{h}\right). \tag{5.6}$$

As described above, the energy function is written as a combination of CRF and RBM energy functions. However, due to the varying number of superpixels for different images, the RBM energy function in Equation (5.4) requires nontrivial modifications. In other words, we cannot simply connect label (visible) nodes defined over superpixels to hidden nodes as in Equation (5.4) because 1) the RBM is defined on a fixed number of visible nodes and 2) the number of superpixels and their underlying graph structure can vary across images.

5.4.1 Virtual pooling layer

To resolve this issue, we introduce a *virtual, fixed-sized* pooling layer between the label and the hidden layers, where we map each superpixel label node into the *virtual* visible nodes of the $R \times R$ square grid. This is shown in Figure 5.2, where the top two layers can be thought of as an RBM with the visible nodes $\bar{\mathbf{y}}_r$ representing a surrogate (i.e., pooling) for the labels $\mathbf{y}_s$ that overlap with the grid bin r. Specifically, we define

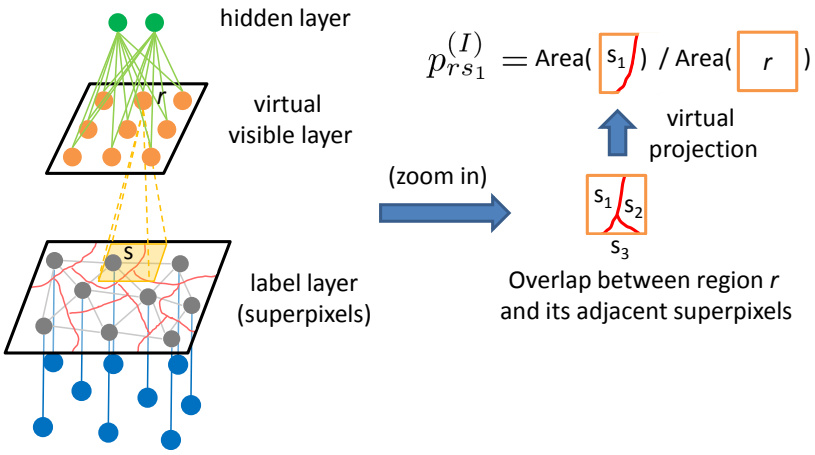


Figure 5.2: The GLOC model. The top two layers can be thought of as an RBM with the (virtual) visible nodes $\bar{\mathbf{y}}_r$ and the hidden nodes. To define the RBM over a fixed-size visible node grid, we use an image-specific "projection matrix" $\{p^{(I)}_{rs}\}$ that transfers (top-down and bottom-up) information between the label layer and the virtual grid of the RBM's visible layer. See text for details.

the energy function between the label nodes and the hidden nodes for an image I as follows:

$$E_{\text{rbm}}(\mathcal{Y}, \mathbf{h}) = -\sum_{r=1}^{R^2}\sum_{l=1}^{L}\sum_{k=1}^{K} \bar{y}_{rl} W_{rlk} h_k - \sum_{k=1}^{K} b_k h_k - \sum_{r=1}^{R^2}\sum_{l=1}^{L} c_{rl}\bar{y}_{rl}. \tag{5.7}$$

Here, the virtual visible nodes $\bar{y}_{rl} = \sum_{s=1}^{S} p_{rs} y_{sl}$ are deterministically mapped from the superpixel label nodes using the projection matrix $\{p_{rs}\}$ that determines the contribution of label nodes to each node of the grid. The projection matrix is defined as follows:[2]

$$p_{rs} = \frac{Area(Region(s) \cap Region(r))}{Area(Region(r))},$$

where $Region(s)$ and $Region(r)$ denote sets of pixels corresponding to superpixel s and grid r, respectively. Due to the deterministic connection, the pooling layer is actually a *virtual* layer that only exists to map between the superpixel nodes and the hidden nodes. We can also view our GLOC model as having a set of grid-structured nodes that performs average pooling over the adjacent superpixel nodes.

[2]The projection matrix $\{p_{rs}\}$ is a sparse, non-negative matrix of dimension $R^2 \times S$. Note that the projection matrix is specific to each image since it depends on the structure of the superpixel graph.

5.4.2 Spatial CRF

As an additional baseline, we describe a modification to the CRF presented in Section 5.3.1. In some cases, even after conditioning on class, feature likelihoods may depend on position. For example, knowing that hair rests on the shoulders makes it less likely to be gray. This intuition is behind our Spatial CRF model.

Specifically, when the object in the image is aligned, we can learn a spatially dependent set of weights that are specific to a cell in an $N \times N$ grid. (Note that this grid can be a different size than the $R \times R$ grid used by the RBM.) We learn a separate set of node weights for each cell in a grid, but the edge weights are kept globally stationary.

Using a similar projection technique to that described in Section 5.4.1, we define the node energy function as

$$E_{\text{node}}(\mathcal{Y}, \mathcal{X}_{\mathcal{V}}) = -\sum_{s \in \mathcal{V}} \sum_{l=1}^{L} y_{sl} \sum_{n=1}^{N^2} p_{sn} \sum_{d=1}^{D_n} \Gamma_{ndl} x_{sd}, \tag{5.8}$$

where $\Gamma \in \mathbb{R}^{N^2 \times D \times L}$ is a 3D tensor specifying the connection weights between the superpixel node features and labels at each spatial location. In this energy function, we define a different projection matrix $\{p_{sn}\}$ which specifies the mapping from the $N \times N$ virtual grid to superpixel label nodes.[3]

5.4.3 Inference and learning

Inference. Since the joint inference of superpixel labels and the hidden nodes is intractable, we resort to the mean-field approximation. Specifically, we find a fully factorized distribution $Q(\mathcal{Y}, \mathbf{h}; \mu, \gamma) = \prod_{s \in \mathcal{V}} Q(\mathbf{y}_s) \prod_{k=1}^{K} Q(h_k)$, with $Q(\mathbf{y}_s = l) \triangleq \mu_{sl}$ and $Q(h_k = 1) \triangleq \gamma_k$, that minimizes $\text{KL}\left(Q(\mathcal{Y}, \mathbf{h}; \mu, \gamma) \| P(\mathcal{Y}, \mathbf{h} | \mathcal{X})\right)$. We describe the mean-field inference steps in Algorithm 2 and its derivation in Appendix C.

[3]Note that the projection matrices used in the RBM and spatial CRF are different in that $\{p_{rs}\}$ used in the RBM describes a projection from superpixel to grid ($\sum_{s=1}^{S} p_{rs} = 1$), whereas $\{p_{sn}\}$ used in the spatial CRF describes a mapping from a grid to superpixel ($\sum_{n=1}^{N^2} p_{sn} = 1$).

Algorithm 2 Mean-Field Inference

1: Initialize $\boldsymbol{\mu}^{(0)}$ and $\boldsymbol{\gamma}^{(0)}$ as follows:

$$\mu_{sl}^{(0)} = \frac{\exp\left(f_{sl}^{\text{node}}\right)}{\sum_{l'} \exp\left(f_{sl'}^{\text{node}}\right)}$$
$$\gamma_k^{(0)} = \sigma\left(\sum_{r,l}\left(\sum_s p_{rs}\mu_{sl}^{(0)}\right) W_{rlk} + b_k\right)$$

where

$$f_{sl}^{\text{node}}\left(\mathcal{X}_{\mathcal{V}}, \{p_{sn}\}, \Gamma\right) = \sum_{n,d} p_{sn} x_{sd} \Gamma_{ndl}$$

2: **for** t=0:*maxiter* (or until convergence) **do**

3: update $\boldsymbol{\mu}^{(t+1)}$ as follows: $\mu_{sl}^{(t+1)} =$

$$\frac{\exp\left(f_{sl}^{\text{node}} + f_{sl}^{\text{edge}}\left(\boldsymbol{\mu}^{(t)}\right) + f_{sl}^{\text{rbm}}\left(\boldsymbol{\gamma}^{(t)}\right)\right)}{\sum_{l'} \exp\left(f_{sl'}^{\text{node}} + f_{sl'}^{\text{edge}}\left(\boldsymbol{\mu}^{(t)}\right) + f_{sl'}^{\text{rbm}}\left(\boldsymbol{\gamma}^{(t)}\right)\right)}$$

where

$$f_{sl}^{\text{edge}}\left(\boldsymbol{\mu}; \mathcal{X}_{\mathcal{E}}, \mathcal{E}, \Psi\right) = \sum_{j:(s,j)\in\mathcal{E}} \sum_{l',e} \mu_{jl'} \Psi_{ll'e} x_{sje}$$
$$f_{sl}^{\text{rbm}}\left(\boldsymbol{\gamma}; \{p_{rs}\}, \mathbf{W}, \mathbf{C}\right) = \sum_{r,k} p_{rs}\left(W_{rlk}\gamma_k + c_{rl}\right)$$

4: update $\boldsymbol{\gamma}^{(t+1)}$ as follows:

$$\gamma_k^{(t+1)} = \sigma\left(\sum_{r,l}\left(\sum_s p_{rs}\mu_{sl}^{(t+1)}\right) W_{rlk} + b_k\right)$$

5: **end for**

Learning. In principle, we can train the model parameters $\{\mathbf{W}, \mathbf{b}, \mathbf{C}, \Gamma, \Psi\}$ simultaneously to maximize the conditional log-likelihood. In practice, however, it is beneficial to provide a proper initialization (or *pretrain*) to those parameters. We provide an overview of the training procedure in Algorithm 3.

First, we adapted the pretraining method of deep Boltzmann machines (DBM) [111] to train the conditional RBM (CRBM).[4] Specifically, we pretrain the model parameters $\{\mathbf{W}, \mathbf{b}, \mathbf{C}\}$ of the CRBM as if it is a top layer of the DBM to avoid double-counting when combined with the edge potential in the GLOC model. Second, the CRBM and the GLOC models can be trained to either maximize the conditional log-likelihood using contrastive divergence (CD) or *minimize generalized perceptron loss* [79] using CD-PercLoss [94]. In fact, Mnih et al. [94] suggested that CD-PercLoss would be a better choice for structured output prediction problems since it directly penalizes the model for wrong predictions during training. We empirically observed that CD-PercLoss performed slightly better than CD.

Algorithm 3 Training GLOC model

1: Pretrain $\{\Gamma, \Psi\}$ to maximize the conditional log-likelihood of the *spatial CRF* model (See Equations (5.1), (5.2), and (5.8)).
2: Pretrain $\Theta = \{\mathbf{W}, \mathbf{b}, \mathbf{C}\}$ to maximize the conditional log-likelihood of the *conditional RBM*, which is defined as:

$$\max_{\Theta} \log \sum_{\mathbf{h}} P_{\text{crbm}}(\mathcal{Y}, \mathbf{h} | \mathcal{X}_{\mathcal{V}})$$
$$P_{\text{crbm}}(\mathcal{Y}, \mathbf{h} | \mathcal{X}_{\mathcal{V}}) \propto \exp\left(-E_{\text{node}}(\mathcal{Y}, \mathcal{X}_{\mathcal{V}}; \Gamma) - E_{\text{rbm}}(\mathcal{Y}, \mathbf{h}; \Theta)\right)$$

3: Train $\{\mathbf{W}, \mathbf{b}, \mathbf{C}, \Gamma, \Psi\}$ to maximize the conditional log-likelihood of the *GLOC* model (See Equation (5.5)).

[4] Note that our CRBM is different from the one defined in [94] in that 1) our model has no connection between the conditioning nodes $\mathcal{X}$ and the hidden nodes, and 2) our model uses a projection (e.g., virtual pooling) matrix to deal with the varying number of label nodes over the images.

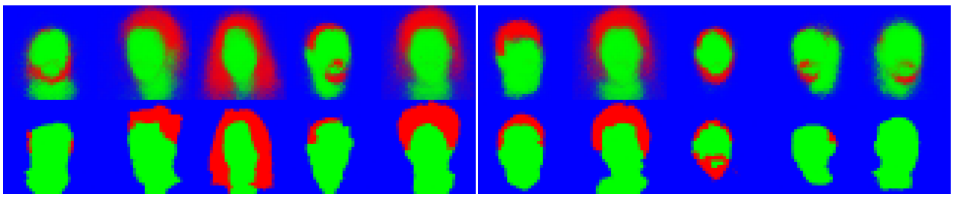


Figure 5.3: Generated samples from the RBM (first row) and the closest matching examples in the training set (second row). The RBM can generate novel, realistic examples by combining hair, beard and mustache shapes along with diverse face shapes.

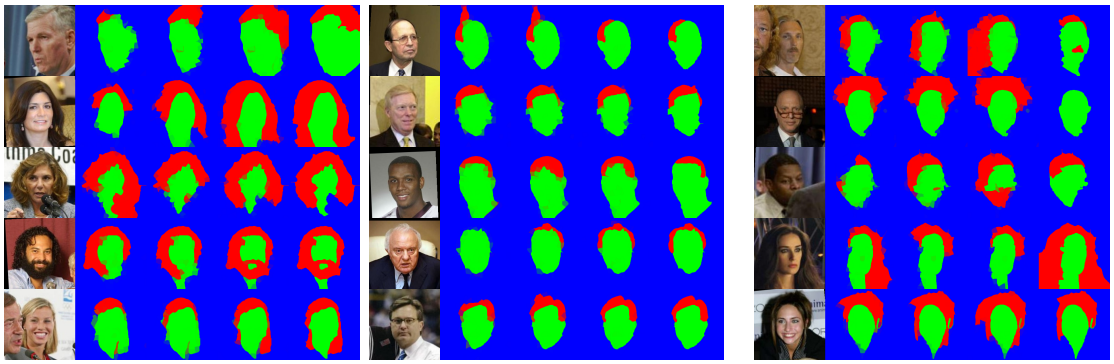
5.4.4 Discussion

In many cases, it is advantageous to learn generative models with deep architectures. In particular, Eslami et al. [28] suggest that the ShapeBM, a special instance of the DBM, can be a better generative model than the RBM when they are only given several hundred training examples. However, when given sufficient training data (e.g., a few thousand), we found that the RBM can still learn a global shape prior with good generalization performance. In Figure 5.3, we show both generated samples from an RBM and their closest training examples.[5] The generated samples are diverse and are clearly different from their most similar examples in the training set. This suggests that our model is learning an interesting decomposition of the shape distributions for faces. Furthermore, RBMs are easier to train than DBMs in general, which motivates the use of RBMs in our model. In principle, however, we can also use such deep architectures in our GLOC model as a rich global shape prior without much modification to inference and learning.

5.5 Experiments

We evaluated our proposed model on a task to label face images from the LFW data set [49] as hair, skin, and background. We used the "funneled" version of LFW, in which images have been coarsely aligned using a congealing-style joint alignment

[5] We compute the L_2 distance between the generated samples and the examples in the training set.



(a) Successful examples (b) Failure examples

Figure 5.4: Sample segmentation results on images from the LFW data set. The images contain extremely challenging scenarios such as multiple distractor faces, occlusions, strong highlights, and pose variation. The left of Figure 5.4(a) shows images in which the GLOC model made relatively large improvements to the baseline. The right of Figure 5.4(a) shows more subtle changes made by our model. The results in Figure 5.4(b) show typical failure cases. The columns correspond to 1) original image which has been aligned to a canonical position using funneling [48], 2) CRF, 3) spatial CRF, 4) GLOC and 5) ground truth labeling. Note that the CRBM model results are not shown here.

approach [48]. Although better automatic alignments of these images exist, such as the LFW-a data set [141], it does not contain color information, which is important for our application.

The LFW website provides the segmentation of each image into superpixels, which are small, relatively uniform pixel groupings.[6] We provide ground truth for a set of 2927 LFW images by labeling each superpixel as either hair, skin, or background [1]. While some superpixels may contain pixels from more than one region, most superpixels are generally "pure" hair, skin, or background.

There are several reasons why we used superpixel labeling instead of pixel labeling for this problem. First, the superpixel representation is computationally much more efficient. The number of nodes would be too large for pixel labeling since the LFW images are of size 250×250. However, each image can be segmented into 200-250 superpixels, resulting in the same number of nodes in the CRF, and this allowed us

[6] Available at `http://vis-www.cs.umass.edu/lfw/lfw_funneled_superpixels_fine.tgz`.

to do tractable inference using LBP or mean-field. In addition, superpixels can help smooth features such as color. For example, if the superpixel is mostly black but contains a few blue pixels, the blue pixels will be smoothed out from the feature vector, which can simplify inference.

We adopted the same set of features as in Huang et al. [50]. For each superpixel we used the following node features:

- Color: Normalized histogram over 64 bins generated by running K-means over pixels in LAB space.
- Texture: Normalized histogram over 64 textons generated according to [89].
- Position: Normalized histogram of the proportion of a superpixel that falls within each of the 8×8 grid elements on the image.[7]

The following edge features were computed between adjacent superpixels:

- Sum of PB [92] values along the border.
- Euclidean distance between mean color histograms.
- Chi-squared distance between texture histograms as computed in [50].

We evaluated the labeling performance of four different models: a standard CRF, the spatial CRF, the CRBM, and our GLOC model. We provide the summary results in Table 5.1. We divided the labeled examples into $2,000$ for training and 927 for testing, and performed 5-fold cross-validation by randomly splitting the training data into 5. We trained our model using batch gradient descent and selected the model hyperparameters that performed best on the validation set. After cross-validation, we set $K = 400$, $R = 24$, and $N = 16$, and the model was evaluated on the test set. On a multicore AMD Opteron, average inference time per example was 0.254 seconds for the GLOC model and 0.063 seconds for the spatial CRF.

As shown in Table 5.1, the GLOC model substantially improves the superpixel labeling accuracy over the baseline CRF model as well as the spatial CRF and CRBM

[7]Note that the position feature is only used in the CRF.

Method	Accuracy	Error Reduction
LR	90.91% ± 0.011	-22.94%
Spatial LR	92.11% ± 0.021	-6.71%
CRF (baseline)	92.61% ± 0.310	–
Spatial CRF	93.88% ± 0.042	17.23%
CRBM	94.06% ± 0.026	19.58%
GLOC	94.95% ± 0.026	31.72%

Table 5.1: Labeling accuracies for each model. We report the mean of superpixel-wise labeling accuracy and corresponding 95% confidence interval in the second column, and the error reduction over the CRF on test set in the third column.

models. While absolute accuracy improvements (necessarily) become small as accuracy approaches 95%, the reduction in errors are substantial.

Furthermore, there are significant qualitative differences in many cases, as we illustrate in Figure 5.4(a). The samples on the left show significant improvement over the spatial CRF, and the ones on the right show more subtle changes made by the GLOC model. Here, we represent the confidence of the guess (posterior) by color intensity. The confident guess appears as a strong red, green, or blue color, and a less confident guess appears as a lighter mixture of colors. As we can see, the global shape prior of the GLOC model helps "clean up" the guess made by the spatial CRF in many cases, resulting in a more confident prediction.

In many cases, the RBM prior encourages a more realistic segmentation by either "filling in" or removing parts of the hair or face shape. For example, the woman in the second row on the left set recovers the left side of her hair and gets a more recognizable hair shape under our model. Also, the man in the first row on the right set gets a more realistic looking hair shape by removing the small (incorrect) hair shape on top of his head. This effect may be due to the top-down global prior in our GLOC model, whereas simpler models such as the spatial CRF do not have this information. In addition, there were cases (such as the woman in the fifth row of the left set) where an additional face in close proximity to the centered face may confuse the model. In this case, the CRF

and spatial CRF models make mistakes, but since the GLOC model has a strong shape model, it was able to find a more recognizable segmentation of the foreground face.

On the other hand, the GLOC model sometimes makes errors. We show typical failure examples in Figure 5.4(b). As we can see, the model made significant errors in their hair regions. Specifically, in the first row, the hair of a nearby face is similar in color to the hair of the foreground face as well as the background, and our model incorrectly guesses more hair by emphasizing the hair shape prior, perhaps too strongly. In addition, there are cases in which occlusions cause problems, such as the third row. However, we point out that the occlusions are frequently handled correctly by our model (e.g., the microphone in the third row of the left set in Figure 5.4(a)).

5.5.1 Comparison to prior work

We also evaluated our model on the data set used in [137]. This data set contains $1,046$ LFW (unfunneled) images whose pixels are manually labeled into 4 regions (Hair, Skin, Background, and Clothing). Following their evaluation setup, we randomly split the data in half and used one half for training and the rest for testing. We repeated this procedure five times, and report the average pixel accuracy as a final result.

We first generated the superpixels and features for each image, then ran our GLOC model to get label guesses for each superpixel, and finally mapped back to pixels for evaluation (it was necessary to map to pixels at the end because the ground truth is provided in pixels). We noticed that even with a perfect superpixel labeling, this mapping already incurs approximately 3% labeling error. However, our approach was sufficient to obtain a good pixel-wise accuracy of 90.7% (91.7% superpixel-wise accuracy), which improves by 0.7% upon their best reported result of 90.0%. The ground truth for a superpixel is a normalized histogram of the pixel labels in the superpixel.

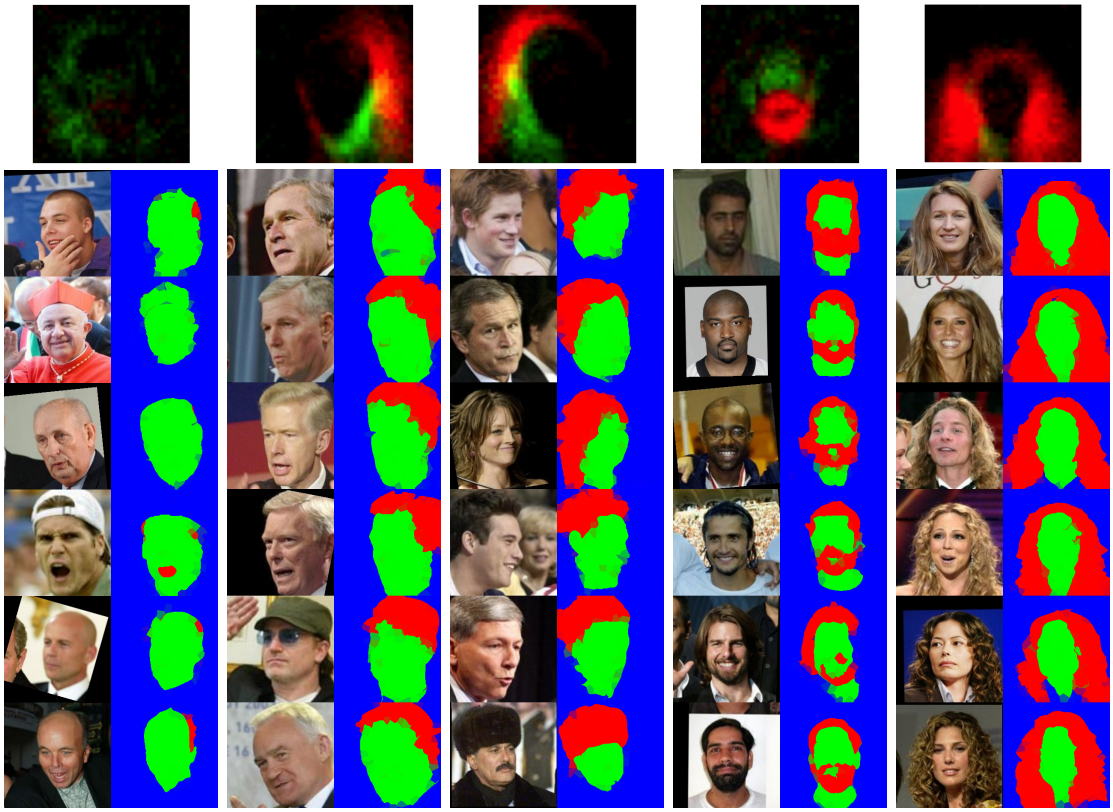


Figure 5.5: The filter visualization in each column show that the GLOC model learns latent structure or visual attribute automatically from the data that can be interpreted as (from left to right) "no hair showing", "looking left", "looking right", "beard/occluded chin", "big hair". In each column, we retrieve the images from LFW (except images used in training and validation) with the highest activations for each of 5 hidden units, and provide their segmentation results. Although the retrieved matches are not perfect, they clearly have semantic, high-level content.

5.5.2 Attributes and retrieval

While the labeling accuracy (as shown in Section 5.5) is a direct way of measuring progress, we have an additional goal in our work: to build models that capture the natural statistical structure in faces. It is not an accident that human languages have words for beards, baldness, and other salient high-level attributes of human face appearance. These attributes represent coherent and repeated structure across the faces we see everyday. Furthermore, these attributes are powerful cues for recognition, as demonstrated by Kumar et al. [69].

One of the most exciting aspects of RBMs and their deeper extensions are that

these models can learn latent structure automatically. Recent work has shown that unsupervised learning models can learn meaningful structure without being explicitly trained to do so (e.g., [75, 51, 52]).

In our experiments, we ran our GLOC model on all LFW images other than those used in training and validation, and sorted them based on each hidden unit activation. Each of the five columns in Figure 5.5 shows a set of retrieved images and their guessed labelings for a particular hidden unit. In many cases, the retrieved results for the hidden units form meaningful clusters. These units seem highly correlated with "lack of hair", "looking left", "looking right", "beard or occluded chin", and "big hair". Thus, the learned hidden units may be useful as attribute representations for faces.

5.6 Conclusion

Face segmentation and labeling is challenging due to the diversity of hair styles, head poses, clothing, occlusions, and other phenomena that are difficult to model, especially in a database like LFW. Our GLOC model combines the CRF and the RBM to model both local and global structure in face segmentations. Our model has consistently reduced the error in face labeling over previous models which lack global shape priors. In addition, we have shown that the hidden units in our model can be interpreted as face attributes, which were learned without any attribute-level supervision.

CHAPTER VI

Improved Multimodal Deep Learning with Variation of Information

6.1 Introduction

Different types of multiple data modalities can be used to describe the same event. For example, images, which are often represented with pixels or image descriptors, can also be described with accompanying text (e.g., user tags or subtitles) or audio data (e.g., human voice or natural sound). There have been several applications of multimodal learning from multiple domains such as emotion and speech recognition with audio-visual data [61, 99, 53], robotics applications with visual and depth data [71, 86, 134, 105], and medical applications with visual and temporal data [115]. For each application, data from multiple sources are *semantically* correlated, and sometimes provide complementary information about each other. To facilitate information exchange, it is important to capture a high-level association between data modalities with a compact set of latent variables. However, learning associations between multiple *heterogeneous* data distributions is a challenging problem.

A naive approach is to concatenate the data descriptors from different input sources to construct a single high-dimensional feature vector and use it to solve a unimodal representation learning problem. However, the correlation between features in each

data modality is much stronger than that between data modalities. As a result, the learning algorithms are easily tempted to learn dominant patterns in each data modality separately while giving up learning patterns that occur simultaneously in multiple data modalities, as suggested by [99]. To resolve this issue, deep learning methods, such as deep autoencoders [45] or deep Boltzmann machines (DBM) [111], have been adapted [99, 121], where the common strategy is to learn joint representations that are shared across multiple modalities at the higher layer of the deep network, after learning layers of modality-specific networks. The rationale is that the learned features may have less within-modality correlation than raw features, and this makes it easier to capture patterns across data modalities. This has shown promise, but there still remains the challenging question of how to learn associations between multiple heterogeneous data modalities so that we can effectively deal with missing data modalities at testing time.

One necessary condition for a good generative model of multimodal data is the ability to predict or reason about missing data modalities given partial observation. To this end, we propose a novel multimodal representation learning framework that explicitly aims at this goal. The key idea is to minimize the information distance between data modalities through the shared latent representations. More concretely, we train the model to minimize the *variation of information* (VI), an information theoretic measure that computes the distance between random variables, i.e., multiple data modalities. Note that this is in contrast to previous approaches on multimodal deep learning, which are based on maximum (joint) likelihood (ML) learning [99, 121]. We explain as to how our method could be more effective in learning the joint representation of multimodal data than ML learning, and show theoretical insights why the proposed learning objective is sufficient to estimate the data-generating joint distribution of multimodal data. We apply the proposed framework to multimodal restricted Boltzmann machine (MRBM) and propose two learning algorithms, based on contrastive divergence [94] and multi-prediction training [36]. Finally, we extend to multimodal deep recurrent neural

network (MDRNN) for unsupervised finetuning of whole network. In experiments, we demonstrate the state-of-the-art visual recognition performance on MIR-Flickr database and PASCAL VOC2007 database with and without text observations at testing time.

6.2 Multimodal Learning with Variation of Information

In this section, we propose a novel training objective based on the VI. We make a comparison to the ML objective, a typical learning objective for training generative models of multimodal data, to give an insight as to how our proposed method can be better for multimodal data. Finally, we establish a theorem showing that the proposed learning objective is sufficient to obtain a good generative model that fully recovers the joint data-generating distribution of multimodal data.

Notation. We use uppercase letters X, Y to denote random variables, lowercase letters x, y for realizations. Let $P_{\mathcal{D}}$ be the data-generating distribution and P_θ the model distribution parametrized by θ. For presentation clarity, we slightly abuse the notation for Q to denote conditional $(Q(x|y), Q(y|x))$, marginal $(Q(x), Q(y))$, as well as joint distributions $(Q(x, y))$. The type of distribution of Q should be clear from the context.

6.2.1 Minimum variation of information learning

Motivated by the necessary condition for good generative models to reason about the missing data modality, it seems natural to learn to maximize the amount of information that one data modality has about the others. We quantify such an amount of information between data modalities using variation of information. The VI is an information theoretic measure that computes the information distance between two random

variables (e.g., data modalities), and is written as follows:[1]

$$\mathrm{VI}_Q(X,Y) = -\mathbb{E}_{Q(X,Y)}\Big[\log Q(X|Y) + \log Q(Y|X)\Big] \tag{6.1}$$

where $Q(X,Y) = P_\theta(X,Y)$ is any joint distribution on random variables (X,Y) parametrized by θ. Informally, VI is small when the conditional likelihoods $Q(X|Y)$ and $Q(Y|X)$ are "peaked", meaning that X has low entropy conditioned on Y and vice versa. Following the intuition, we define new multimodal learning criteria, a *minimum variation of information* (MinVI) learning, as follows:

$$\textbf{\textit{MinVI}:}\quad \min_\theta \mathcal{L}^{\mathrm{VI}}(\theta),\quad \mathcal{L}^{\mathrm{VI}}(\theta) = -\mathbb{E}_{P_{\mathcal{D}}(X,Y)}\Big[\log P_\theta(X|Y) + \log P_\theta(Y|X)\Big] \tag{6.2}$$

Note the difference that we take the expectation over $P_{\mathcal{D}}$ in $\mathcal{L}^{\mathrm{VI}}(\theta)$. Furthermore, we observe that the MinVI objective can be decomposed into a sum of two negative conditional LLs. This indeed aligns well with our initial motivation of reasoning about missing data modality. In the following, we provide more insight into our MinVI objective in relation to the ML objective, which is a standard learning objective in generative models.

6.2.2 Relation to maximum likelihood learning

The ML objective function can be written as a minimization of the negative LL (NLL) as follows:

$$\textbf{\textit{ML}:}\quad \min_\theta \mathcal{L}^{\mathrm{NLL}}(\theta),\quad \mathcal{L}^{\mathrm{NLL}}(\theta) = -\mathbb{E}_{P_{\mathcal{D}}(X,Y)}\Big[\log P_\theta(X,Y)\Big], \tag{6.3}$$

[1] In practice, we use finite samples of the training data and use a regularizer (e.g., l_2 regularizer) to avoid overfitting to the finite sample distribution.

and we can show that the NLL objective function is reformulated as follows:

$$2\mathcal{L}^{\mathrm{NLL}}(\theta) = \underbrace{KL\left(P_{\mathcal{D}}(X)\|P_\theta(X)\right) + KL\left(P_{\mathcal{D}}(Y)\|P_\theta(Y)\right)}_{(a)} +$$

$$\underbrace{\mathbb{E}_{P_{\mathcal{D}}(X)}\Big[KL\left(P_{\mathcal{D}}(Y|X)\|P_\theta(Y|X)\right)\Big] + \mathbb{E}_{P_{\mathcal{D}}(Y)}\Big[KL\left(P_{\mathcal{D}}(X|Y)\|P_\theta(X|Y)\right)\Big]}_{(b)} + C, \quad (6.4)$$

where C is a constant which is irrelevant to θ. Note that (b) is equivalent to $\mathcal{L}^{\mathrm{VI}}(\theta)$ in Equation (6.2) up to a constant. We provide a full derivation of Equation (6.4) in Appendix D.1.

Ignoring the constant, the NLL objective has four KL divergence terms. Since KL divergence is non-negative and is zero only when two distributions match, the ML learning in Equation (6.3) can be viewed as a distribution matching problem involving (a) marginal likelihoods and (b) conditional likelihoods. Here, we argue that (a) is more difficult to optimize than (b) because there are often too many modes in the marginal distribution. Compared to the marginal distribution, the number of modes can be dramatically reduced in the conditional distribution since the conditioning variables may restrict the support of random variable effectively. Therefore, (a) may become a dominant factor to be minimized during the optimization process and as a trade-off, (b) will be easily compromised, which makes it difficult to learn a good association between data modalities. On the other hand, the MinVI objective focuses on modeling the conditional distributions (Equation (6.4)), which is arguably easier to optimize. Indeed, similar argument has been made for generalized denoising autoencoders (DAEs) [7] and generative stochastic networks (GSNs) [10], which focus on learning the transition operators (e.g., $P_\theta(X|\tilde{X})$, where $\tilde{X}$ is a corrupted version of data X, or $P_\theta(X|H)$, where H can be arbitrary latent variables) to bypass an intractable problem of learning density model $P_\theta(X)$.

6.2.3 Theoretical results

Bengio et al. [7, 10] proved that learning transition operators of DAEs or GSNs is sufficient to learn a good generative model that estimates a data-generating distribution. Under similar assumptions, we establish a theoretical result that we can obtain a good density estimator for joint distribution of multimodal data by learning the transition operators derived from the conditional distributions of one data modality given the other. In the multimodal learning framework, we define the transition operators $T_n^{\mathcal{X}}$ and $T_n^{\mathcal{Y}}$ for Markov chains of data modalities X and Y, respectively. Specifically, $T_n^{\mathcal{X}}\left(x[t]|x[t-1]\right) = \sum_{y\in\mathcal{Y}} P_{\theta_n}\left(x[t]|y\right) P_{\theta_n}\left(y|x[t-1]\right)$, where $P_{\theta_n}\left(X|Y\right)$ and $P_{\theta_n}\left(Y|X\right)$ are model conditional distributions after learning from the training data of size n. $T_n^{\mathcal{Y}}$ is defined in a similar way. Note that we do not require that the model conditionals are derived from an analytically defined joint distribution. Now, we formalize the theorem as follows:

Theorem VI.1. *For finite state space $\mathcal{X}, \mathcal{Y}$, if, $\forall x \in \mathcal{X}, \forall y \in \mathcal{Y}$, $P_{\theta_n}(\cdot|y)$ and $P_{\theta_n}(\cdot|x)$ converges in probability to $P_{\mathcal{D}}(\cdot|y)$ and $P_{\mathcal{D}}(\cdot|x)$, respectively, and $T_n^{\mathcal{X}}$ and $T_n^{\mathcal{Y}}$ are ergodic Markov chains, then, as the number of examples $n \to \infty$, the asymptotic distribution $\pi_n(X)$ and $\pi_n(Y)$ converge to data-generating marginal distributions $P_{\mathcal{D}}(X)$ and $P_{\mathcal{D}}(Y)$, respectively. Moreover, the joint probability distribution $P_{\theta_n}\left(X, Y\right)$ converges to $P_{\mathcal{D}}\left(X, Y\right)$ in probability.*

The proof is provided in Appendix D.2. The theorem ensures that the MinVI objective can lead to a good generative model estimating the joint data-generating distribution of multimodal data. The theorem holds under two assumptions: consistency of density estimators and ergodicity of transition operators. The ergodicity condition is satisfied for a wide variety of neural networks, such as RBM or DBM.[2] The consistency assumption is more difficult to satisfy, and the aforementioned deep energy-based

[2]For energy-based models like RBM and DBM, it is straightforward to see that every state has non-zero probability and can be reached from any other state. However, the mixing of the chain might be slow in practice.

models or RNN may not satisfy the condition due to the model capacity limitation or approximated posteriors (e.g., factorial distribution). However, deep architectures are arguably among the most promising models for approximating the true conditionals from multimodal data. We expect that more accurate approximation of the true conditional distributions would lead to better performance in our multimodal learning framework, and we leave it for future work.

We note that our Theorem VI.1 is related to composite likelihood methods [87] and dependency networks [42]. For composite likelihood, the consistency result is derived upon a well-defined graphical model (e.g., Markov network) and the joint distribution converges in the sense that the maximum composite likelihood estimators are consistent for the parameters associated with the graphical model. However, in Theorem VI.1, it is not necessary to design a full graphical model (e.g., of the joint distribution) with analytical forms; for example, the two conditionals can be defined by neural networks with different parameters. In this case, the joint distribution is defined implicitly, and the setting is similar to general dependency networks [42]. However, [42] uses ordered pseudo-Gibbs samplers which may be unstable (i.e., inconsistencies between the local conditionals and the true conditionals can be amplified to a large inconsistency between the model joint distribution and the true joint distribution). In our case, we prove that the implicit model joint distribution will converge to the true joint distribution under assumptions that can plausibly hold for deep architectures.

6.3 Application to Multimodal Deep Learning

In this section, we describe the MinVI learning in multimodal deep learning framework. To overview our pipeline, we use the commonly used network architecture that consists of layers of modality-specific deep networks followed by a layer of neural network that jointly models the multiple modalities [99, 121]. The network is trained in two steps: In layer-wise pretraining, each layer of modality-specific deep network is trained

using restricted Boltzmann machines (RBMs). For the top-layer shared network, we train MRBM with MinVI objective (Section 6.3.2). Then, we finetune the whole deep network by constructing multimodal deep recurrent neural network (MDRNN) (Section 6.3.3).

6.3.1 Restricted Boltzmann machines for multimodal learning

The restricted Boltzmann machine (RBM) is an undirected graphical model that defines the distribution of visible units using hidden units. For multimodal input, we define the joint distribution of multimodal RBM (MRBM) [99, 121] as $P(x, y, h) = \frac{1}{Z} \exp\Big(- E(x, y, h)\Big)$ with the energy function:

$$E(x, y, h) = -\sum_{i=1}^{D_x} \sum_{k=1}^{K} x_i W_{ik}^x h_k - \sum_{j=1}^{D_y} \sum_{k=1}^{K} y_j W_{jk}^y h_k - \sum_{k=1}^{K} b_k h_k - \sum_{i=1}^{D_x} c_i^x x_i - \sum_{j=1}^{D_y} c_j^y y_j, \quad (6.5)$$

where Z is the normalizing constant, $x \in \{0, 1\}^{D_x}$, $y \in \{0, 1\}^{D_y}$ are the binary visible units of multimodal input (i.e., observations), and $h \in \{0, 1\}^K$ are the binary hidden units (i.e., latent variables). $W^x \in \mathbb{R}^{D_x \times K}$ defines the weights between x and h, and $W^y \in \mathbb{R}^{D_y \times K}$ defines the weights between y and h. $c^x \in \mathbb{R}^{D_x}$, $c^y \in \mathbb{R}^{D_y}$, and $b \in \mathbb{R}^K$ are bias vectors corresponding to x, y, and h, respectively. Note that the MRBM is equivalent to an RBM whose visible units are constructed by concatenating the visible units of multiple input modalities, i.e., $v = [x\,; y]$.

Due to bipartite structure, units in the same layer are conditionally independent given the units of the other layer, and the conditional probabilities are written as follows:

$$P(h_k = 1 \mid x, y) = \sigma\Big(\sum_i W_{ik}^x x_i + \sum_j W_{jk}^y y_j + b_k \Big), \quad (6.6)$$

$$P(x_i = 1 \mid h) = \sigma\Big(\sum_k W_{ik}^x h_k + c_i^x \Big), \quad P(y_j = 1 \mid h) = \sigma\Big(\sum_k W_{jk}^y h_k + c_j^y \Big), \quad (6.7)$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$. Similar to the standard RBM, the MRBM can be trained to maximize the joint LL ($\log P(x, y)$) using stochastic gradient descent (SGD) while approximating the gradient with CD [44] or persistent CD (PCD) [127]. In our case, however, we train the MRBM in MinVI criteria. We will discuss the inference and training algorithms in Section 6.3.2.

When we have access to all data modalities, we can use Equation (6.6) for exact posterior inference. On the other hand, when some of the input modalities are missing, the inference is intractable, and we resort to the variational method. For example, when we are given x but not y, the true posterior can be approximated with a fully factorized distribution $Q(y, h) = \prod_j \prod_k Q(y_j)Q(h_k)$ by minimizing the $KL\Big(Q(y, h) \| P_\theta(y, h|x)\Big)$. This leads to the following fixed-point equations:

$$\hat{h}_k = \sigma\Big(\sum_i W^x_{ik} x_i + \sum_j W^y_{jk} \hat{y}_j + b_k\Big), \; \hat{y}_j = \sigma\Big(\sum_k W^y_{jk} \hat{h}_k + c^y_j\Big), \tag{6.8}$$

where $\hat{h}_k = Q(h_k)$ and $\hat{y}_j = Q(y_j)$. We provide a derivation in Appendix D.3. The variational inference proceeds by alternately updating the mean-field parameters $\hat{h}$ and $\hat{y}$ that are initialized with all zeros.

6.3.2 Training algorithms

CD-PercLoss. As in Equation (6.2), the objective function can be decomposed into two conditional LLs, and the MRBM with MinVI objective can be trained equivalently by training the two conditional RBMs (CRBMs) while sharing the weights. Since the objective functions are the sum of two conditional LLs, we compute the (approximate) gradient of each CRBM separately using CD-PercLoss [94] and accumulate them to update parameters.[3]

[3]In CD-PercLoss learning, we run separate Gibbs chains for different conditioning variables and select the negative particles with the lowest free energy among sampled particles. We refer [94] for further details.

Multi-Prediction. We found a few practical issues of CD-PercLoss training in our application. In particular, there exists a difference between the encoding process of training and testing, especially when the unimodal query (e.g., when one of the data modalities is missing) is considered for testing. As an alternative objective, we propose multi-prediction (MP) training of MRBM in MinVI criteria. The MP training was originally proposed to train deep Boltzmann machines [36] as an alternative to the stochastic approximation learning [111]. The idea is to train the model to be good at predicting any subset of input variables given the rest of them by constructing the recurrent network with encoding function derived from the variational inference problem.

The MP training can be adapted to learn MRBM with MinVI objective with some modifications. For example, the CRBM with an objective $\log P(y|x)$ can be trained by randomly selecting the subset of variables to be predicted only from the target modality y, but the conditioning modality x is assumed to be given in all cases. Specifically, given an arbitrary subset $S \subset \{1, \cdots, D_y\}$ drawn from the independent Bernoulli distribution P_S, the MP algorithm predicts $y_S = \{y_j : j \in S\}$ given x and $y_{\setminus S} = \{y_j : j \notin S\}$ through the iterative encoding function derived from fixed-point equations:

$$\hat{h}_k = \sigma\Big(\sum_i W^x_{ik} x_i + \sum_{j \in S} W^y_{jk} \hat{y}_j + \sum_{j \notin S} W^y_{jk} y_j + b_k\Big),\ \hat{y}_j = \sigma\Big(\sum_k W^y_{jk} \hat{h}_k + c^y_j\Big), j \in S, \quad (6.9)$$

which is a solution to the variational inference problem $\min_Q KL\left(Q(y_S, h) \| P_\theta(y_S, h|x, y_{\setminus S})\right)$ with factorized distribution $Q(y_S, h) = \prod_{j \in S} \prod_k Q(y_j) Q(h_k)$. Note that Equation (6.9) is similar to the Equation (6.8) except that only $y_j, j \in S$ are updated. Using an iterative encoding function, the network parameters are trained using SGD while computing the gradient by backpropagating the error between the prediction and the ground truth of y_S through the derived recurrent network. The MP formulation (e.g., encoding function) of the CRBM with $\log P(x|y)$ can be derived similarly, and the gradients are simply the addition of two gradients that are computed individually.

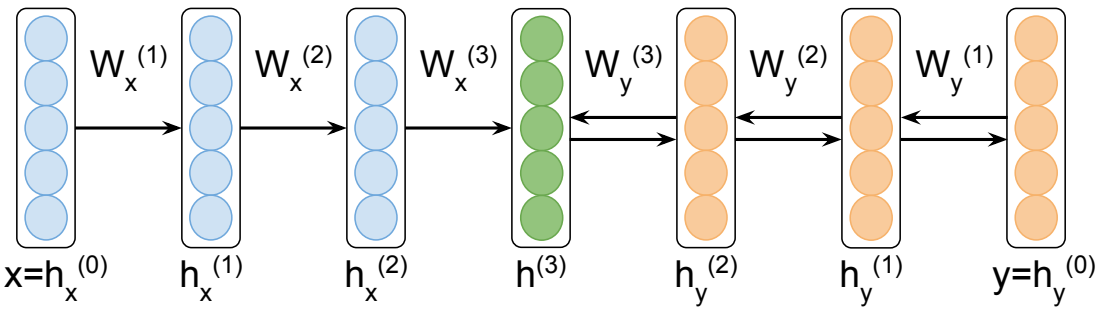


Figure 6.1: An instance of MDRNN with target y given x. Multiple iterations of bottom-up updates ($y \rightarrow h^{(3)}$; Equation (6.11) and (6.12)) and top-down updates ($h^{(3)} \rightarrow y$; Equation (6.13)) are performed. The arrow indicates encoding direction.

We have two additional hyper parameters, the number of mean-field updates and the sampling ratio of a subset S to be predicted from the target data modality. In our experiments, it was sufficient to use $10 \sim 20$ iterations until convergence. We used a sampling ratio of 1 (i.e., all the variables in the target data modality are to be predicted) since we are already conditioned on one data modality, which is sufficient to make a good prediction of variables in the target data modality.

6.3.3 Finetuning with recurrent neural network

Motivated from the MP training of MRBM, we propose a multimodal deep recurrent neural network (MDRNN) that tries to predict the target modality given the input modality through the recurrent encoding function. The MDRNN iteratively performs a full pass of bottom-up and top-down encoding from bottom-layer visible variables to top-layer joint representation back to bottom-layer through the modality-specific deep network corresponding to the target. We show an instance of $L = 3$ layer MDRNN in

Figure 6.1, and the encoding functions are written as follows:[4]

$$x \to h_x^{(L-1)} : h_x^{(l)} = \sigma\left(W^{x,(l)\top} h_x^{(l-1)} + b^{x,(l)}\right), l = 1 \to L-1, \tag{6.10}$$

$$y \to h_y^{(L-1)} : h_y^{(l)} = \sigma\left(W^{y,(l)\top} h_y^{(l-1)} + b^{y,(l)}\right), l = 1 \to L-1, \tag{6.11}$$

$$h_x^{(L-1)}, h_y^{(L-1)} \to h^{(L)} : h^{(L)} = \sigma\left(W^{x,(L)\top} h_x^{(L-1)} + W^{y,(L)\top} h_y^{(L-1)} + b^{(L)}\right), \tag{6.12}$$

$$h^{(L)} \to y : h_y^{(l-1)} = \sigma\left(W^{y,(l)} h_y^{(l)} + b^{y,(l-1)}\right), l = L \to 1. \tag{6.13}$$

Here, we define $h_x^{(0)} = x$ and $h_y^{(0)} = y$, and the visible variables of the target modality are initialized with zeros. In other words, in the initial bottom-up update, we compute $h^{(L)}$ only from x while setting $y = 0$ using Equations (6.10), (6.11), and (6.12). Then, we run multiple iterations of top-down (Equation (6.13)) and bottom-up updates (Equations (6.11) and (6.12)). Finally, we compute the gradient by backpropagating the reconstruction error of target modality through the network.

6.4 Experiments

6.4.1 Toy example on MNIST

In our first experiment, we evaluate the proposed learning algorithm on the MNIST handwritten digit recognition dataset [78]. We consider left and right halves of the digit images as two input modalities and report the recognition performance with different combinations of input modalities at the test time, such as full (left + right) or missing (left or right) data modalities. We compare the performance of the MRBM trained with 1) ML objective using PCD [127], or MinVI objectives with 2) CD-PercLoss or 3) MP training. The recognition errors are provided in Table 6.1. Compared to ML training, the recognition errors for unimodal queries are reduced by more than a

[4]There could be different ways of constructing MDRNN; for instance, one can construct the RNN with DBM-style mean-field updates. In our empirical evaluation, however, running full pass of bottom-up and top-down updates performed the best, and DBM-style updates didn't give competitive results.

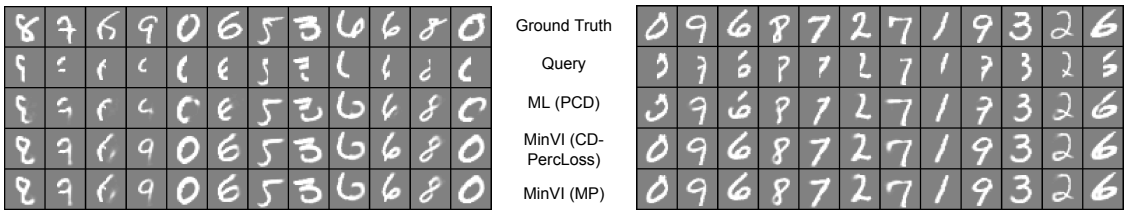


Figure 6.2: Visualization of samples with inferred missing modality. From top to bottom, we visualize ground truth, left or right halves of digits, generated samples with inferred missing modality using MRBM with ML objective, MinVI objective using CD-PercLoss and MP training methods.

Input modalities at test time	Left+Right	Left	Right
ML (PCD)	**1.57**%	14.98%	18.88%
MinVI (CD-PercLoss)	1.71%	9.42%	11.02%
MinVI (MP)	1.73%	**6.58**%	**7.27**%

Table 6.1: Test set errors on handwritten digit recognition dataset using MRBMs with different training objectives and learning methods. The joint representation was fed into linear SVM for classification.

half with MP training of MinVI objective. For multimodal queries, the model trained with ML objective performed the best, although the performance gain was incremental. CD-PercLoss training of MinVI objective also showed significant improvement over ML training, but the errors were not as low as those obtained with MP training. We hypothesize that, although it is an approximation of MinVI objective, the exact gradient for MP algorithm makes learning more efficient than CD-PercLoss. For the rest of the chapter, we focus on MP training method.

In Figure 6.2, we visualize the generated samples conditioned on one input modality (e.g., left or right halves of digits). There are many samples generated by the models with MinVI objective that look clearly better than those generated by the model with ML objective.

6.4.2 MIR-Flickr database

In this section, we evaluate our methods on MIR-Flickr database [54], which is composed of 1 million examples of images and their user tags collected from the social photo-sharing website Flickr. Among those, 25000 examples were annotated with 24 potential topics and 14 regular topics, which leads to 38 classes in total with distributed class membership. The topics included object categories such as dog, flower, and people, or scenic concepts such as sky, sea, and night.

We used the same visual and text features as in [121].[5] Specifically, the image feature was a 3857 dimensional vector composed of Pyramid Histogram of Words (PHOW) features [14], GIST [100], and MPEG-7 descriptors [90]. We preprocessed the image features to have zero mean and unit variance for each dimension across all examples. The text feature was a word count vector of $2,000$ most frequent tags. The number of tags varied from 0 to 72, with 5.15 tags per example in average.

Following the experimental protocol [55, 121], we randomly split the labeled examples into $15,000$ for training and $10,000$ for testing, and used $5,000$ from training set for validation. We iterated the procedure for 5 times and report the mean average precision (mAP) averaged over 38 classes.

Model architecture. We used the network composed of $[3857, 1024, 1024]$ variables for visual pathway, $[2000, 1024, 1024]$ variables for text pathway, and 2048 variables for top-layer MRBM, as used in [121]. As described in Section 6.3, we pretrained the modality-specific deep networks in a greedy layerwise way, and finetuned the whole network by initializing MDRNN with the pretrained network. Specifically, we used gaussian RBM for the bottom layer of visual pathway and binary RBM for text pathway.[6] The intermediate layers were trained with binary RBMs, and the top-layer MRBM was

[5] `http://www.cs.toronto.edu/~nitish/multimodal/index.html`

[6] We assumed text features as binary, which is different from [121] where they modeled using replicated-softmax RBM [110]. The rationale is that the tags are not likely to be assigned more than once for single image.

Model	Multimodal query
Autoencoder	0.610
Multimodal DBM [121]	0.609
Multimodal DBM† [122]	0.641
MK-SVM [38]	0.623
TagProp [131]	0.640
MDRNN	**0.686** $\pm$ 0.003
Model	Unimodal query
Autoencoder	0.495
Multimodal DBM [121]	0.531
MK-SVM [38]	0.530
MDRNN	**0.607** $\pm$ 0.005

Table 6.2: Test set mAPs on MIR-Flickr database. We implemented autoencoder following the description in [99]. Multimodal DBM† is supervised finetuned model. See [122] for details.

trained using MP training algorithm. For the layer-wise pretraining of RBMs, we used PCD [127] to approximate the gradient. Since our algorithm requires both data modalities during training, we excluded examples with too sparse or no tags from unlabeled dataset and used about 750K examples with at least 2 tags. After unsupervised training, we extracted joint feature representations of the labeled training data and use them to train multiclass logistic regression classifiers.

Recognition tasks. For recognition tasks, we trained multiclass logistic regression classifiers using joint representations as input features. Depending on the availability of data modalities at testing time, we evaluated the performance using multimodal queries (i.e., both visual and text data are available) and unimodal queries (i.e., visual data is available while the text data is missing). In Table 6.2, we report the test set mAPs of our proposed model and compared to other methods. The proposed MDRNN outperformed the previous state-of-the-art in multimodal queries by 4.5% in mAP. The performance improvement becomes more significant for unimodal queries, achieving 7.6% improvement in mAP over the best published result. As we used the same input features

in [121], the results suggest that our proposed algorithm learns better representations shared across multiple modalities.

For a closer look into our model, we performed an additional control experiment to explore the benefit of recurrent encoding of MDRNN. Specifically, we compared the performance of the models with different number of mean-field iterations.[7] We report the validation set mAPs of models with different number of iterations ($0 \sim 10$) in Table 6.3. For multimodal query, the MDRNN with 10 iterations improves the recognition performance by only 0.8% compared to the model with 0 iterations. However, the improvement becomes significant for unimodal query, achieving 5.0% performance gain. In addition, the largest improvement was made when we have at least one iteration (from 0 to 1 iteration, 3.4% gain; from 1 to 10 iteration, 1.6% gain). This suggests that a crucial factor of improvement comes from the inference with reconstructed missing data modality (e.g., text features), and the quality of inferred missing modality improves as we increase the number of iterations.

# iterations	0	1	2	3	5	10
Multimodal query	0.677	0.678	0.679	0.680	0.682	**0.685**
Unimodal query	0.557	0.591	0.599	0.602	0.605	**0.607**

Table 6.3: Validation set mAPs on MIR-Flickr database with different number of mean-field iterations.

Retrieval tasks. We performed retrieval tasks using multimodal and unimodal input queries. Following [121], we selected $5,000$ image-text pairs from the test set to form a database and use $1,000$ disjoint set of examples from the test set as queries. For each query example, we computed the relevance score to the data points as a cosine similarity of joint representations. The binary relevance labels between query and the

[7] In [99], Ngiam et al. proposed the "video-only" deep autoencoder whose objective is to predict audio data and reconstruct video data when only video data is given as an input during the training. Our baseline model (MDRNN with 0 iterations) is similar, but different since we don't have a reconstruction training objective.

data points are determined 1 if any of the 38 class labels are overlapped. Our proposed model achieves **0.633** mAP with multimodal query and **0.638** mAP with unimodal query. This significantly outperforms the performance of multimodal DBM [121], which reported 0.622 mAP with multimodal query and 0.614 mAP with unimodal query. We show retrieved examples with multimodal queries in Figure 6.3.

Figure 6.3: Retrieval results with multimodal queries. The leftmost image-text pairs are multimodal query samples and those in the right side of the bar are retrieved samples with the highest similarities to the query sample from the database.

6.4.3 PASCAL VOC 2007

We evaluate the proposed algorithm on PASCAL VOC 2007 database. The original dataset does not contain user tags, but Guillaumin et al. [38] have collected user tags from Flickr website.[8]

Motivated by the success of convolutional neural networks (CNNs) on large-scale visual object recognition [65], we used the DeCAF$_7$ features [25] as an input features for visual pathway, where DeCAF$_7$ is 4096 dimensional feature extracted from the CNN trained on ImageNet [24]. For text features, we used the vocabulary of size 804 suggested by [38]. For unsupervised feature learning of MDRNN, we used unlabeled data of

[8] `http://lear.inrialpes.fr/people/guillaumin/data.php`

MIR-Flickr database while converting the text features using the new vocabulary from PASCAL database. The network architecture used in this experiment was as follows: $[4096, 1536, 1536]$ variables for the visual pathway, $[804, 512, 1536]$ variables for the text pathway, and 2048 variables for top-layer joint network.

Following the standard practice, we reported the mAP over 20 object classes. The performance improvement of our proposed method was significant, achieving **81.5**% mAP with multimodal queries and **76.2**% mAP with unimodal queries, whereas the performance of the baseline model was 74.5% mAP with multimodal queries (DeCAF_7 + Text) and 74.3% mAP with unimodal queries (DeCAF_7).

6.5 Conclusion

Motivated by the property of good generative models of multimodal data, we proposed a novel multimodal deep learning framework based on variation of information. The minimum variation of information objective enables to learn good shared representations of multiple heterogeneous data modalities with a better prediction of missing input modality. We demonstrated the effectiveness of our proposed method on multimodal RBM and its deep extensions and showed state-of-the-art recognition performance on MIR-Flickr database and competitive performance on PASCAL VOC 2007 database with multimodal (visual + text) and unimodal (visual only) queries.

CHAPTER VII

Learning to Predict Structured Outputs using Stochastic Convolutional Networks

7.1 Introduction

To build an end-to-end system for structured output prediction, one needs to incorporate the probabilistic inference, as it may not be a simple many-to-one function approximation problem (e.g., recognition and classification), but could be a task of mapping an input to many possible outputs, especially when the input data contains missing values or ambiguities. In other words, it is desirable to model the non-unimodal distribution of structured output. Although the great success of convolutional neural networks (CNNs) on large-scale visual recognition [65] motivates us to apply deep neural networks to this problem, the CNN with deterministic inference is not suitable in modeling a distribution with multiple modes [125].

In this chapter, we propose convolutional neural networks with Gaussian stochastic neurons for structured output prediction and representation learning. In the light of recent development in variational inference and learning of directed graphical models, such as variational auto-encoder [62, 107, 63], we propose a conditional variational auto-encoder (condVAE). The condVAE is a conditional directed graphical model whose input (or observation) modulates both the prior on Gaussian latent variables and the

output variables. It is trained to maximize the conditional log-likelihood, and we formulate the variational learning objective of the condVAE in the framework of stochastic gradient variational Bayes (SGVB) [62]. In addition, we provide novel strategies to build robust structured prediction algorithms, such as recurrent prediction network architecture, input noise-injection, and multi-scale prediction training methods.

In experiments, we demonstrate the effectiveness of our proposed algorithm in comparison to the deterministic deep neural network counterparts in generating diverse but realistic output representations using stochastic inference. First, we demonstrate the importance of stochastic neurons in modeling the structured output when the input data is partially provided. Furthermore, we show that the proposed training schemes are complimentary, leading to strong pixel-level object segmentation and labeling performance on Caltech-UCSD Birds 200 and the subset of Labeled Faces in the Wild dataset.

7.2 Preliminary

7.2.1 Variational Auto-encoder

The variational auto-encoder (VAE) [62, 107] is a directed graphical model with certain types of latent variables, such as Gaussian latent variables. A generative process of the VAE is as follows: a set of latent variable $\mathbf{z}^*$ is generated from the prior distribution $p_\theta(\mathbf{z})$ and the data $\mathbf{x}^*$ is generated by the generative distribution $p_\theta(\mathbf{x}|\mathbf{z}^*)$ conditioned on $\mathbf{z}^*$: $\mathbf{z}^* \sim p_\theta(\mathbf{z}), \mathbf{x}^* \sim p_\theta(\mathbf{x}|\mathbf{z}^*)$.

The parameter estimation and posterior inference of directed graphical models with distributed latent variable representation is challenging since they exploit intractable posterior inference problem. However, the model parameters of the VAE can be efficiently estimated in stochastic gradient variational Bayes (SGVB) [62] framework, where the variational lower bound of the log-likelihood is used as a surrogate objective

function. Specifically, the variational lower bound is written as follows:

$$\begin{aligned}
\log p_\theta(\mathbf{x}) &= KL\left(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z}|\mathbf{x})\right) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\Big[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x},\mathbf{z})\Big] \quad (7.1)\\
&\geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\Big[-\log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{x},\mathbf{z})\Big] \quad (7.2)\\
&= -KL\left(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z})\right) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\Big[\log p_\theta(\mathbf{x}|\mathbf{z})\Big] \quad (7.3)
\end{aligned}$$

In this framework, a proposal distribution $q_\phi(\mathbf{z}|\mathbf{x})$, which is also known as a "recognition" model, is introduced to approximate the true posterior $p_\theta(\mathbf{z}|\mathbf{x})$. The multilayer perceptrons (MLPs) are used to model the recognition and the generation models. Assuming Gaussian latent variables, the first term of Equation (7.3) can be marginalized, while the second term is not. Instead, the second term can be approximated by drawing samples $\mathbf{z}^{(l)}$ by the recognition distribution $q_\phi(\mathbf{z}|\mathbf{x})$, and the empirical objective of the VAE with Gaussian latent variables is written as follows:

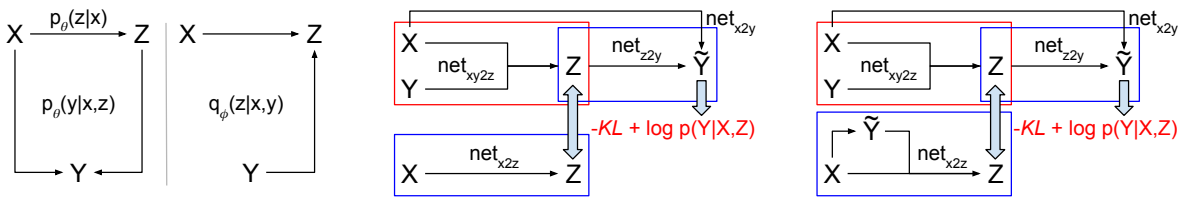
$$\widetilde{\mathcal{L}}_{\text{VAE}}(\mathbf{x};\theta,\phi) = -KL\left(q_\phi(\mathbf{z}|\mathbf{x})\|p_\theta(\mathbf{z})\right) + \frac{1}{L}\sum_{l=1}^{L}\log p_\theta(\mathbf{x}|\mathbf{z}^{(l)}) \quad (7.4)$$

$$\text{where } \mathbf{z}^{(l)} = g_\phi(\mathbf{x},\epsilon^{(l)}),\ \epsilon^{(l)} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$$

Note that the recognition distribution $q_\phi(\mathbf{z}|\mathbf{x})$ is reparameterized with a deterministic, differentiable function $g_\phi(\cdot,\cdot)$, whose arguments are data $\mathbf{x}$ and the noise variable ϵ. This trick allows error backpropagation through the Gaussian latent variables, which is essential in VAE training as it is composed of multiple MLPs for recognition and generation models. As a result, the VAE can be trained efficiently using stochastic gradient descent (SGD).

7.3 Conditional Variational Auto-encoder

In this section, we formulate a conditional variational auto-encoder (condVAE) for structured output prediction. There are three types of variables in condVAE: input



(a) graphical model (b) condVAE (schematic) (c) recurrent prediction net

Figure 7.1: (a) A graphical model representation of the condVAE describing the generative (left) and recognition (right) processes, (b) a schematic flowchart of the condVAE during the training, and (c) that of the condVAE with recurrent prediction network.

variable $\mathbf{x}$, output variable $\mathbf{y}$, and the latent variable $\mathbf{z}$. The generative process of the condVAE is given as follows: for given observation $\mathbf{x}^*$, $\mathbf{z}^*$ is drawn from the conditional prior distribution $p_\theta(\mathbf{z}|\mathbf{x}^*)$, and the output $\mathbf{y}^*$ is generated from the generation distribution $p_\theta(\mathbf{y}|\mathbf{x}^*, \mathbf{z}^*)$. Note that the conditional prior of the latent variable $\mathbf{z}$ is modulated by the input $\mathbf{x}$ in our formulation; however, the constraint can be easily relaxed to make the latent variable statistically independent of an input variable, i.e., $p_\theta(\mathbf{z}|\mathbf{x}) = p_\theta(\mathbf{z})$ [63].

The variational lower bound of the condVAE is written as follows:

$$\log p_\theta(\mathbf{y}|\mathbf{x}) \quad \geq \quad -KL\left(q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})\|p_\theta(\mathbf{z}|\mathbf{x})\right) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})}\Big[\log p_\theta(\mathbf{y}|\mathbf{x},\mathbf{z})\Big] \tag{7.5}$$

and the empirical objective function of the condVAE is written accordingly:

$$\widetilde{\mathcal{L}}_{\text{condVAE}}(\mathbf{x},\mathbf{y};\theta,\phi) = -KL\left(q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})\|p_\theta(\mathbf{z}|\mathbf{x})\right) + \frac{1}{L}\sum_{l=1}^{L}\log p_\theta(\mathbf{y}|\mathbf{x},\mathbf{z}^{(l)}) \tag{7.6}$$

$$\text{where } \mathbf{z}^{(l)} = g_\phi(\mathbf{x},\mathbf{y},\epsilon^{(l)}),\ \epsilon^{(l)} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$$

The condVAE is composed of multiple MLPs as shown in Figure 7.1(b). Specifically, we construct recognition network net_{xy2z} for $q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})$, conditional prior network net_{x2z} for $p_\theta(\mathbf{z}|\mathbf{x})$, and a generation network for $p_\theta(\mathbf{y}|\mathbf{x},\mathbf{z})$, which can be decomposed into latent-to-output network net_{z2y} and input-to-output network net_{x2y}. We discuss inference and

learning subsequently.

7.3.1 Output inference and estimation of the conditional likelihood

Once the model parameters are estimated, we make a prediction of an output $\mathbf{y}$ from an input $\mathbf{x}$ by following a generative process of the condVAE. To evaluate the model on structured output prediction tasks, we can measure a prediction accuracy by performing a deterministic inference without sampling $\mathbf{z}$, i.e., $\mathbf{y}^* = \arg\max_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{z}^*), \mathbf{z}^* = \mathbb{E}\Big[\mathbf{z}|\mathbf{x}\Big]$.[1]

Another way of evaluating the conditional generative model is to compute the conditional likelihood (CL) of the test data. A straightforward approach is to draw samples $\mathbf{z}$'s using conditional prior and take the average of the likelihoods, which we call an estimation by generative sampling (GS):

$$p_\theta(\mathbf{y}|\mathbf{x}) \approx \frac{1}{S}\sum_{s=1}^{S} p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{z}^{(s)}), \quad \mathbf{z}^{(s)} \sim p_\theta(\mathbf{z}|\mathbf{x}) \tag{7.7}$$

A key drawback of this approach is that we need to draw large number of samples to obtain an accurate estimation to the true conditional likelihoods. Alternatively, as proposed in [107], we use the importance sampling to estimate the conditional likelihoods of the condVAE:

$$p_\theta(\mathbf{y}|\mathbf{x}) \approx \frac{1}{S}\sum_{s=1}^{S} \frac{p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{z}^{(s)})p_\theta(\mathbf{z}^{(s)}|\mathbf{x})}{q_\phi(\mathbf{z}^{(s)}|\mathbf{x}, \mathbf{y})}, \quad \mathbf{z}^{(s)} \sim q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{y}) \tag{7.8}$$

7.3.2 Learning to predict structured output

The model parameters of the MLP components in condVAE can be jointly trained in the framework of SGVB, whose gradient can be efficiently computed using backprop-

[1] Alternatively, we can draw multiple $\mathbf{z}$'s from the conditional prior distribution and use the average of the posteriors to make a prediction, i.e., $\mathbf{y}^* = \arg\max_{\mathbf{y}} \frac{1}{L}\sum_{l=1}^{L} p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{z}^{(l)}), \mathbf{z}^{(l)} \sim p_\theta(\mathbf{z}|\mathbf{x})$. In practice, however, the stochastic inference is not recommended when evaluating the prediction accuracy since it requires a large number of samples to be drawn (e.g., $100 \sim 10000$) to improve upon deterministic inference.

agation [62, 107].

One should note that, in the condVAE, there are two pathways to draw samples that are used for either training or testing. In other words, the generation network $p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{z})$ is modulated by $\mathbf{z}$ sampled from the recognition distribution $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{y})$ during the training, while for testing, it is modulated by $\mathbf{z}$ drawn from the conditional prior $p_\theta(\mathbf{z}|\mathbf{x})$. This implies that, to make an accurate prediction for an unseen test data, the conditional prior, which draws latent variables *without* $\mathbf{y}$, should be trained to hallucinate the recognition distribution modulated by *both* $\mathbf{x}$ and $\mathbf{y}$. Although the negative KL divergence term in Equation (7.6) enforces two distributions to be similar, there exists a bit of discrepancy between two inference networks.

To reduce this gap and enhance the discriminative power, we propose to train generation network of the condVAE in a discriminative way by formulating an instance of Gaussian stochastic neural network (GSNN). The GSNN is a class of stochastic feed-forward neural network whose stochastic neurons are all Gaussian variables. We construct the GSNN by bridging the conditional prior network (net_{x2z}) and the latent-to-output generation network (net_{z2y}). The objective function of the GSNN induced from the condVAE is written as follows:

$$\widetilde{\mathcal{L}}_{\text{GSNN}}(\mathbf{x}, \mathbf{y}; \theta, \phi) = \frac{1}{L} \sum_{l=1}^{L} \log p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{z}^{(l)}) \text{ , where } \mathbf{z}^{(l)} = g_\theta(\mathbf{x}, \epsilon^{(l)}),\ \epsilon^{(l)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad (7.9)$$

Note that in Equation (7.9), we sample $\mathbf{z}$ from a differentiable deterministic function $g_\theta(\cdot, \cdot)$, which is a reparameterization of the conditional prior distribution $p_\theta(\mathbf{z}|\mathbf{x})$, not the recognition distribution $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{y})$ as in Equation (7.6). Finally, we obtain a hybrid objective function by combining the two objective functions of condVAE and GSNN:

$$\widetilde{\mathcal{L}}_{\text{hybrid}} = \alpha \widetilde{\mathcal{L}}_{\text{condVAE}} + (1 - \alpha) \widetilde{\mathcal{L}}_{\text{GSNN}} \quad (7.10)$$

α balances the two objectives. Note that when $\alpha = 1$, we recover the condVAE objective;

when $\alpha = 0$, the trained model will be simply a GSNN without recognition network.

7.3.3 Conditional VAE for image segmentation and labeling

There have been several approaches proposed for image segmentation and scene labeling [31, 102, 33] using CNNs. Indeed, each of the MLPs in the condVAE can be extended to convolutional architecture while maintaining an efficient learning based on stochastic optimization. In addition, we provide novel strategies to obtain a robust model for image segmentation.

7.3.3.1 Training with input omission noise

Learning representation of structured output variables using deep convolutional network is challenging since obtaining sufficient amount of training data with structured labels is much more expensive than those with class labels. To learn a neural network for structured prediction that can generalize well to unseen data with a limited number of labeled data, we propose to train the network while injecting noise to the input. The learning procedure is very simple; corrupt the input data $\mathbf{x}$ into $\tilde{\mathbf{x}}$ according to predefined noise process and optimize the network with the following objective: $\widetilde{\mathcal{L}}(\tilde{\mathbf{x}}, \mathbf{y})$. The noise process could be arbitrary; for example, for object segmentation application, one can consider block or random omission noise. This can be viewed as providing more challenging input-output prediction situation during the training where there exists block occlusion or missing input, which indeed is motivated by a desirable property of robust structured prediction algorithms. The proposed training strategy also is related to the denoising training methods [132], but in our case, we inject noise to the input data only and does not try to reconstruct them.

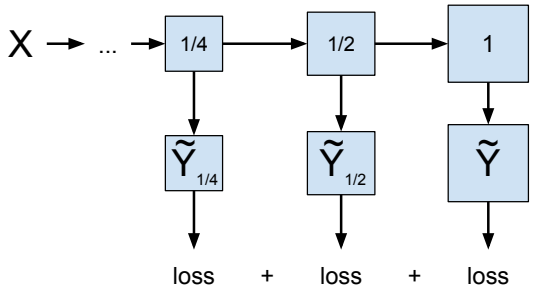


Figure 7.2: Multi-scale training.

7.3.3.2 Very deep convolutional network with recurrent prediction network

The recurrent network architecture has been applied for structured output prediction problems [102, 58, 120] to sequentially update the guess by revising the previous prediction. Furthermore, the recurrent architecture enables to effectively deepen the convolutional network, which has shown to be essential in achieving a strong visual recognition performance [124, 116], while not significantly increasing the number of model parameters due to weight sharing.

As we see in Figure 7.1(b), there are two pathways to the output $\mathbf{y}$; one from the latent variable $\mathbf{z}$ (net_{z2y}), and the other from the input variable $\mathbf{x}$ (net_{x2y}). Although the latent-to-output pathway plays an important role in final prediction, the direct input-to-output pathway can make a reasonable initial prediction on $\mathbf{y}$. In our case, we propose to bridge the output of the direct pathway $\tilde{\mathbf{y}} = \text{net}_{\text{x2y}}(\mathbf{x})$ as an input to the conditional prior network net_{x2z}. This results in a symmetric network architecture between the recognition and conditional prior networks, which allows better optimization with respect to the negative KL divergence term in the objective function. We illustrate our proposal recurrent architecture in Figure 7.1(c).

7.3.3.3 Multi-scale prediction training

As the image size gets larger (e.g., 128×128), it becomes more challenging to make a fine-grained pixel-level prediction (e.g., reconstruction of the image, prediction of

semantic labels). The multi-scale approaches have been used in the sense of forming a multi-scale image pyramid for an input [31], rather than multi-scale prediction at the output, which allows you to make a sequential low-to-high resolution prediction. Here, we propose to train a proposed stochastic convolutional networks to predict an output at different scales. By doing so, we can make a global-to-local, coarse-to-fine-grained prediction of pixel-level semantic labels. Figure 7.2 describes the multi-scale prediction at 3 different scales (1/4, 1/2, and original) for the training.

7.4 Experiments

We demonstrate the importance of stochastic neurons in modeling the distribution of the structured output variables with multiple modes. For the proof of concept, we create an artificial experimental setting for structured output prediction using MNIST database [78]. Then, we evaluate the proposed condVAE models on several benchmark database for object segmentation and labeling, such as Caltech-UCSD Birds (CUB) [138] and Labeled Faces in the Wild (LFW) [49].

7.4.1 Toy example: MNIST

To highlight the importance of probabilistic inference through stochastic neurons for structured output variables, we execute an experiment using MNIST database. Specifically, we divide each digit image into four quadrants, and take one, two, or three quadrant(s) as an input and the remaining quadrants as an output.[2] As we increase the number of quadrants for an output, the input to output mapping becomes closer to a one-to-many mapping.

We trained the proposed models (condVAE, GSNN) and the baseline deep neural network and compare their performance. The same network architecture, the MLP with

[2]Similar experimental setting has been used in multimodal learning framework, where the left- and right halves of the digit images are used as two data modalities [3, 120].

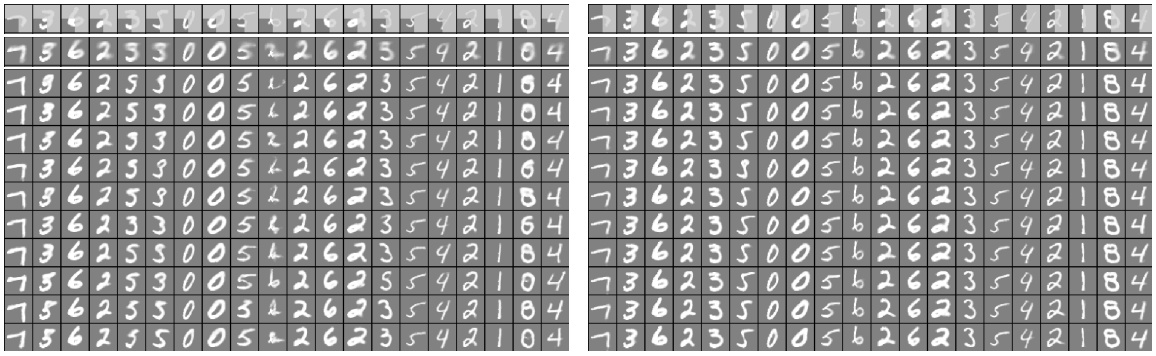


Figure 7.3: Visualization of generated samples with (left) 1 quadrant and (right) 2 quadrants for an input. We show (first) the input and the ground truth output overlaid with gray color, and (second) samples generated by the baseline NNs, and (rest) samples drawn from the condVAEs.

negative CLL	1 quad.	2 quads.	3 quads.
NN (baseline)	100.03 / 99.75	62.14 / 62.18	26.01 / 25.99
GSNN	100.03 / 99.82	62.48 / 62.41	26.20 / 26.29
condVAE	68.62 / 68.39	45.57 / 45.34	20.97 / 20.96
condVAE (IS)	64.05 / **63.91**	44.96 / **44.73**	20.97 / **20.95**
Performance gap	35.98 / 35.91	17.51 / 17.68	5.23 / 5.33
- per pixel	0.061 / 0.061	0.045 / 0.045	0.027 / 0.027

Table 7.1: The negative CLL on the validation/test sets of MNIST database. We increase the number of quadrants for an input from 1 to 3, and estimate the CLL using generative sampling by default. The performance gap between the condVAE (IS) and the baseline NN is reported.

two-layers of 1000 ReLUs for recognition, conditional prior, or generation networks, followed by 200 Gaussian latent variables, was used for all different models in various experimental settings. The early stopping is used during the training based on the estimation of the conditional likelihoods on the validation set.

For qualitative analysis, we visualize the generated output samples in Figure 7.3. As we can see, the baseline NNs can only make a single deterministic prediction and as a result the output looks blurry and doesn't look realistic digit images when combined with the input. On the other hand, the samples generated by the condVAEs are more realistic and diverse in shape; sometimes they can even change their identity, such as from 3 to 5 or 4 to 9, and vice versa.

Models	negative CLL (val)	negative CLL
CNN (baseline)	3420.90 ±96.87	3489.00 ±82.09
GSNN	3323.67 ±111.16	3426.57 ±48.49
condVAE	**786.60** ±15.32	**785.44** ±7.38
hybrid ($\alpha = 0.5$)	996.39 ±19.18	998.93 ±10.32

Table 7.2: The negative CLL on CUB database. We used importance sampling to estimate the CLL of condVAE and hybrid models.

We also provide a quantitative evidence by estimating the conditional log-likelihoods (CLLs) in Table 7.1. The CLLs of the proposed models are estimated in two ways as described in Section 7.3.1 For generative sampling, we draw $10,000$ samples per example for accurate estimation. For importance sampling, however, 100 samples per example was enough to obtain an accurate estimation of the CLL. We observed that the estimated CLLs of the condVAE significantly outperforms the baseline NN. Moreover, as measured by the per pixel performance gap, the performance improvement becomes more significant as we use smaller number of quadrants for an input, which is expected as the input-output mapping becomes closer to one-to-many mapping.

7.4.2 Visual Object Segmentation and Labeling

Caltech-UCSD Birds (CUB) database [138] includes $6,033$ images of birds from 200 species with annotations such as a bounding box of birds and a segmentation mask. Later, Yang et al. [144] annotated these images with more fine-grained segmentation masks by cropping the bird patches using the bounding boxes and resized them into 128×128 pixels. The training/test split proposed in [138] was used in our experiment, and for validation purpose, we partition the training set into 10 folds and cross-validated with the mean intersection over union (IoU) score over the folds. The final prediction on the test set was made by averaging the posterior from ensemble of 10 networks that are trained on each of the 10 folds separately. We increase the number of training examples by horizontal flipping the input and output images.

We extensively evaluate the variations of our proposed methods, such as condVAE,

Models		pixel (val)	IoU (val)	pixel	IoU
CNN (baseline)	recur., msc	92.25 ±0.27	81.89 ±0.67	93.24	83.96
GSNN	recur., msc	92.46 ±0.24	82.31 ±0.60	93.39	84.26
condVAE	flat, ssc	89.62 ±0.35	76.55 ±0.95	90.35	78.05
condVAE	recur. ssc	91.17 ±0.35	79.75 ±0.93	91.93	81.31
condVAE	recur., msc	92.24 ±0.27	81.86 ±0.74	93.03	83.53
hybrid ($\alpha = 0.5$)	recur., msc	92.60 ±0.25	82.57 ±0.83	93.35	84.16
CNN	recur., msc, NI	92.94 ±0.20	83.30 ±0.51	**93.76**	**85.09**
GSNN	recur., msc, NI	93.09 ±0.18	83.61 ±0.52	**93.93**	**85.38**
condVAE	recur., msc, NI	92.80 ±0.27	83.03 ±0.70	93.50	84.48
hybrid ($\alpha = 0.5$)	recur., msc, NI	93.03 ±0.30	83.47 ±0.66	**93.81**	**85.13**

Table 7.3: Labeling results on CUB database. We report both pixel-wise labeling accuracy and IoU score of the foreground region. The term "recur." refers the network with recurrent prediction architecture, which is used as opposed to the "flat", and "ssc" and "msc" refers single-scale and multi-scale prediction training, respectively. The "NI" refers the noise-injection training.

GSNN, and their combination while changing α, and provide a summary results on segmentation mask prediction task in Table 7.3. Specifically, we report the performance of the models with different network architectures (e.g., with and without recurrent prediction network) and training methods (e.g., multi-scale prediction or noise-injection training).

First of all, we note the significant performance improvement from all of the techniques proposed in Section 7.3.3, such as recurrent network architecture, multi-scale prediction training as well as noise-injection training. Having them as a default architecture and training method, we evaluated the variations of our proposed models and the baseline CNN. Interestingly, the baseline CNN already showed much better segmentation results compared to the previous state-of-the-art obtained by the max-margin Boltzmann machine (pixel accuracy: 90.42, IoU: 75.92 with GraphCut for post-processing) [144]. The condVAE model showed slightly worse performance than the baseline CNN; however, the variations of our proposed methods, such as GSNN and the hybrid model outperformed the baseline CNNs. In addition, we evaluate the CLL of the models and summarized the results in Table 7.2. As expected, the proposed

Models	pixel (val)	pixel	neg. CLL (val)	neg. CLL
CNN (baseline)	92.72 ±0.28	92.54 ±0.08	–	–
GSNN	92.88 ±0.18	92.61 ±0.19	–	–
condVAE	92.80 ±0.30	92.62 ±0.14	–	–
hybrid ($\alpha = 0.5$)	92.95 ±0.21	92.77 ±0.14	–	–
CNN (NI)	**93.58** ±0.34	**93.36** ±0.14	4864.07 ±167.82	5144.48 ±348.47
GSNN (NI)	**93.79** ±0.29	**93.51** ±0.21	4518.26 ±394.94	4901.84 ±319.20
condVAE (NI)	93.31 ±0.28	93.19 ±0.12	**1193.70** ±147.41	**1193.23** ±125.98
hybrid (NI, $\alpha = 0.5$)	**93.74** ±0.33	**93.50** ±0.14	1835.75 ±172.34	1841.48 ±147.23

Table 7.4: Labeling results and the negative CLL on LFW database. We report the pixel-wise 4-way (skin, hair, clothes, background) prediction accuracy. We used recurrent architecture and the models are trained with multi-scale prediction training and noise-injection methods. We used importance sampling to estimate the CLL of condVAE and hybrid models.

condVAE models (e.g., condVAE, hybrid) significantly outperform the baseline CNN, which confirms that the condVAE can generate better output representations in terms of higher compatibility to the ground truth by drawing multiple samples.

Labeled Faces in the Wild (LFW) database [49] has been widely used for face recognition and verification benchmark. As mentioned in [50], the face images that are segmented and labeled into semantically meaningful region labels (e.g., hair, skin, clothes) can greatly help understanding of the image through the visual attributes, which can be easily obtained from the face shape.

Following the segmentation and region labeling protocols [137, 58], we evaluate the performance of face parts labeling on the subset of LFW database. Specifically, we used the labeled dataset used in [137], which contains $1,046$ images that are labeled into 4 semantic categories, such as hair, skin, clothes, and background. We resized the images from 250×250 into 128×128 to use the same network architecture to the one used in the CUB experiment.

We provide a summary results of pixel-level segmentation accuracy and the negative CLL in Table 7.4. Similar trend has been observed as in the CUB database; the proposed stochastic neural networks performed slightly better or competitive to the baseline CNN

Dataset (measure)		CUB (IoU)		LFW (pixel)	
noise level	block size	CNN	condVAE	CNN	condVAE
25%	1	89.37	**98.52**	96.93	**99.22**
	4	88.74	**98.07**	96.55	**99.09**
	8	90.72	**96.78**	97.14	**98.73**
50%	1	74.95	**95.95**	91.84	**97.29**
	4	70.48	**94.25**	90.87	**97.08**
	8	76.07	**89.10**	92.68	**96.15**
70%	1	62.11	**89.44**	85.27	**89.71**
	4	57.68	**84.36**	85.70	**93.16**
	8	63.59	**76.87**	87.83	**92.06**

Table 7.5: Interactive segmentation results with input omission noise and weak supervision on CUB (left) and LFW (right) database. We report the pixel-level accuracy on the first validation set.

in terms of segmentation accuracy; however, the negative CLL of the proposed condVAE model is significantly lower than the baseline CNN, which ensures we can find a realistic samples from multiple samples. Finally, the performance of our methods outperform those of other existing methods, which report 90.0% in [137] or 90.7% in [58], setting state-of-the-art performance on LFW segmentation database.

7.4.3 Interactive object segmentation with input occlusion

We experimented with the interactive object segmentation task for the case of significant input noise with a weak supervision or partial output observation. To emulate the scenario, we randomly omit the input pixels at different levels of noise (25%, 50%, 70%) and different block size (1, 4, 8) and provide it as an input to the algorithm. For larger block size, the noise becomes more structured and this can be viewed as an interactive segmentation task with an occlusion. We also provide a partial segmentation with omission using the same mask used to omit the input pixels.

To make a prediction for condVAE with partial output observation ($\mathbf{y}_o$), we perform

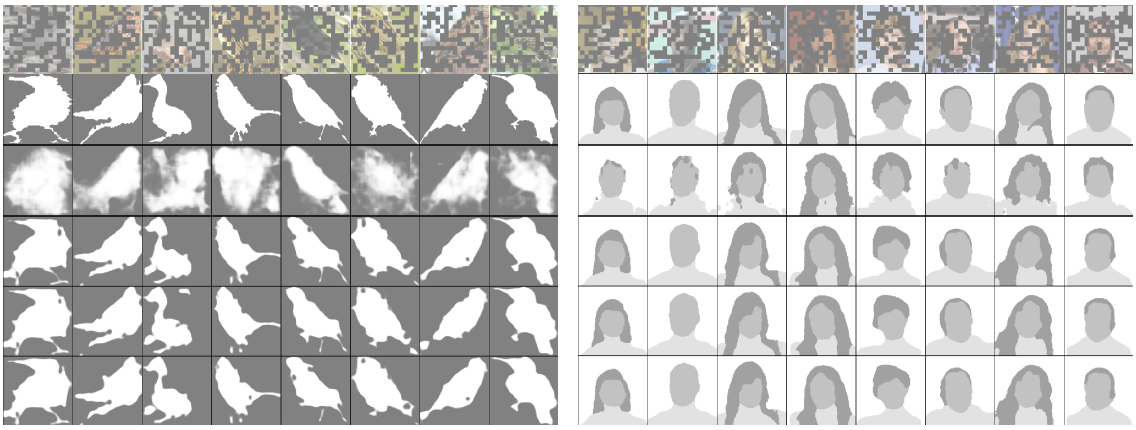


Figure 7.4: Visualization of (first row) input image with omission noise (noise level: 50%, block size: 8), (second row) ground truth segmentation, and (third) prediction by CNN, and (fourth to sixth) the generated samples by condVAE on (left) CUB and (right) LFW database.

iterative inference of unobserved output ($\mathbf{y}_u$) and the latent variables ($\mathbf{z}$) [107], i.e.,

$$\mathbf{y}_u^* \sim p_\theta(\mathbf{y}_u|\mathbf{x}, \mathbf{z}^*) \leftrightarrow \mathbf{z}^* \sim q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{y}_o, \mathbf{y}_u^*) \tag{7.11}$$

We report the summary results in Table 7.5. The condVAE performs well even when the noise level is high (e.g., 50%), where the CNN significantly fails. This is because the condVAE utilizes the partial segmentation information to refine the prediction of the rest iteratively. We also visualize the generated samples by the CNN and condVAE at noise level of 50% and in Figure 7.4. As we can see, the prediction made by the CNN is very blurry, but those samples generated by condVAE are sharper while maintaining the reasonable shapes. In addition, we can make multiple realistic segmentation masks that look different to each other using condVAE.

7.5 Conclusion

Modeling multi-modal distribution of the structured output variables is an essential research question to achieve good performance on structured output prediction problems. In this chapter, we proposed a stochastic neural networks for structured

output prediction based on the conditional deep generative model with Gaussian latent variables. The proposed model is scalable to large images and efficient in inference and learning. We demonstrated the importance of probabilistic inference when the distribution of output space has multiple modes, and showed strong performance in discriminative (e.g., segmentation accuracy) and generative (e.g., estimation of conditional log-likelihood, visualization of generated samples) tasks.

CHAPTER VIII

Conclusion and Future Work

8.1 Conclusion

Learning representation of the data is fascinating research problem as the machine can identify and capture the statistical structures and underlying patterns automatically from the data to better understand the high-level concepts as human do. *Deep learning* has shown a great success and promise in representation learning research in recent decades. In this thesis, we addressed a few of deficiencies in the current deep learning algorithms and demonstrated the strong empirical performance improvement in many applications of machine learning and computer vision.

In Part **??**, we have focused on research problems of learning low-level generic features in the framework of unsupervised learning. The unsupervised learning algorithms learn a generic representation of the data from a large number of unlabeled data, which is usually cheap and easy to obtain. The goal of unsupervised learning is often set as to learn patterns to represent the data well as there is no specific target task from the beginning. However, a good reconstruction of the training data doesn't necessarily lead to a good feature representation since the sensory data (e.g., image or audio signal) is very complex and highly variable, and contains a lot of irrelevant patterns or factors of variation, such as intrinsic data transformation or the noise from environment. One of the biggest criticism of the representation learning algorithms was

the difficulty of their use for non-experts outside the core deep learning research community. In Chapter II, we addressed this difficulty of learning by deriving an efficient strategy for training RBMs by exploring the connections between mixture models and RBMs and demonstrated both theoretically and empirically that the performance of sparse RBMs is similar to or significantly better than mixture models. In Chapter III and IV, we explored solutions to the problem of learning from noisy and complex sensory data. Specifically, we proposed a framework of learning invariant representation to local transformations, such as translation, rotation, or scale changes of visual sensory data or frequency or time-domain translations of auditory signals, using a variant of RBMs and sparse coding in Chapter III. In addition, we developed a point-wise gated Boltzmann machine in Chapter IV to learn useful representation from noisy data. The idea is to decompose the noisy components or factors of variation from the data with higher-order interaction by jointly learning and selecting features via bottom-up and top-down inference. The model showed effectiveness in learning useful representation when the data contains irrelevant patterns.

In Part **??**, we have developed an improved learning algorithms for structured output prediction. Although they have shown a great success, the current supervised deep learning methods, such as convolutional neural networks, may not be an optimal solution for the problems of structured output prediction. To develop an intelligent agent to make a prediction for complex output, one needs to move from modeling the many-to-one mapping, which indeed can be solved by the CNNs when sufficient training data is provided, towards the one-to-many or many-to-many mappings. In Chapter V, we developed a novel conditional generative model for face parts labeling by combining the best properties of the CRF for local consistency and the RBM for global shape prior. As a follow up, in Chapter VII, we took a step towards developing an end-to-end prediction system for such a complex output using stochastic convolutional neural network that can be trainable simply by backpropagation. We demonstrated an importance

of probabilistic inference for structured outputs through qualitative analysis as well as a strong image segmentation and labeling performance. Such a structured output prediction model can also be applied in multimodal representation learning problems. In Chapter VI, we proposed a new learning framework for multimodal representation learning based on the variation of information, whose learning procedure corresponds to estimating the conditional likelihoods of each data modality given the rest. By doing so, we can learn associations between data modalities better, and can effectively handle the case where there are some data modalities missing at testing time. We demonstrated an effectiveness of our proposed learning algorithm on image+text database for recognition with multimodal and unimodal scenario.

8.2 Future Work

For future research, I am going to delve into developing end-to-end systems for structured output prediction tasks using deep learning. Obtaining a sufficiently large volume of training data with fine-grained structured labels might be the biggest challenge for this problem. Instead, I am going to tackle the problem by using other resources (e.g., multimodal data, weakly or semi-supervised learning with coarse labels) and objective functions (e.g., variation of information, multi-task learning).

Multimodal representation learning for structured output prediction

Multiple sensory data are sometimes complementary to each other, providing useful information to improve pattern recognition performance. Learning generative models of multimodal data is very difficult due to the heterogeneity of the data. Fortunately, we have established an improved learning algorithm for generative models of multimodal data using variation of information in Chapter VI. On the other hand, learning classifier for structured output is challenging due to the lack of fully labeled training data. In such a case, the semi-supervised learning, where we train the model not only to predict

the output, but also to represent the input, can be employed to enhance better generalization to an unseen data at test time. We combine these two ideas to resolve the small training data issue in structured output prediction.

Semi-supervised multi-task learning with fine-grained structured labels

With an improved structured output prediction systems using deep generative models as a rich prior distribution (e.g., Chapter V), we aim to improve the performance of classical categorical prediction problems by jointly predicting the structured output labels and their higher-level concepts (e.g., categorical label). The structured output labels can be viewed as "attributes", and the structured prediction task is an intermediate goal for better categorical prediction in some sense. For example, the visual attributes, such as the existence of objects in the scene or textual description of the images (e.g., user tags from web images), help better predict the high-level concept of the examples, even when those attribute labels don't exist in the test time. We can view these structured output or attribute variables as auxiliary data which can be estimated from or complement to the original sensory data, and it is important to learn a good association among sensory input, attribute, and categorical output variables. To tackle this problem we develop a hybrid generative and discriminative deep networks for jointly predicting the (structured) attribute and categorical labels. For discriminative part between sensory input to output variables, we employ CNNs, a state-of-the-art visual recognition algorithm. For generative part that models the association between attribute and categorical variables, we use deep generative models such as DBN or DBM. Similarly to the approach illustrated in Chapter V, we use the pairwise potential across attribute variables. We train the model with multi-task objectives, where the training accuracy of structured and categorical variables are jointly maximized. To accommodate a large number of training data with missing fine-grained structured labels (e.g., we have millions of web images with user tags, but only few of them are labeled with

high-level visual concept in Flickr database), we train our algorithm in semi-supervised learning framework.

APPENDICES

APPENDIX A

Supplementary material of Chapter II

A.1 Corollary II.3

Corollary II.3. *The binary RBM (i.e., when the visible units are binary) with a softmax constraint on hidden units and the mixture of Bernoulli models are equivalent.*

Proof. Similarly to the proof of Proposition II.1, we prove by constructing the following conversions. Before that, we first define models; the Bernoulli mixture model (BMM) is defined as follows:

$$P(\mathbf{v}) = \sum_j \pi_j \prod_i \theta_{ij}^{v_i} (1 - \theta_{ij})^{1 - v_i} \tag{A.1}$$

The RBM with binary visible units, or equivalently, the binary RBM, is defined as follows:

$$P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \exp\left(-E(\mathbf{v}, \mathbf{h})\right) \tag{A.2}$$

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{i,j} v_i W_{ij} h_j - \sum_j b_j h_j \tag{A.3}$$

The softmax constraint can be applied on the hidden units, i.e., $\sum_j h_j \leq 1$, to formulate the binary-softmax RBM.

(1) From binary-softmax RBM to BMM($\boldsymbol{\theta}_k$): We begin by the decomposition using a chain rule:

$$P(\mathbf{v}, \mathbf{h}) = P(\mathbf{v}|\mathbf{h})P(\mathbf{h}),$$

where

$$P(\mathbf{h}) = \frac{1}{Z}\sum_{\mathbf{v}} \exp(-E(\mathbf{v}, \mathbf{h})).$$

Since there are only a finite number of hidden unit configurations, we can explicitly enumerate the prior probabilities:

$$P(h_j = 1) = \frac{\sum_{\mathbf{v}} \exp(-E(\mathbf{v}, h_j = 1))}{\sum_{j'}\sum_{\mathbf{v}} \exp(-E(\mathbf{v}, h_{j'} = 1))}$$

Defining $\tilde{\pi}_j \triangleq \sum_{\mathbf{v}} \exp(-E(\mathbf{v}, h_j = 1))$, we have $P(h_j = 1) = \frac{\tilde{\pi}_j}{\sum_{j'} \tilde{\pi}_{j'}} \triangleq \pi_j$, where $\tilde{\pi}_j$ can be calculated analytically as follows:

$$\begin{aligned}
\tilde{\pi}_j &= \sum_{\mathbf{v}} \exp(-E(\mathbf{v}, h_j = 1)) \\
&= \sum_{\mathbf{v}} \exp(\mathbf{v}^T\mathbf{w}_j + \mathbf{v}^T\mathbf{c} + b_j) \\
&= \exp(b_j)\prod_i (1 + \exp(W_{ij} + c_i))
\end{aligned}$$

Using this definition, we can show the following equality:

$$P(\mathbf{v}) = \sum_j \pi_j \prod_i (\text{sigm}(W_{ij} + c_i))^{v_i}(1 - \text{sigm}(W_{ij} + c_i))^{1-v_i}.$$

(2) From BMM($\boldsymbol{\theta}_k$) to binary-softmax RBM: We will show this by construction. Suppose we have the following BMM with $K + 1$ mixture components:

$$P(\mathbf{v}) = \sum_{j=0}^{K} \pi_j \prod_i \theta_{ij}^{v_i}(1 - \theta_{ij})^{1-v_i}. \tag{A.4}$$

We can convert from this BMM($\boldsymbol{\theta}_k$) to a binary-softmax RBM using the following transformations:

$$\begin{aligned} c_i &= \log \frac{\theta_{i0}}{1-\theta_{i0}} \\ W_{ij} &= \log \frac{\theta_{ij}}{1-\theta_{ij}} - c_i,\ j = 1, ..., K \\ b_j &= \log \frac{\pi_j}{\pi_0} + \sum_i \log(1-\theta_{ij}). \end{aligned} \tag{A.5}$$

Using the transformation equations, we can recover the conditional distributions of binary-softmax RBM as follows:

$$\begin{aligned} P(\mathbf{v}|h_j=1) &= \prod_i (\text{sigm}(W_{ij}+c_i))^{v_i}(1-\text{sigm}(W_{ij}+c_i))^{1-v_i} \\ &= \prod_i \left(\frac{\frac{\theta_{ij}}{1-\theta_{ij}}}{1+\frac{\theta_{ij}}{1-\theta_{ij}}}\right)^{v_i} \left(1-\frac{\frac{\theta_{ij}}{1-\theta_{ij}}}{1+\frac{\theta_{ij}}{1-\theta_{ij}}}\right)^{1-v_i} = \prod_i \theta_{ij}^{v_i}(1-\theta_{ij})^{1-v_i} \\ P(h_j=1|\mathbf{v}) &= \frac{\pi_j \prod_i \theta_{ij}^{v_i}(1-\theta_{ij})^{1-v_i}}{\sum_{j'=0}^K \pi_{j'} \prod_i \theta_{ij'}^{v_i}(1-\theta_{ij'})^{1-v_i}} \\ &= \frac{\pi_j \exp\left(\sum_i v_i \log \frac{\theta_{ij}}{1-\theta_{ij}}\right) \exp\left(\sum_i \log(1-\theta_{ij})\right)}{\sum_{j'=0}^K \pi_{j'} \exp\left(\sum_i v_i \log \frac{\theta_{ij'}}{1-\theta_{ij'}}\right) \exp\left(\sum_i \log(1-\theta_{ij'})\right)} \\ &= \frac{\pi_j \exp\left(\sum_i v_i W_{ij}\right) \exp\left(b_j - \log\frac{\pi_j}{\pi_0}\right)}{\sum_{j'=0}^K \pi_{j'} \exp\left(\sum_i v_i W_{ij'}\right) \exp\left(b_{j'} - \log\frac{\pi_{j'}}{\pi_0}\right)} \\ &= \frac{\exp\left(\sum_i v_i W_{ij} + b_j\right)}{1+\sum_{j'=1}^K \exp\left(\sum_i v_i W_{ij'} + b_{j'}\right)} = \frac{\exp\left(\mathbf{w}_j^T \mathbf{v} + b_j\right)}{1+\sum_{j'=1}^K \exp\left(\mathbf{w}_{j'}^T \mathbf{v} + b_{j'}\right)} \end{aligned} \tag{A.6, A.7}$$

Therefore, a BMM can be converted to binary RBM with a softmax constraint. □

A.2 Corollary II.4

Corollary II.4. *GMM($\mathbf{0}, \boldsymbol{\Sigma}_k$) with arbitrary covariance matrices and the factored 3-way RBM [104] with a softmax constraint on hidden units are equivalent.*

Proof. We first define models; the GMM with zero mean and arbitrary covariance ma-

trices $\mathbf{\Sigma}_k$ is defined as follows:

$$P(\mathbf{v}) = \sum_{k=0}^{K} \pi_k \mathcal{N}(\mathbf{v}; \mathbf{0}, \mathbf{\Sigma}_k) \tag{A.8}$$

We also consider a Factored 3-way RBM with softmax constraint on the hidden units (we will call it a *Factored-softmax RBM*). The energy function of the Factored-softmax RBM is given as follows:

$$P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \exp(-E(\mathbf{v}, \mathbf{h})) \tag{A.9}$$

$$E(\mathbf{v}, \mathbf{h}) = -\frac{1}{2} \sum_{k,f=1}^{K,F} h_k P_{kf} \Big(\sum_{i=1}^{D} v_i C_{if} \Big)^2 - \sum_{k=1}^{K} b_k h_k \tag{A.10}$$

$$= -\frac{1}{2} \sum_{f=1}^{F} \mathbf{h}^T P_f \Big(\mathbf{v}^T C_f \Big)^2 - \mathbf{b}^T \mathbf{h} \tag{A.11}$$

with the softmax constraint $\sum_k h_k \leq 1$. P_f, C_f are f-th columns of $\mathbf{P}$, $\mathbf{C}$, respectively. The conditional probabilities are derived as follows:

$$P(\mathbf{v}|\mathbf{h}) = \mathcal{N}(\mathbf{v}; \mathbf{0}, -\Big[\mathbf{C}\text{diag}(\mathbf{h}^T\mathbf{P})\mathbf{C}^T\Big]^{-1}) \tag{A.12}$$

$$P(h_k = 1|\mathbf{v}) = \frac{\exp(\frac{1}{2}\sum_f P_{kf} \Big(\sum_{i=1}^{D} v_i C_{if} \Big)^2 + b_k)}{1 + \sum_{k'=1}^{K} \exp(\frac{1}{2}\sum_f^F P_{k'f} \Big(\sum_{i=1}^{D} v_i C_{if} \Big)^2 + b_{k'})}, \tag{A.13}$$

(1) From factored-softmax RBM to GMM($\mathbf{0}, \mathbf{\Sigma}_k$): We enumerate the prior probability of hidden configurations as follows:

$$P(h_k = 1) = \frac{\int d\mathbf{v} \exp(-E(\mathbf{v}, h_k = 1))}{\sum_{k'=1}^{K} \int d\mathbf{v} \exp(-E(\mathbf{v}, h_{k'} = 1))} \tag{A.14}$$

Defining $\tilde{\pi}_k = \int d\mathbf{v} \exp(-E(\mathbf{v}, h_k = 1))$, we have $P(h_k = 1) = \frac{\tilde{\pi}_k}{\sum_{k'=1}^{K} \tilde{\pi}_{k'}} \triangleq \pi_k$ and $\tilde{\pi}_k$

can be analytically calculated as follows:

$$\begin{aligned}\tilde{\pi}_k &= \int d\mathbf{v} \exp(-E(\mathbf{v}, h_k = 1)) \\ &= \int d\mathbf{v} \exp(\frac{1}{2}\sum_{f=1}^{F} P_{kf}\mathbf{v}^T C_f C_f^T \mathbf{v} + b_k) \\ &= (\sqrt{2\pi})^D |\mathbf{\Sigma}_k|^{\frac{1}{2}} \exp(b_k)\end{aligned}$$

where $\mathbf{\Sigma}_k = -(C \text{ diag}(P_{k\cdot})C^T)^{-1}$. Using this definition, we have the following equality:

$$P(\mathbf{v}) = \sum_{k=1}^{N} P(\mathbf{v}|h_k^c = 1)P(h_k^c = 1) = \sum_{k=1}^{K_c} \pi_k \mathcal{N}(\mathbf{v}; \mathbf{0}, \mathbf{\Sigma}_k). \tag{A.15}$$

This completes the first part of the proof that Factored-softmax RBM can be written as a GMM with zero mean and an covariance matrix $\mathbf{\Sigma}_k$.

(2) From GMM($\mathbf{0}, \mathbf{\Sigma}_k$) to factored-softmax RBM: Now suppose that we have the following GMM with $K + 1$ components:

$$P(\mathbf{v}) = \sum_{k=0}^{K} \pi_k \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_k). \tag{A.16}$$

For simplicity, let's assume $\mathbf{\Sigma}_k$'s are positive definite. Then, we can use singular value decomposition to decompose $\mathbf{\Sigma}_k^{-1}$ into

$$\mathbf{\Sigma}_k^{-1} = U_k \Lambda_k U_k^T = \sum_{i=1}^{D} (U_k)_i (U_k)_i^T \lambda_{k,i}. \tag{A.17}$$

Let $F = (K + 1) \times D$. We define $\mathbf{C}$ with dimension $D \times F$ and $\mathbf{P}$ with dimension $F \times (K + 1)$ as follows:

$$\begin{aligned}C_{kD+i} &= (U_k)_i, \\ F_{kD+i,k} &= -\lambda_{k,i}, i = 1, ..., D, k = 1, ..., K\end{aligned}$$

where C_j is a j-th column of $\mathbf{C}$ and $F_{i,j}$ is a (i,j)-th component of $\mathbf{F}$. The components that were not specified in the above equations are set to 0. Finally, by defining $b_k = \log(\pi_k \left|\mathbf{\Sigma}_k\right|^{-\frac{1}{2}}) - \log(\pi_0 \left|\mathbf{\Sigma}_0\right|^{-\frac{1}{2}})$, we can write the posterior probability of hidden units given the visible units as:

$$
\begin{aligned}
P(h_k = 1|\mathbf{v}) &= \frac{\pi_k \mathcal{N}(\mathbf{v}; \mathbf{0}, \mathbf{\Sigma}_k)}{\sum_{k'=0}^{K} \pi_{k'} \mathcal{N}(\mathbf{v}; \mathbf{0}, \mathbf{\Sigma}_{k'})} \\
&= \frac{\pi_k \left|\mathbf{\Sigma}_k\right|^{-\frac{1}{2}} \exp\left(\frac{1}{2} \sum_{f=1}^{F} \mathbf{v}^T C_f C_f^T \mathbf{v} P_{kf}\right)}{\sum_{k'=0}^{K} \pi_{k'} \left|\mathbf{\Sigma}_{k'}\right|^{-\frac{1}{2}} \exp\left(\frac{1}{2} \sum_{f=1}^{F} \mathbf{v}^T C_f C_f^T \mathbf{v} P_{k'f}\right)} \\
&= \frac{\pi_k \left|\mathbf{\Sigma}_k\right|^{-\frac{1}{2}} \exp\left(\frac{1}{2} \sum_{f=1}^{F} \mathbf{v}^T C_f C_f^T \mathbf{v} P_{kf}\right)}{\sum_{k'=0}^{K} \pi_{k'} \left|\mathbf{\Sigma}_{k'}\right|^{-\frac{1}{2}} \exp\left(\frac{1}{2} \sum_{f=1}^{F} \mathbf{v}^T C_f C_f^T \mathbf{v} P_{k'f}\right)} \\
&= \frac{\exp\left(\frac{1}{2} \sum_{f=1}^{F} P_{kf} (\mathbf{v}^T C_f)^2 + b_k\right)}{1 + \sum_{k'=1}^{K} \exp\left(\frac{1}{2} \sum_{f=1}^{F} P_{k'f} (\mathbf{v}^T C_f)^2 + b_{k'}\right)}
\end{aligned}
$$

Therefore, we recover the posterior of the Factored-softmax RBM in Equation (A.13), and this completes the proof.

□

APPENDIX B

Supplementary material of Chapter IV

B.1 Derivation of Equations

B.1.1 Derivation of Equations (4.3)–(4.5)

The energy function of the PGBM is given as follows:

$$E^{\mathrm{U}}(\mathbf{v},\mathbf{z},\mathbf{h}) = -\sum_{r=1}^{R}\sum_{i=1}^{D}\sum_{k=1}^{K_r}(z_i^r v_i)\,W_{ik}^r h_k^r - \sum_{r=1}^{R}\sum_{k=1}^{K_r} b_k^r h_k^r - \sum_{r=1}^{R}\sum_{i=1}^{D}(z_i^r v_i)\,c_i^r \tag{4.1}$$

$$= -\sum_{r=1}^{R}(\mathbf{z}^r \odot \mathbf{v})^T \mathbf{W}^r \mathbf{h}^r - \sum_{r=1}^{R}(\mathbf{b}^r)^T\mathbf{h}^r - \sum_{r=1}^{R}(\mathbf{c}^r)^T(\mathbf{z}^r \odot \mathbf{v}), \tag{4.2}$$

$$\text{s.t.} \quad \sum_{r=1}^{R} z_i^r = 1,\ i = 1,\cdots,D.$$

where the operator $\odot$ denotes element-wise multiplication, i.e., $(\mathbf{z}^r \odot \mathbf{v})_i = z_i^r v_i$. The visible, hidden, and switch units are conditionally independent given the other two

types of units and the conditional probabilities can be derived as follows:

$$
\begin{aligned}
P\left(h_k^r = 1 \mid \mathbf{z}, \mathbf{v}\right) &= \frac{\exp\left(\sum_{i=1}^{D}\left(z_i^r v_i\right) W_{ik}^r + b_k^r\right)}{\sum_{h_k^r \in \{0,1\}} \exp\left(\sum_{i=1}^{D}\left(z_i^r v_i\right) W_{ik}^r h_k^r + b_k^r h_k^r\right)} \\
&= \frac{\exp\left(\sum_{i=1}^{D}\left(z_i^r v_i\right) W_{ik}^r + b_k^r\right)}{1 + \exp\left(\sum_{i=1}^{D}\left(z_i^r v_i\right) W_{ik}^r + b_k^r\right)} \\
&= \sigma\left(\sum_{i=1}^{D}\left(z_i^r v_i\right) W_{ik}^r + b_k^r\right) = \sigma\left(\left(\mathbf{z}^r \odot \mathbf{v}\right)^T \mathbf{W}_{\cdot k}^r + b_k^r\right) \qquad (4.3) \\
P\left(v_i = 1 \mid \mathbf{z}, \mathbf{h}\right) &= \frac{\exp\left(\sum_{r=1}^{R}\sum_{k=1}^{K_r} z_i^r W_{ik}^r h_k^r + \sum_{r=1}^{R} z_i^r c_i^r\right)}{\sum_{v_i \in \{0,1\}} \exp\left(\sum_{r=1}^{R}\sum_{k=1}^{K_r}\left(z_i^r v_i\right) W_{ik}^r h_k^r + \sum_{r=1}^{R}\left(z_i^r v_i\right) c_i^r\right)} \\
&= \frac{\exp\left(\sum_{r=1}^{R}\sum_{k=1}^{K_r} z_i^r W_{ik}^r h_k^r + \sum_{r=1}^{R} z_i^r c_i^r\right)}{1 + \exp\left(\sum_{r=1}^{R}\sum_{k=1}^{K_r} z_i^r W_{ik}^r h_k^r + \sum_{r=1}^{R} z_i^r c_i^r\right)} \\
&= \sigma\left(\sum_r z_i^r\Big(\sum_k W_{ik}^r h_k^r + c_i^r\Big)\right) = \sigma\left(\sum_r z_i^r\left(\mathbf{W}_{i\cdot}^r \mathbf{h}^r + c_i^r\right)\right) \qquad (4.4) \\
P\left(z_i^r = 1 \mid \mathbf{v}, \mathbf{h}\right) &= \frac{\exp\left(\sum_{k=1}^{K_r} v_i W_{ik}^r h_k^r + v_i c_i^r\right)}{\sum_r \exp\left(\sum_{k=1}^{K_r}\left(z_i^r v_i\right) W_{ik}^r h_k^r + \left(z_i^r v_i\right) c_i^r\right)} \\
&= \frac{\exp\left(v_i\Big(\sum_{k=1}^{K_r} W_{ik}^r h_k^r + c_i^r\Big)\right)}{\sum_r \exp\left(\left(z_i^r v_i\right)\Big(\sum_{k=1}^{K_r} W_{ik}^r h_k^r + c_i^r\Big)\right)} \\
&= \frac{\exp\left(v_i\left(\mathbf{W}_{i\cdot}^r \mathbf{h}^r + c_i^r\right)\right)}{\sum_s \exp\left(v_i\left(\mathbf{W}_{i\cdot}^s \mathbf{h}^s + c_i^s\right)\right)} \qquad (4.5)
\end{aligned}
$$

B.1.2 Derivation of Equations (4.7)–(4.8)

We present the supervised PGBM with two mixture components, where we assign the first component to be task-relevant. The energy function is defined as follows:

$$
E^{\mathrm{S}}(\mathbf{v}, \mathbf{z}, \mathbf{h}, \mathbf{y}) = E^{\mathrm{U}}(\mathbf{v}, \mathbf{z}, \mathbf{h}) - \mathbf{y}^T \mathbf{U} \mathbf{h}^1 - \mathbf{d}^T \mathbf{y} \qquad (4.6)
$$

subject to $z_i^1 + z_i^2 = 1,\ i = 1, \cdots, D$. The label vector $\mathbf{y} \in \{0,1\}^L$ is in the 1-of-L representation. $\mathbf{U} \in \mathbb{R}^{L \times K_1}$ is the weight matrix between the task-relevant hidden units and the label units, and $\mathbf{d}$ is the label bias vector. The conditional probabilities can be

derived as follows:

$$
\begin{aligned}
P\left(h_k^1 = 1 \mid \mathbf{z}, \mathbf{v}, \mathbf{y}\right) &= \frac{\exp\left(\sum_{i=1}^{D}\left(z_i^1 v_i\right) W_{ik}^1 + b_k^1 + \sum_{l=1}^{L} y_l U_{lk}\right)}{\sum_{h_k^1 \in \{0,1\}} \exp\left(\sum_{i=1}^{D}\left(z_i^1 v_i\right) W_{ik}^1 h_k^1 + b_k^1 h_k^1 + \sum_{l=1}^{L} y_l U_{lk} h_k^1\right)} \\
&= \frac{\exp\left(\sum_{i=1}^{D}\left(z_i^1 v_i\right) W_{ik}^1 + b_k^1 + \sum_{l=1}^{L} y_l U_{lk}\right)}{1 + \exp\left(\sum_{i=1}^{D}\left(z_i^1 v_i\right) W_{ik}^1 + b_k^1 + \sum_{l=1}^{L} y_l U_{lk}\right)} \\
&= \sigma\left(\sum_{i=1}^{D}\left(z_i^1 v_i\right) W_{ik}^1 + b_k^1 + \sum_{l=1}^{L} y_l U_{lk}\right) \\
&= \sigma\left((\mathbf{z}^1 \odot \mathbf{v})^T \mathbf{W}_{\cdot k}^1 + b_k^1 + \mathbf{U}_{\cdot k}^T \mathbf{y}\right) \qquad (4.7)
\end{aligned}
$$

$$
P\left(y_l = 1 \mid \mathbf{h}^1\right) = \frac{\exp\left(\sum_{k=1}^{K_1} U_{lk} h_k^1 + d_l\right)}{\sum_{s=1}^{L} \exp\left(\sum_{k=1}^{K_1} U_{sk} h_k^1 + d_s\right)} = \frac{\exp\left(\mathbf{U}_{l\cdot}\mathbf{h}^1 + d_l\right)}{\sum_s \exp\left(\mathbf{U}_{s\cdot}\mathbf{h}^1 + d_s\right)}. \qquad (4.8)
$$

B.1.3 Derivation of Equations (4.19)–(4.21)

The energy function of the CPGBM with R components is written as follows:

$$
\begin{aligned}
E(\mathbf{v}, \mathbf{z}, \mathbf{h}) = -\sum_{r=1}^{R}\left[\sum_{l=1}^{L}\sum_{m,n} z_{m,n}^r v_{m,n}^l c_l^r - \sum_{k=1}^{K_r}\sum_{i,j} h_{i,j}^{r,k}\left(\sum_{l=1}^{L}(\widetilde{\mathbf{W}}^{r,k,l} * (\mathbf{z}^r \odot \mathbf{v}^l))_{i,j} + b_k^r\right)\right] \\
\text{s.t.} \quad \sum_{r=1}^{R} z_{m,n}^r = 1, \ m, n = 1, \cdots, N_V \qquad (4.17)
\end{aligned}
$$

where the second term can be written as follows:

$$
\begin{aligned}
&\sum_{k=1}^{K_r}\sum_{i,j} h_{i,j}^{r,k}\left(\sum_{l=1}^{L}(\widetilde{\mathbf{W}}^{r,k,l} * (\mathbf{z}^r \odot \mathbf{v}^l))_{i,j} + b_k^r\right) \\
&= \sum_{k=1}^{K_r}\sum_{l=1}^{L}\sum_{i,j,m,n} h_{i,j}^{r,k} W_{m,n}^{r,k,l} z_{i+m-1,j+n-1}^r v_{i+m-1,j+n-1}^l + \sum_{k=1}^{K_r}\sum_{i,j} h_{i,j}^{r,k} b_k^r \\
&= \sum_{l=1}^{L}\sum_{m,n} z_{m,n}^r v_{m,n}^l \sum_{k=1}^{K_r}(\mathbf{W}^{r,k,l} * \mathbf{h}^{r,k})_{m,n} + \sum_{k=1}^{K_r}\sum_{i,j} h_{i,j}^{r,k} b_k^r
\end{aligned}
$$

Using above variations of the energy function, the conditional probabilities of hidden, switch, and visible units given the other two types of variables for CPGBM can be derived as follows:

$$P(h_{i,j}^{r,k}=1|\mathbf{v},\mathbf{z}) = \frac{\exp\left(\sum_{l=1}^{L}(\widetilde{\mathbf{W}}^{r,k,l} * (\mathbf{z}^r \odot \mathbf{v}^l))_{i,j} + b_k^r\right)}{1+\exp\left(\sum_{l=1}^{L}(\widetilde{\mathbf{W}}^{r,k,l} * (\mathbf{z}^r \odot \mathbf{v}^l))_{i,j} + b_k^r\right)}$$

$$= \sigma\left(\sum_{l=1}^{L}(\widetilde{\mathbf{W}}^{r,k,l} * (\mathbf{z}^r \odot \mathbf{v}^l))_{i,j} + b_k^r\right) \tag{4.19}$$

$$P(z_{m,n}^r=1|\mathbf{v},\mathbf{h}) = \frac{\exp\left(\sum_l v_{m,n}^l(\sum_k(\mathbf{W}^{r,k,l} * \mathbf{h}^{r,k})_{m,n} + c_l^r)\right)}{\sum_s \exp\left(\sum_l v_{m,n}^l(\sum_k(\mathbf{W}^{s,k,l} * \mathbf{h}^{s,k})_{m,n} + c_l^s)\right)} \tag{4.20}$$

$$P(v_{m,n}^l=1|\mathbf{h},\mathbf{z}) = \sigma\left(\sum_r z_{m,n}^r\left[\sum_k(\mathbf{W}^{r,k,l} * \mathbf{h}^{r,k})_{m,n} + c_l^r\right]\right) \tag{4.21}$$

APPENDIX C

Supplementary material of Chapter V

C.1 Derivation of Algorithm 2

We derive the mean-field updates described in Algorithm 2 by solving the following optimization problem:

$$Q^* = \underset{Q(\mathcal{Y},\mathbf{h})}{\arg\min}\, \mathrm{KL}\left(Q(\mathcal{Y},\mathbf{h}) \| P_\theta(\mathcal{Y},\mathbf{h}|\mathcal{X})\right) \tag{C.1}$$

subject to $Q(\mathcal{Y},\mathbf{h}) = \prod_{s\in\mathcal{V},k} Q(\mathbf{y}_s)Q(h_k)$. For multinomial output variables and binary hidden variables, we define $\gamma_k = Q(h_k = 1)$ and $\mu_{sl} = Q(y_s = l)$, subject to $\sum_{l=1}^{L} \mu_{sl} = 1$. The KL divergence can be written as follows:

$$\mathrm{KL}\left(Q(\mathcal{Y},\mathbf{h}) \| P_\theta(\mathcal{Y},\mathbf{h}|\mathcal{X})\right) = \sum_{\mathcal{Y},\mathbf{h}} Q(\mathcal{Y},\mathbf{h}) \left(\log Q(\mathcal{Y},\mathbf{h}) - \log P_\theta(\mathcal{Y},\mathbf{h}|\mathcal{X})\right) \tag{C.2}$$

where the first term can be further decomposed into

$$\begin{aligned}
&\sum_{\mathcal{Y},\mathbf{h}} Q(\mathcal{Y},\mathbf{h}) \log Q(\mathcal{Y},\mathbf{h}) \\
&= \sum_{s,l} \sum_{y_{sl}} Q(y_{sl}) \log Q(y_{sl}) + \sum_{k} \sum_{h_k \in \{0,1\}} Q(h_k) \log Q(h_k) \qquad \text{(C.3)} \\
&= \sum_{s,l} \mu_{sl} \log \mu_{sl} + \sum_{k} \gamma_k \log \gamma_k + (1-\gamma_k) \log(1-\gamma_k) \qquad \text{(C.4)}
\end{aligned}$$

The second term of Equation (C.1) can be written as follows:

$$\sum_{\mathcal{Y},\mathbf{h}} Q(\mathcal{Y},\mathbf{h}) \log P_\theta(\mathcal{Y},\mathbf{h}|\mathcal{X}) = \sum_{\mathcal{Y},\mathbf{h}} Q(\mathcal{Y},\mathbf{h}) \left(\log \exp\left(-E_{\text{gloc}}(\mathcal{Y},\mathcal{X},\mathbf{h})\right) - \log Z(\mathcal{X})\right) \quad \text{(C.5)}$$

Since $Z(\mathcal{X}) = \sum_{\mathcal{Y},\mathbf{h}} \exp\left(-E_{\text{gloc}}(\mathcal{Y},\mathcal{X},\mathbf{h})\right)$ is independent of μ_{sl}'s and γ_k's, we can ignore the second term for optimization. Before we proceed, let's remind the energy function of the model: From Equation (5.6), $E_{\text{gloc}}(\mathcal{Y},\mathcal{X},\mathbf{h}) = E_{\text{crf}}(\mathcal{Y},\mathcal{X}) + E_{\text{rbm}}(\mathcal{Y},\mathbf{h})$ where

$$E_{\text{crf}}(\mathcal{Y},\mathcal{X}) = -\sum_{s=1}^{S}\sum_{l=1}^{L} y_{sl} \sum_{n=1}^{N^2} p_{sn} \sum_{d=1}^{D_n} \Gamma_{ndl} x_{sd} - \sum_{(i,j)} \sum_{l,l'=1}^{L} \sum_{e=1}^{D_e} y_{il} y_{jl'} \Psi_{ll'e} x_{ije}, \quad \text{(5.2),(5.8)}$$

$$E_{\text{rbm}}(\mathcal{Y},\mathbf{h}) = -\sum_{r=1}^{R^2}\sum_{l=1}^{L}\sum_{k=1}^{K}\sum_{s=1}^{S} p_{rs} y_{sl} W_{rlk} h_k - \sum_{k=1}^{K} b_k h_k - \sum_{r=1}^{R^2}\sum_{l=1}^{L} c_{rl} \sum_{s=1}^{S} p_{rs} y_{sl}. \quad \text{(5.7)}$$

Then, the first term of Equation (C.5) can be written as follows:

$$\begin{aligned}
&\sum_{\mathcal{Y},\mathbf{h}} Q(\mathcal{Y},\mathbf{h}) \left(\log \exp\left(-E_{\text{gloc}}(\mathcal{Y},\mathcal{X},\mathbf{h})\right)\right) = -\sum_{\mathcal{Y},\mathbf{h}} Q(\mathcal{Y},\mathbf{h}) E_{\text{gloc}}(\mathcal{Y},\mathcal{X},\mathbf{h}) \\
&= \sum_{s,l,n,d} \mu_{sl} p_{sn} \Gamma_{ndl} x_{sd} + \sum_{(i,j),l,l',e} \mu_{il} \mu_{jl'} \Psi_{ll'e} x_{ije} + \sum_{s,l,r} \mu_{sl} p_{rs} \left(\sum_k W_{rlk} \gamma_k + c_{rl}\right) + \sum_k b_k \gamma_k \qquad \text{(C.6)}
\end{aligned}$$

Finally, the optimization problem in Equation (C.1) can be written as follows:

$$Q^* = \arg\min_{\mu_{sl},\gamma_k} \Bigg[- \sum_{s,l,n,d} \mu_{sl} p_{sn} \Gamma_{ndl} x_{sd} - \sum_{(i,j),l,l',e} \mu_{il} \mu_{jl'} \Psi_{ll'e} x_{ije} - \sum_k b_k \gamma_k - \tag{C.7}$$
$$\sum_{s,l,r} \mu_{sl} p_{rs} \left(\sum_k W_{rlk} \gamma_k + c_{rl} \right) + \sum_{s,l} \mu_{sl} \log \mu_{sl} + \sum_k \gamma_k \log \gamma_k + (1 - \gamma_k) \log(1 - \gamma_k) \Bigg]$$

subject to $\sum_l \mu_{sl} = 1$. Taking derivatives with respect to μ_{sl} and γ_k, we obtain

$$\frac{\partial \mathrm{KL}}{\partial \mu_{sl}} = \log \mu_{sl} + 1 - \sum_{n,d} p_{sn} \Gamma_{ndl} x_{sd} - \sum_{(s,j)l',e} \mu_{jl'} \Psi_{ll'e} x_{sje} - \sum_r \left(\sum_k W_{rlk} \gamma_k + c_{rl} \right) \tag{C.8}$$

$$\frac{\partial \mathrm{KL}}{\partial \gamma_k} = \log \frac{\gamma_k}{1 - \gamma_k} - \sum_{s,l,r} \mu_{sl} p_{rs} W_{rlk} - b_k = 0 \tag{C.9}$$

Combining Equation (C.8) with the following multinomial unit constraint $\sum_l \mu_{sl} = 1$, we have $\mu_{sl} = \frac{\tilde{\mu}_{sl}}{\sum_{l'} \tilde{\mu}_{sl'}}$, where

$$\tilde{\mu}_{sl} = \exp \Big(\underbrace{\sum_{n,d} p_{sn} \Gamma_{ndl} x_{sd}}_{f_{sl}^{\mathrm{node}}} + \underbrace{\sum_{(s,j),l',e} \mu_{jl'} \Psi_{ll'e} x_{sje}}_{f_{sl}^{textedge}} + \underbrace{\sum_r p_{rs} \Big(\sum_k W_{rlk} \gamma_k + c_{rl} \Big)}_{f_{sl}^{\mathrm{rbm}}} \Big) \tag{C.10}$$

$$\gamma_k = \sigma \Big(\sum_{s,l,r} \mu_{sl} p_{rs} W_{rlk} + b_k \Big) \tag{C.11}$$

By formulating the iterative block update equations with respect to μ_{sl} and γ_k, we complete the derivation of Algorithm 2.

APPENDIX D

Supplementary material of Chapter VI

D.1 Derivation of Equation (6.4)

The NLL objective function can be written as

$$
\begin{aligned}
2\mathcal{L}^{\mathrm{NLL}}(\theta) &= -2\mathbb{E}_{P_{\mathcal{D}}}\Big[\log P_\theta(X,Y)\Big] \\
&= -\mathbb{E}_{P_{\mathcal{D}}}\Big[\log P_\theta(X|Y) + \log P_\theta(Y)\Big] - \mathbb{E}_{P_{\mathcal{D}}}\Big[\log P_\theta(Y|X) + \log P_\theta(X)\Big] \\
&= -\mathbb{E}_{P_{\mathcal{D}}}\Big[\log P_\theta(X|Y) + \log P_\theta(Y|X)\Big] - \mathbb{E}_{P_{\mathcal{D}}}\Big[\log P_\theta(X) + \log P_\theta(Y)\Big] \\
&= \mathcal{L}^{\mathrm{VI}}(\theta) - \mathbb{E}_{P_{\mathcal{D}}}\Big[\log P_\theta(X)\Big] - \mathbb{E}_{P_{\mathcal{D}}}\Big[\log P_\theta(Y)\Big] \qquad \text{(D.1)} \\
&= \mathcal{L}^{\mathrm{VI}}(\theta) + \underbrace{\mathbb{E}_{P_{\mathcal{D}}}\left[\log \frac{P_{\mathcal{D}}(X)}{P_\theta(X)}\right]}_{\mathrm{KL}(P_{\mathcal{D}}(X)\|P_\theta(X))} + \underbrace{\mathbb{E}_{P_{\mathcal{D}}}\left[\log \frac{P_{\mathcal{D}}(Y)}{P_\theta(Y)}\right]}_{\mathrm{KL}(P_{\mathcal{D}}(Y)\|P_\theta(Y))} \qquad \text{(D.2)} \\
&\qquad \underbrace{-\mathbb{E}_{P_{\mathcal{D}}}\Big[\log P_{\mathcal{D}}(X)\Big] - \mathbb{E}_{P_{\mathcal{D}}}\Big[\log P_{\mathcal{D}}(Y)\Big]}_{C_1} \\
&= \mathcal{L}^{\mathrm{VI}}(\theta) + \mathrm{KL}\left(P_{\mathcal{D}}(X)\|P_\theta(X)\right) + \mathrm{KL}\left(P_{\mathcal{D}}(Y)\|P_\theta(Y)\right) + C_1 \qquad \text{(D.3)}
\end{aligned}
$$

where Equation (D.1) holds by the definition of $\mathcal{L}^{\mathrm{VI}}(\theta)$. Note that C_1 is independent of θ. Similarly, we can rewrite the MinVI objective as

$$\mathcal{L}^{\mathrm{VI}}(\theta) = -\mathbb{E}_{P_{\mathcal{D}}}\Big[\log P_\theta(X|Y) + \log P_\theta(Y|X)\Big] \tag{D.4}$$

$$= \mathbb{E}_{P_{\mathcal{D}}}\left[\log \frac{P_{\mathcal{D}}(X|Y)}{P_\theta(X|Y)}\right] + \mathbb{E}_{P_{\mathcal{D}}}\left[\log \frac{P_{\mathcal{D}}(Y|X)}{P_\theta(Y|X)}\right] \tag{D.5}$$

$$\underbrace{-\mathbb{E}_{P_{\mathcal{D}}}\Big[\log P_{\mathcal{D}}(X|Y)\Big] - \mathbb{E}_{P_{\mathcal{D}}}\Big[\log P_{\mathcal{D}}(Y|X)\Big]}_{C_2}$$

where in Equation (D.5), we have

$$\mathbb{E}_{P_{\mathcal{D}}}\left[\log \frac{P_{\mathcal{D}}(X|Y)}{P_\theta(X|Y)}\right] = \sum_y P_{\mathcal{D}}(y)\mathbb{E}_{P_{\mathcal{D}}(X|y)}\left[\log \frac{P_{\mathcal{D}}(X|y)}{P_\theta(X|y)}\right] \tag{D.6}$$

$$= \mathbb{E}_{P_{\mathcal{D}}(Y)}\Big[\mathrm{KL}\left(P_{\mathcal{D}}(X|Y)\|P_\theta(X|Y)\right)\Big] \tag{D.7}$$

Finally, we have

$$\mathcal{L}^{\mathrm{VI}}(\theta) = \mathbb{E}_{P_{\mathcal{D}}(X)}\Big[\mathrm{KL}\left(P_{\mathcal{D}}(Y|X)\|P_\theta(Y|X)\right)\Big] + \mathbb{E}_{P_{\mathcal{D}}(Y)}\Big[\mathrm{KL}\left(P_{\mathcal{D}}(X|Y)\|P_\theta(X|Y)\right)\Big] + C_2. \tag{D.8}$$

C_2 is independent of θ and by setting $C = C_1 + C_2$, we derive the Equation (6.4).

D.2 Proof of Theorem VI.1

Proposition D.1 ([7, 10])**.** *Assume that $\mathcal{X}$ is a finite state space. Let T_n and T be irreducible transition matrices that have stationary distributions $\pi_n(X)$ and $\pi(X)$, respectively, where $\pi(X) = P_{\mathcal{D}}(X)$ is a data-generating distribution of X. If T_n converges to T entrywise, then $\pi_n(X)$ converges to $P_{\mathcal{D}}(X)$ entrywise.*

Proof. Let $|\mathcal{X}|$ be the number of states of variable X. For simplicity, we denote $\pi = \pi(X)$ and $\pi_n = \pi_n(X)$. Since the transition matrix T is irreducible, the stationary

distribution π is unique. In other words, π is characterized by the following equations:

$$\sum_{k=1}^{|\mathcal{X}|} T_{j,k}\pi_k = \pi_j, \forall j \in \{1, \cdots, |\mathcal{X}|\} \tag{D.9}$$

$$\sum_{k=1}^{|\mathcal{X}|} \pi_k = 1, \tag{D.10}$$

$$\sum_{j=1}^{|\mathcal{X}|} T_{j,k} = 1, \forall j \in \{1, \cdots, |\mathcal{X}|\}. \tag{D.11}$$

Here, (D.11) holds since T is a transition matrix. It is easy to see that one of the equations from (D.9) is redundant; for example, $\sum_{k=1}^{|\mathcal{X}|} T_{|\mathcal{X}|,k}\pi_k = \pi_{|\mathcal{X}|}$ can be recovered from other equations of (D.9), (D.10), and (D.11). Therefore, we can combine the above system of linear equations in an equivalent form as follows:

$$\underbrace{\begin{bmatrix} T_{1,1}-1 & T_{1,2} & \cdots & T_{1,|\mathcal{X}|} \\ T_{2,1} & T_{2,2}-1 & \cdots & T_{2,|\mathcal{X}|} \\ \vdots & \cdots & \cdots & \vdots \\ T_{|\mathcal{X}|-1,1} & \cdots & \cdots & T_{|\mathcal{X}|-1,|\mathcal{X}|} \\ 1 & 1 & \cdots & 1 \end{bmatrix}}_{=\widetilde{T}} \pi = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1, \end{bmatrix} \tag{D.12}$$

where $\widetilde{T}$ is defined accordingly. Since π exists and is unique, the null space of $\widetilde{T}$ must be empty and $\widetilde{T}$ is invertible. Now we have

$$\pi = \widetilde{T}^{-1} \begin{bmatrix} 0 & 0 & \cdots & 1 \end{bmatrix}^\top \tag{D.13}$$

and similarly,

$$\pi_n = \widetilde{T}_n^{-1} \begin{bmatrix} 0 & 0 & \cdots & 1 \end{bmatrix}^\top. \tag{D.14}$$

Since T_n converges to T entrywise, $\widetilde{T}_n$ converges to $\widetilde{T}$ entrywise, and $\widetilde{T}_n^{-1}$ also converges to $\widetilde{T}^{-1}$ entrywise. Therefore, we conclude π_n converges to $\pi = P_{\mathcal{D}}(X)$ entrywise [47].

Since on a finite-dimensional space, all norms are equivalent [5], the above convergence, in fact, holds for any norm. □

Now, we provide a proof of Theorem VI.1.

Proof of Theorem VI.1. To prove the convergence of marginal distributions, it is sufficient to show the convergence of transition operators. Since $|\mathcal{X}|$ and $|\mathcal{Y}|$ are finite, for any $\epsilon > 0$, $\delta > 0$ there exists N such that $\forall n \geq N$, with probability at least $1 - \delta$, $\forall x \in \mathcal{X}, \forall y \in \mathcal{Y}$,

$$|P_{\theta_n}(y|x) - P_{\mathcal{D}}(y|x)| < \epsilon,\ |P_{\theta_n}(x|y) - P_{\mathcal{D}}(x|y)| < \epsilon$$

The transition operators are defined as follows:

$$\begin{aligned} T_n^{\mathcal{Y}}(y[t]|y[t-1]) &= \sum_{x \in \mathcal{X}} P_{\theta_n}(y[t]|x)\, P_{\theta_n}(x|y[t-1]), \\ T^{\mathcal{Y}}(y[t]|y[t-1]) &= \sum_{x \in \mathcal{X}} P_{\mathcal{D}}(y[t]|x)\, P_{\mathcal{D}}(x|y[t-1]). \end{aligned}$$

For data-generating distribution, $P_{\mathcal{D}}(x|y)$ and $P_{\mathcal{D}}(y|x)$ are derived from $P_{\mathcal{D}}(x, y)$. Then, for $\forall n \geq N$, we have, for any $y_t, y_{t-1} \in \mathcal{Y}$, with probability at least $1 - \delta$,

$$\begin{aligned} &\left| T_n^{\mathcal{Y}}(y_t|y_{t-1}) - T^{\mathcal{Y}}(y_t|y_{t-1}) \right| \\ &\quad \leq \left| \sum_{x \in \mathcal{X}} P_{\theta_n}(y_t|x)\, P_{\theta_n}(x|y_{t-1}) - P_{\mathcal{D}}(y_t|x)\, P_{\mathcal{D}}(x|y_{t-1}) \right| \\ &\quad \leq |\mathcal{X}| \max_{x \in \mathcal{X}} \left| P_{\theta_n}(y_t|x)\, P_{\theta_n}(x|y_{t-1}) - P_{\mathcal{D}}(y_t|x)\, P_{\mathcal{D}}(x|y_{t-1}) \right| \qquad \text{(D.15)} \\ &\quad \leq |\mathcal{X}|(2\epsilon) \end{aligned}$$

As we assume finite sets $\mathcal{X}$ and $\mathcal{Y}$, this proves the convergence (in probability) of transition operator $T_n^{\mathcal{Y}}$ to $T^{\mathcal{Y}}$. The same argument holds for the convergence of transition operator $T_n^{\mathcal{X}}$ to $T^{\mathcal{X}}$. Together with Proposition D.1, we have proved the convergence of

asymptotic marginal distribution $\pi_n(X)$ and $\pi_n(Y)$ to data-generating marginal distributions $P_{\mathcal{D}}(X)$ and $P_{\mathcal{D}}(Y)$, respectively.

Now, let's look at the joint probability distributions $P_{\theta_n}(x, y) = P_{\theta_n}(x|y)P_{\theta_n}(y)$ and similarly, $P_{\mathcal{D}}(x, y) = P_{\mathcal{D}}(x|y)P_{\mathcal{D}}(y)$. From a similar argument as above, with probability at least $1 - \delta$, there exists N' such that the following inequalities hold $\forall n \geq N', \forall x \in \mathcal{X}, \forall y \in \mathcal{Y}$:

$$\left|P_{\theta_n}(y) - P_{\mathcal{D}}(y)\right| < \epsilon, \ \left|P_{\theta_n}(x|y) - P_{\mathcal{D}}(x|y)\right| < \epsilon \tag{D.16}$$

Therefore, using the similar argument in Equation (D.15), we have

$$\left|P_{\theta_n}(x, y) - P_{\mathcal{D}}(x, y)\right| < 2\epsilon \tag{D.17}$$

and this completes the proof. □

D.3 Derivation of Equation (6.8)

We derive the mean-field updates described by Equation (6.8) by solving the following optimization problem:

$$Q^* = \arg\min_{Q(y,h)} \mathrm{KL}\left(Q(y, h) \| P_\theta(y, h|x)\right) \tag{D.18}$$

subject to $Q(y, h) = \prod_{j,k} Q(y_j)Q(h_k)$. Assuming binary visible and binary hidden variables, we define $\hat{h}_k = Q(h_k = 1)$ and $\hat{y}_j = Q(y_j = 1)$. The KL divergence can be written as follows:

$$\mathrm{KL}\left(Q(y, h) \| P_\theta(y, h|x)\right) = \sum_{y,h} Q(y, h)\left(\log Q(y, h) - \log P_\theta(y, h|x)\right) \tag{D.19}$$

where the first term can be further decomposed into

$$\begin{aligned}
&\sum_{y,h} Q(y,h)\log Q(y,h) \\
&= \sum_{j}\sum_{y_j\in\{0,1\}} Q(y_j)\log Q(y_j) + \sum_{k}\sum_{h_k\in\{0,1\}} Q(h_k)\log Q(h_k) \\
&= \sum_{j} \hat{y}_j\log\hat{y}_j + (1-\hat{y}_j)\log(1-\hat{y}_j) + \sum_{k}\hat{h}_k\log\hat{h}_k + (1-\hat{h}_k)\log(1-\hat{h}_k) \qquad \text{(D.20)}
\end{aligned}$$

The second term of Equation (D.19) can be written as follows:

$$\sum_{y,h} Q(y,h)\log P_\theta(y,h|x) = \sum_{y,h} Q(y,h)\left(\log\exp\left(-E(x,y,h)\right) - \log Z(x)\right) \qquad \text{(D.21)}$$

Since $Z(x) = \sum_{y,h}\exp\left(-E(x,y,h)\right)$ is independent of $\hat{h}_k$'s or $\hat{y}_j$'s, we can ignore the second term for optimization. Then, $\sum_{y,h} Q(y,h)\log\exp\left(-E(x,y,h)\right)$

$$\begin{aligned}
&= \sum_{y,h} Q(y,h)\left(\sum_{i,k} x_i W^x_{ik} h_k + \sum_{j,k} y_j W^y_{jk} h_k + \sum_k b_k h_k + \sum_i c^x_i x_i + \sum_j c^y_j y_j\right) \qquad \text{(D.22)} \\
&= \sum_{j,k}\left(\sum_i x_i W^x_{ik}\hat{h}_k + \sum_i c^x_i x_i + \hat{y}_j W^y_{jk}\hat{h}_k + b_k\hat{h}_k + c^y_j\hat{y}_j\right) \qquad \text{(D.23)}
\end{aligned}$$

Finally, the optimization problem in Equation (D.18) can be written as follows:

$$\begin{aligned}
Q^* = \arg\min_{\hat{h}_k,\hat{y}_j}\Bigg[&-\sum_{j,k}\left(\sum_i x_i W^x_{ik}\hat{h}_k + \sum_i c^x_i x_i + \hat{y}_j W^y_{jk}\hat{h}_k + b_k\hat{h}_k + c^y_j\hat{y}_j\right) \\
&+\sum_j \hat{y}_j\log\hat{y}_j + (1-\hat{y}_j)\log(1-\hat{y}_j) + \sum_k \hat{h}_k\log\hat{h}_k + (1-\hat{h}_k)\log(1-\hat{h}_k)\Bigg] \qquad \text{(D.24)}
\end{aligned}$$

Taking derivatives with respect to $\hat{h}_k$ and $\hat{y}_j$, we obtain

$$\frac{\partial \mathrm{KL}}{\partial \hat{h}_k} = \log\frac{\hat{h}_k}{1-\hat{h}_k} - \sum_i x_i W^x_{ik} - \sum_j \hat{y}_j W^y_{jk} - b_k = 0 \qquad \text{(D.25)}$$

$$\frac{\partial \mathrm{KL}}{\partial \hat{y}_j} = \log\frac{\hat{y}_j}{1-\hat{y}_j} - \sum_k \hat{h}_k W^y_{jk} - c^y_j = 0 \qquad \text{(D.26)}$$

Solving these, we have

$$\begin{aligned}
\hat{h}_k &= \frac{\exp\left(\sum_i x_i W^x_{ik} + \sum_j \hat{y}_j W^y_{jk} + b_k\right)}{1 + \exp\left(\sum_i x_i W^x_{ik} + \sum_j \hat{y}_j W^y_{jk} + b_k\right)} = \sigma\left(\sum_i x_i W^x_{ik} + \sum_j \hat{y}_j W^y_{jk} + b_k\right) \\
\hat{y}_j &= \frac{\exp\left(\sum_k \hat{h}_k W^y_{jk} + c^y_j\right)}{1 + \exp\left(\sum_k \hat{h}_k W^y_{jk} + c^y_j\right)} = \sigma\left(\sum_k \hat{h}_k W^y_{jk} + c^y_j\right)
\end{aligned} \tag{6.8}$$

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] `vis-www.cs.umass.edu/lfw/part_labels/`, 2013. 78

[2] Ankur Agarwal and Bill Triggs. Hyperfeatures–multilevel local coding for visual recognition. In *Proceedings of the European Conference on Computer Vision*, pages 30–43. Springer, 2006. 1, 9

[3] Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep canonical correlation analysis. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1247–1255, 2013. 110

[4] Pablo Arbeláez, Bharath Hariharan, Chunhui Gu, Saurabh Gupta, Lubomir Bourdev, and Jitendra Malik. Semantic segmentation using regions and parts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3378–3385. IEEE, 2012. 70

[5] George Bachman and Lawrence Narici. *Functional Analysis*. Dover Publications, 2012. 140

[6] Pierre Baldi. Autoencoders, unsupervised learning, and deep architectures. *ICML Unsupervised and Transfer Learning*, 27:37–50, 2012. 1

[7] Y. Bengio, L. Yao, G. Alain, and P. Vincent. Generalized denoising auto-encoders as generative models. In *Advances in Neural Information Processing Systems*, 2013. 88, 89, 138

[8] Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009. 45

[9] Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in Neural Information Processing Systems*, 19:153, 2007. 1, 9, 28, 45

[10] Yoshua Bengio, Eric Thibodeau-Laufer, Guillaume Alain, and Jason Yosinski. Deep generative stochastic networks trainable by backprop. In *Proceedings of the International Conference on Machine Learning*, 2014. 88, 89, 138

[11] Christopher M Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995. 1

[12] Christopher M Bishop. *Pattern Recognition and Machine Learning*, volume 1. Springer New York, 2006. 20

[13] Eran Borenstein and Shimon Ullman. Combined top-down/bottom-up segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(12): 2109–2125, 2008. 70

[14] Anna Bosch, Andrew Zisserman, and Xavier Munoz. Image classification using random forests and ferns. In *Proceedings of the International Conference on Computer Vision*, 2007. 97

[15] Y-Lan Boureau, Francis Bach, Yann LeCun, and Jean Ponce. Learning mid-level features for recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 1, 9, 12, 24, 62

[16] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. 43

[17] Hung-An Chang and James R Glass. Hierarchical large-margin gaussian mixture models for phonetic classication. In *Proceedings of IEEE workshop on Automatic Speech Recognition and Understanding*, 2007. 43

[18] Dan C Cireşan, Ueli Meier, Jonathan Masci, Luca M Gambardella, and Jürgen Schmidhuber. High-performance neural networks for visual object classification. *arXiv preprint arXiv:1102.0183*, 2011. 40

[19] Adam Coates and Andrew Y Ng. The importance of encoding versus training with sparse coding and vector quantization. In *Proceedings of the International Conference on Machine Learning*, 2011. xii, 40, 41

[20] Adam Coates and Andrew Y Ng. Selecting receptive fields in deep networks. In *Advances in Neural Information Processing Systems*, 2011. 40, 41

[21] Adam Coates, Honglak Lee, and Andrew Y Ng. An analysis of single-layer networks in unsupervised feature learning. In *International Conference on Artificial Intelligence and Statistics*, 2011. xii, 25, 29, 40, 42

[22] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2005. 1, 9, 12

[23] Steven Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 28(4):357–366, 1980. 1

[24] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 2, 100

[25] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the International Conference on Machine Learning*, 2014. 100

[26] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159, 2011. 2

[27] S Eslami and Christopher Williams. A generative model for parts-based object segmentation. In *Advances in Neural Information Processing Systems*, 2012. 69

[28] S Eslami, , Nicolas Heess, and John Winn. The shape Boltzmann machine: A strong model of object shape. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 66, 69, 77

[29] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 64

[30] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 2008. 24, 56, 58

[31] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1915–1929, 2013. 108, 110

[32] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. In *CVPR Workshop on Generative Model Based Vision*, 2004. 1, 9, 23, 24, 61

[33] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Region-based convolutional networks for accurate object detection and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP(99):1–1, 2015. ISSN 0162-8828. doi: 10.1109/TPAMI.2015.2437384. 108

[34] Xavier Glorot, Antoire Bordes, and Yoshua Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the International Conference on Machine Learning*, 2011. 1

[35] Hanlin Goh, Nicolas Thome, Matthieu Cord, and Joo-Hwee Lim. Unsupervised and supervised visual codes with restricted Boltzmann machines. In *Proceedings of the European Conference on Computer Vision*, 2012. 62

[36] I. Goodfellow, M. Mirza, A. Courville, and Y. Bengio. Multi-prediction deep Boltzmann machines. In *Advances in Neural Information Processing Systems*, 2013. 85, 93

[37] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. Technical report, California Institute of Technology, 2007. 23, 24, 25, 62

[38] Matthieu Guillaumin, Jakob Verbeek, and Cordelia Schmid. Multimodal semi-supervised learning for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 98, 100

[39] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003. 46

[40] Xuming He, Richard S Zemel, and Miguel Á Carreira-Perpiñán. Multiscale conditional random fields for image labeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2004. 69, 70

[41] Xuming He, Richard S Zemel, and Debajyoti Ray. Learning and incorporating top-down cues in image segmentation. In *Proceedings of the European Conference on Computer Vision*, 2006. 70

[42] David Heckerman, David Maxwell Chickering, Christopher Meek, Robert Rounthwaite, and Carl Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *The Journal of Machine Learning Research*, 1:49–75, 2001. 90

[43] Nicolas Heess, Nicolas Le Roux, and John Winn. Weakly supervised learning of foreground-background segmentation using masked RBMs. In *Proceedings of the International Conference on Artificial Neural Networks*, 2011. 48

[44] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002. 14, 71, 92

[45] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. 85

[46] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006. 1, 9, 28, 45

[47] Roger A Horn and Charles R Johnson. *Matrix Analysis*. Cambridge university press, 2012. 139

[48] Gary B Huang, Vidit Jain, and Erik Learned-Miller. Unsupervised joint alignment of complex images. In *Proceedings of the International Conference on Computer Vision*, 2007. xi, 78

[49] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled Faces in the Wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, 2007. 77, 110, 114

[50] Gary B. Huang, Manjunath Narayana, and Erik Learned-Miller. Towards unconstrained face recognition. In *CVPR Workshop on Perceptual Organization in Computer Vision*, 2008. 65, 68, 79, 114

[51] Gary B Huang, Honglak Lee, and Erik Learned-Miller. Learning hierarchical representations for face verification with convolutional deep belief networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 83

[52] Gary B. Huang, Marwan Mattar, Honglak Lee, and Erik Learned-Miller. Learning to align from scratch. In *Advances in Neural Information Processing Systems*, 2012. 83

[53] Jing Huang and Brian Kingsbury. Audio-visual deep learning for noise robust speech recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2013. 84

[54] Mark J. Huiskes and Michael S. Lew. The MIR Flickr retrieval evaluation. In *Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval*, MIR '08, pages 39–43, 2008. 97

[55] Mark J. Huiskes, Bart Thomee, and Michael S. Lew. New trends and ideas in visual concept detection: The MIR Flickr retrieval evaluation initiative. In *Proceedings of the International Conference on Multimedia Information Retrieval*, MIR '10, pages 527–536, 2010. 97

[56] Aapo Hyvärinen, Patrik Hoyer, and Mika Inki. Topographic independent component analysis. *Neural Computation*, 13(7):1527–1558, 2001. ISSN 0899-7667. doi: http://dx.doi.org/10.1162/089976601750264992. 31

[57] Anil Jain and Douglas Zongker. Feature selection: Evaluation, application, and small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):153–158, 1997. 46

[58] Andrew Kae, Kihyuk Sohn, Honglak Lee, and Erik Learned-Miller. Augmenting crfs with boltzmann machine shape priors for image labeling. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2019–2026. IEEE, 2013. 109, 114, 115

[59] Koray Kavukcuoglu, Marc Aurelio Ranzato, Rob Fergus, and Yann Le-Cun. Learning invariant features through topographic filter maps. In *CVPR*, 2009. 31

[60] Koray Kavukcuoglu, Pierre Sermanet, Y-Lan Boureau, Karol Gregor, Michaël Mathieu, and Yann LeCun. Learning convolutional feature hierachies for visual recognition. In *Advances in Neural Information Processing Systems*, 2010. 9, 12, 28, 30, 39

[61] Yelin Kim, Honglak Lee, and Emily Mower Provost. Deep learning for robust feature generation in audiovisual emotion recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2013. 84

[62] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 7, 102, 103, 107

[63] Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014. 7, 102, 105

[64] Jyri J Kivinen and Christopher KI Williams. Transformation equivariant boltzmann machines. In *Artificial Neural Networks and Machine Learning–ICANN 2011*, pages 1–9. Springer, 2011. 30

[65] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012. 100, 102

[66] Alex Krizhevsky and Geoffrey E Hinton. Learning multiple layers of features from tiny images. *Master's Thesis, University of Toronto*, 2009. 29, 40

[67] Alex Krizhevsky and Geoffrey E Hinton. Convolutional deep belief networks on CIFAR-10. *Unpublished manuscript*, 2010. 40

[68] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012. 1, 2

[69] Neeraj Kumar, Alexander C Berg, Peter N Belhumeur, and Shree K Nayar. Attribute and simile classifiers for face verification. In *Proceedings of the International Conference on Computer Vision*, 2009. 65, 82

[70] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning*, 2001. 66, 70

[71] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. RGB-D object recognition: Features, algorithms, and a large scale benchmark. In *Consumer Depth Cameras for Computer Vision*, pages 167–192. Springer, 2013. 84

[72] Hugo Larochelle and Yoshua Bengio. Classification using discriminative restricted Boltzmann machines. In *Proceedings of the International Conference on Machine Learning*, 2008. 48, 53, 54, 59

[73] Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the International Conference on Machine Learning*, 2007. 1, 36, 46, 58

[74] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2006. 1, 9, 12, 24, 62

[75] Quoc V Le, Rajat Monga, Matthieu Devin, Kai Chen, Greg S Corrado, Jeff Dean, and Andrew Y Ng. Building high-level features using large scale unsupervised learning. In *Proceedings of the International Conference on Machine Learning*, 2012. 83

[76] Nicolas Le Roux, Nicolas Heess, Jamie Shotton, and John Winn. Learning a generative model of images by factoring appearance and shape. *Neural Computation*, 23(3):593–650, 2011. 48

[77] 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, 1989. 30

[78] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1, 95, 110

[79] Yann LeCun, Sumit Chopra, Raia Hadsell, Marc'Aurelio Ranzato, and Fu-Jie Huang. A tutorial on energy-based learning. In G. Bakir, T. Hofman, B. Schölkopf, A. Smola, and B. Taskar, editors, *Predicting Structured Data*. MIT Press, 2006. 76

[80] H. Lee, Y. Largman, P. Pham, and A. Y. Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Advances in Neural Information Processing Systems*, 2009. 43

[81] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y Ng. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems*, 2007. 1

[82] Honglak Lee, Chaitanya Ekanadham, and Andrew Y Ng. Sparse deep belief network model for visual area V2. In *Advances in Neural Information Processing Systems*, 2008. xii, 10, 16, 21, 33, 59

[83] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the International Conference on Machine Learning*, 2009. 12

[84] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Unsupervised learning of hierarchical representations with convolutional deep belief networks. *Communications of the ACM*, 54(10):95–103, 2011. 1, 15, 28, 30, 32, 38, 56, 60, 61

[85] Kuang-chih Lee, Dragomir Anguelov, Baris Sumengen, and Salih Burak Gokturk. Markov random field models for hair and face segmentation. In *Automatic Face & Gesture Recognition, 2008. FG'08. 8th IEEE International Conference on*, 2008. 68

[86] Ian Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. In *Robotics: Science and Systems*, 2013. 84

[87] Bruce G Lindsay. Composite likelihood methods. *Contemporary Mathematics*, 80 (1):221–39, 1988. 90

[88] David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision*, 1999. 1, 9, 11, 29

[89] Jitendra Malik, Serge Belongie, Jianbo Shi, and Thomas Leung. Textons, contours and regions: Cue integration in image segmentation. In *Proceedings of the International Conference on Computer Vision*, 1999. 79

[90] Bangalore S Manjunath, J-R Ohm, Vinod V Vasudevan, and Akio Yamada. Color and texture descriptors. *Circuits and Systems for Video Technology, IEEE Transactions on*, 11(6):703–715, 2001. 97

[91] Christopher Poultney MarcâĂŹAurelio Ranzato, Sumit Chopra, and Yann LeCun. Efficient learning of sparse representations with an energy-based model. In *Advances in Neural Information Processing Systems*, 2007. 1, 9, 28, 45

[92] David R Martin, Charless C Fowlkes, and Jitendra Malik. Learning to detect natural image boundaries using brightness and texture. In *Advances in Neural Information Processing Systems*, 2002. 79

[93] Roland Memisevic and Geoffrey E Hinton. Learning to represent spatial transformations with factored higher-order Boltzmann machines. *Neural Computation*, 22(6):1473–1492, 2010. 51

[94] Volodymyr Mnih, Hugo Larochelle, and Geoffrey E Hinton. Conditional restricted Boltzmann machines for structured output prediction. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2011. 76, 85, 92

[95] Kevin P. Murphy, Y. Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 1999. 71

[96] Vinod Nair and Geoffrey Hinton. Implicit mixtures of restricted boltzmann machines. In *Advances in Neural Information Processing Systems*, 2006. 47, 51

[97] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010. 2

[98] Jiquan Ngiam, Zhenghao Chen, Sonia A Bhaskar, Pang W Koh, and Andrew Y Ng. Sparse filtering. In *Advances in Neural Information Processing Systems*, 2011. 42, 43

[99] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. Multimodal deep learning. In *Proceedings of the International Conference on Machine Learning*, 2011. xiii, 84, 85, 90, 91, 98, 99

[100] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3): 145–175, 2001. 97

[101] B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996. 1, 9

[102] Pedro HO Pinheiro and Ronan Collobert. Recurrent convolutional neural networks for scene parsing. *Proceedings of the International Conference on Machine Learning*, 2013. 108, 109

[103] Abdel rahman Mohamed, Tara N. Sainath, George E. Dahl, Bhuvana Ramabhadran, Geoffrey E. Hinton, and Michael A. Picheny. Deep belief networks using discriminative features for phone recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 5060–5063, 2011. 1

[104] Marc'Aurelio Ranzato, Alex Krizhevsky, and Geoffrey E. Hinton. Factored 3-way restricted Boltzmann machines for modeling natural images. In *International Conference on Artificial Intelligence and Statistics*, 2010. 20, 126

[105] Dantam Rao, Mark De Deuge, Navid Nourani-Vatani, Bertrand Douillard, Stefan B Williams, and Oscar Pizarro. Multimodal learning for autonomous underwater vehicles from visual and bathymetric data. In *IEEE International Conference on Robotics and Automation*, 2014. 84

[106] Mohammad Rastegari, Ali Farhadi, and David Forsyth. Attribute discovery via predictable discriminative binary codes. In *Proceedings of the European Conference on Computer Vision*, 2012. 65

[107] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014. 7, 102, 103, 106, 107, 116

[108] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the International Conference on Machine Learning*, 2011. 59

[109] Salah Rifai, Yoshua Bengio, Aaron Courville, Pascal Vincent, and Mehdi Mirza. Disentangling factors of variation for facial expression recognition. In *Proceedings of the European Conference on Computer Vision*, 2012. 48

[110] Ruslan Salakhutdinov and Geoffrey E Hinton. Replicated softmax: an undirected topic model. In *Advances in Neural Information Processing Systems*, 2009. 97

[111] Ruslan Salakhutdinov and Geoffrey E Hinton. Deep Boltzmann machines. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2009. 1, 66, 76, 85, 93

[112] L.K. Saul, T. Jaakkola, and M.I. Jordan. Mean field theory for sigmoid belief networks. *Journal of Artificial Intelligence Research*, 4:61–76, 1996. 71

[113] C. Scheffler, J.M. Odobez, and R. Marconi. Joint adaptive colour modelling and skin, hair and clothing segmentation using coherent probabilistic index maps. In *Proceedings of the British Machine Vision Conference*, 2011. 68

[114] T. J. Sejnowski. Higher-order Boltzmann machines. In *AIP Conference Proceedings on Neural Networks for Computing*, 1987. 51

[115] H-C. Shin, M. R. Orton, D. J. Collins, S. J. Doran, and M. O. Leach. Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4D patient data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1930–1943, 2013. 84

[116] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 109

[117] P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. *Parallel distributed processing: Explorations in the microstructure of cognition*, 1:194–281, 1986. 1, 66, 71

[118] K. Sohn and H. Lee. Learning invariant representations with local transformations. In *Proceedings of the International Conference on Machine Learning*, 2012. 60

[119] Kihyuk Sohn, Dae Yon Jung, Honglak Lee, and Alfred Hero III. Efficient learning of sparse, distributed, convolutional feature representations for object recognition. In *Proceedings of the International Conference on Computer Vision*, 2011. 30, 62, 63

[120] Kihyuk Sohn, Wenling Shang, and Honglak Lee. Improved multimodal deep learning with variation of information. In *Advances in Neural Information Processing Systems*, pages 2141–2149, 2014. 109, 110

[121] N. Srivastava and R. Salakhutdinov. Multimodal learning with deep Boltzmann machines. In *Advances in Neural Information Processing Systems*, 2012. 85, 90, 91, 97, 98, 99, 100

[122] Nitish Srivastava and Ruslan Salakhutdinov. Discriminative transfer learning with tree-based priors. In *Advances in Neural Information Processing Systems*, 2013. xiii, 98

[123] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014. 2

[124] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014. 109

[125] Y. Tang and R. Salakhutdinov. Learning stochastic feedforward neural networks. In *Advances in Neural Information Processing Systems*, 2013. 102

[126] Y. Tang, R. Salakhutdinov, and G. E. Hinton. Robust Boltzmann machines for recognition and denoising. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 48

[127] T. Tieleman. Training restricted Boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the International Conference on Machine Learning*, 2008. 92, 95, 98

[128] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 4, 2012. 2

[129] J. C. van Gemert, J. M. Geusebroek, C. J. Veenman, and A. W. M. Smeulders. Kernel codebooks for scene categorization. In *Proceedings of the European Conference on Computer Vision*, volume 3, 2008. 25

[130] J. H. van Hateren and A. van der Schaaf. Independent component filters of natural images compared with simple cells in primary visual cortex. *Proceedings of the Royal Society B*, 265:359–366, 1998. 39

[131] Jakob Verbeek, Matthieu Guillaumin, Thomas Mensink, and Cordelia Schmid. Image annotation with tagprop on the MIR Flickr set. In *Proceedings of the international conference on Multimedia information retrieval*. ACM, 2010. 98

[132] P. Vincent, H. Larochelle, Y. Bengio, and P. A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the International Conference on Machine Learning*, 2008. 28, 29, 38, 59, 108

[133] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11:3371–3408, 2010. 37

[134] A. Wang, J. Lu, G. Wang, J. Cai, and T-J. Cham. Multi-modal unsupervised feature learning for RGB-D scene labeling. In *Proceedings of the European Conference on Computer Vision*. Springer, 2014. 84

[135] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 11, 12, 24, 25

[136] N. Wang, H. Ai, and S. Lao. A compositional exemplar-based model for hair segmentation. In *Proceedings of the Asian Conference on Computer Vision*, 2011. 68

[137] Nan Wang, Haizhou Ai, and Feng Tang. What are good parts for hair shape modeling? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 68, 81, 114, 115

[138] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-ucsd birds 200. 2010. 110, 112

[139] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In *Advances in Neural Information Processing Systems*, 2001. 46

[140] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *Proceedings of the International Conference on Computer Vision*, 2005. 1, 9

[141] Lior Wolf, Tal Hassner, and Yaniv Taigman. Effective unconstrained face recognition by combining multiple descriptors and learned background statistics. *IEEE-TPAMI*, 33(10):1978–1990, 2011. ISSN 0162-8828. doi: 10.1109/TPAMI.2010.230. URL `http://dx.doi.org/10.1109/TPAMI.2010.230`. 78

[142] Y. Yacoob and L.S. Davis. Detection and analysis of hair. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(7):1164–1169, 2006. 68

[143] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 1, 9, 11, 12, 24, 25, 62

[144] Jimei Yang, Simon Sáfár, and Ming-Hsuan Yang. Max-margin boltzmann machines for object segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 320–327. IEEE, 2014. 112, 113

[145] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the International Conference on Machine Learning*, 1997. 46

[146] K. Yu, Y. Lin, and J. Lafferty. Learning image representations from the pixel level via hierarchical sparse coding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011. 38

[147] Kai Yu, Tong Zhang, and Yihong Gong. Nonlinear learning using local coordinate coding. In *Advances in Neural Information Processing Systems*, 2009. 1, 9

[148] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus. Deconvolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 12, 28, 30

[149] Matthew D Zeiler, M Ranzato, Rajat Monga, M Mao, K Yang, Quoc Viet Le, Patrick Nguyen, A Senior, Vincent Vanhoucke, Jeffrey Dean, et al. On rectified linear units for speech processing. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 3517–3521. IEEE, 2013. 2