

# DEEP LEARNING FOR SIGNAL AND INFORMATION PROCESSING

---

**Li Deng and Dong Yu**

*Microsoft Research  
One Microsoft Way  
Redmond, WA 98052*

## Table of Contents

Chapter 1: Introduction .....	6
1.1    Definitions and Background .....	6
1.2    Organization of This Book .....	9
Chapter 2: Historical Context of Deep Learning .....	10
Chapter 3: Three Classes of Deep Learning Architectures .....	17
3.1    A Three-Category Classification .....	17
3.2    Generative Architectures .....	19
3.3    Discriminative Architectures .....	23
3.4    Hybrid Generative-Discriminative Architectures .....	26
Chapter 4: Generative: Deep Autoencoder .....	29
4.1    Introduction .....	29
4.2    Use of Deep Autoencoder to Extract Speech Features .....	30
4.3    Stacked Denoising Autoencoder .....	40
4.4    Transforming Autoencoder .....	41
Chapter 5: Hybrid: Pre-Trained Deep Neural Network .....	42
5.1    Restricted Boltzmann Machine .....	42
5.2    Stacking up RBMs to Form a DBN/DNN .....	46
5.3    Interfacing DNN with HMM .....	48
Chapter 6: Discriminative: Deep Stacking Networks and Variants .....	50
6.1    Introduction .....	50
6.2    Architecture of DSN .....	51
6.3    Tensor Deep Stacking Network .....	53
Chapter 7: Selected Applications in Speech Recognition .....	56
Chapter 8 : Selected Applications in Language Modeling .....	61

Chapter 9 : Selected Applications in Natural Language Processing.....	63
Chapter 10: Selected Applications in Information Retrieval.....	65
Chapter 11: Selected Applications in Image, Vision, & Multimodal/Multitask Processing .....	67
Chapter 12: Epilogues .....	74
BIBLIOGRAPHY .....	78

# ABSTRACT

This short monograph contains the material expanded from two tutorials that the authors gave, one at APSIPA in October 2011 and the other at ICASSP in March 2012. Substantial updates have been made based on the literature up to March, 2013, covering practical aspects in the fast development of deep learning research during the interim year.

In Chapter 1, we provide the background of deep learning, as intrinsically connected to the use of multiple layers of nonlinear transformations to derive features from the sensory signals such as speech and visual images. In the most recent literature, deep learning is embodied as representation learning, which involves a hierarchy of features or concepts where higher-level concepts are defined from lower-level ones and where the same lower-level concepts help to define higher-level ones. In Chapter 2, a brief historical account of deep learning is presented. In particular, the historical development of speech recognition is used to illustrate the recent impact of deep learning. In Chapter 3, a three-way classification scheme for a large body of work in deep learning is developed. We classify a growing number of deep architectures into generative, discriminative, and hybrid categories, and present qualitative descriptions and a literature survey for each category. From Chapter 4 to Chapter 6, we discuss in detail three popular deep learning architectures and related learning methods, one in each category. Chapter 4 is devoted to deep autoencoders as a prominent example of the (non-probabilistic) generative deep learning architectures. Chapter 5 gives a major example in the hybrid deep architecture category, which is the discriminative feed-forward neural network with many layers using layer-by-layer generative pre-training. In Chapter 6, deep stacking networks and several of the variants are discussed in detail, which exemplify the discriminative deep architectures in the three-way classification scheme.

From Chapters 7-11, we select a set of typical and successful applications of deep learning in diverse areas of signal and information processing. In Chapter 7, we review the applications of deep learning to speech recognition and audio processing. In Chapters 8 and 9, we present recent results of applying deep learning in language modeling and natural language processing, respectively. In Chapters 10 and 11, we discuss, respectively, the applications of deep learning in information retrieval and image, vision, and multimodal processing. Finally, an epilogue is given

in Chapter 12 to summarize what we presented in earlier chapters and to discuss future challenges and directions.

# CHAPTER 1

## INTRODUCTION

---

### 1.1 Definitions and Background

Since 2006, deep structured learning, or more commonly called deep learning or hierarchical learning, has emerged as a new area of machine learning research (Hinton et al., 2006; Bengio, 2009). During the past several years, the techniques developed from deep learning research have already been impacting a wide range of signal and information processing work within the traditional and the new, widened scopes including key aspects of machine learning and artificial intelligence; see overview articles in (Bengio et al., 2013; Hinton et al., 2012; Yu and Deng, 2011; Deng, 2011; Arel et al., 2010, and also the recent New York Times media coverage of this progress in (Markoff, 2012). A series of recent workshops, tutorials, and special issues or conference special sessions have been devoted exclusively to deep learning and its applications to various signal and information processing areas. These include: the 2013 ICASSP's special session on New Types of Deep Neural Network Learning for Speech Recognition and Related Applications, the 2010, 2011, and 2012 NIPS Workshops on Deep Learning and Unsupervised Feature Learning, the 2013 ICML Workshop on Deep Learning for Audio, Speech, and Language Processing; the 2012 ICML Workshop on Representation Learning, the 2011 ICML Workshop on Learning Architectures, Representations, and Optimization for Speech and Visual Information Processing, the 2009 ICML Workshop on Learning Feature Hierarchies, the 2009 NIPS Workshop on Deep Learning for Speech Recognition and Related Applications, the 2008 NIPS Deep Learning Workshop, the 2012 ICASSP tutorial on Deep Learning for Signal and Information Processing, the special section on Deep Learning for Speech and Language Processing in IEEE Transactions on Audio, Speech, and Language Processing (January 2012), and the special issue on Learning Deep Architectures in IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI, 2013). The authors have been actively involved in deep learning research and in organizing several of the above events and editorials. In particular, they gave a comprehensive tutorial on this topic at ICASSP 2012. Part of this book is based on the material presented in that tutorial.

Deep learning has various closely related definitions or high-level descriptions:

- **Definition 1:** A class of machine learning techniques that exploit many layers of non-linear information processing for supervised or unsupervised feature extraction and transformation, and for pattern analysis and classification.
- **Definition 2:** “A sub-field within machine learning that is based on algorithms for learning multiple levels of representation in order to model complex relationships among data. Higher-level features and concepts are thus defined in terms of lower-level ones, and such a hierarchy of features is called a deep architecture. Most of these models are based on unsupervised learning of representations.” (Wikipedia on “Deep Learning” around March 2012.)
- **Definition 3:** “A sub-field of machine learning that is based on learning several levels of representations, corresponding to a hierarchy of features or factors or concepts, where higher-level concepts are defined from lower-level ones, and the same lower-level concepts can help to define many higher-level concepts. Deep learning is part of a broader family of machine learning methods based on learning representations. An observation (e.g., an image) can be represented in many ways (e.g., a vector of pixels), but some representations make it easier to learn tasks of interest (e.g., is this the image of a human face?) from examples, and research in this area attempts to define what makes better representations and how to learn them.” see Wikipedia on “Deep Learning” as of this writing in February 2013; see [http://en.wikipedia.org/wiki/Deep\\_learning](http://en.wikipedia.org/wiki/Deep_learning).
- **Definition 4:** “Deep Learning is a new area of Machine Learning research, which has been introduced with the objective of moving Machine Learning closer to one of its original goals: Artificial Intelligence. Deep Learning is about learning multiple levels of representation and abstraction that help to make sense of data such as images, sound, and text.” See <https://github.com/lisa-lab/DeepLearningTutorials>

Note that deep learning that we discuss in this book is learning in deep architectures for signal and information processing, not deep understanding of the signal or information, although in many cases they may be related. It should also be distinguished from the overloaded term in educational psychology: “Deep learning describes an approach to learning that is characterized by active

engagement, intrinsic motivation, and a personal search for meaning.”  
[http://www.blackwellreference.com/public/tocnode?id=g9781405161251\\_chunk\\_g97814051612516\\_ss1-1](http://www.blackwellreference.com/public/tocnode?id=g9781405161251_chunk_g97814051612516_ss1-1)

Common among the various high-level descriptions of deep learning above are two key aspects: 1) models consisting of many layers of nonlinear information processing; and 2) methods for supervised or unsupervised learning of feature representation at successively higher, more abstract layers. Deep learning is in the intersections among the research areas of neural network, graphical modeling, optimization, pattern recognition, and signal processing. Three important reasons for the popularity of deep learning today are drastically increased chip processing abilities (e.g., general purpose graphical processing units or GPGPUs), the significantly lowered cost of computing hardware, and the recent advances in machine learning and signal/information processing research. These advances have enabled the deep learning methods to effectively exploit complex, compositional nonlinear functions, to learn distributed and hierarchical feature representations, and to make effective use of both labeled and unlabeled data.

Active researchers in this area include those at University of Toronto, New York University, University of Montreal, Microsoft Research, Google, IBM Research, Stanford University, Baidu Corp., UC-Berkeley, UC-Irvine, IDIAP, IDSIA, University of British Columbia, University College London, University of Michigan, Massachusetts Institute of Technology, University of Washington, and numerous other places; see <http://deeplearning.net/deep-learning-research-groups-and-labs/> for a more detailed list. These researchers have demonstrated empirical successes of deep learning in diverse applications of computer vision, phonetic recognition, voice search, conversational speech recognition, speech and image feature coding, semantic utterance classification, hand-writing recognition, audio processing, information retrieval, robotics, and even in the analysis of molecules that may lead to discovery of new drugs as reported recently in (Markoff, 2012).

In addition to the reference list provided at the end of this book, which may be outdated not long after the publication of this book, there are a number of excellent and frequently updated reading lists, tutorials, software, and video lectures online at:

- <http://deeplearning.net/reading-list/>
- [http://ufldl.stanford.edu/wiki/index.php/UFLDL\\_Recommended\\_Readings](http://ufldl.stanford.edu/wiki/index.php/UFLDL_Recommended_Readings)



- <http://www.cs.toronto.edu/~hinton/>
- <http://deeplearning.net/tutorial/>
- [http://ufldl.stanford.edu/wiki/index.php/UFLDL\\_Tutorial](http://ufldl.stanford.edu/wiki/index.php/UFLDL_Tutorial)

## 1.2 Organization of This Book

The rest of the book is organized as follows:

In Chapter 2, we provide a brief historical account of deep learning. In Chapter 3, a three-way classification scheme for a majority of the work in deep learning is developed. They include generative, discriminative, and hybrid deep learning architectures. In Chapter 4, we discuss in detail deep autoencoders as a prominent example of the (non-probabilistic) generative deep learning architectures. In Chapter 5, as a major example in the hybrid deep architecture category, we present in detail the DNN with generative pre-training. In Chapter 6, deep stacking networks and several of the variants are discussed in detail, which exemplify the discriminative deep architectures in the three-way classification scheme.

From Chapters 7-11, we select a set of typical and successful applications of deep learning in diverse areas of signal and information processing. In Chapter 7, we review applications of deep learning in speech recognition and audio processing. In Chapters 8 and 9, we present recent results of applying deep learning in language modeling and natural language processing, respectively. In Chapters 10 and 11, we discuss, respectively, the application of deep learning in information retrieval and image, vision, and multimodal processing.

Finally, an epilogue is given in Chapter 12.

# CHAPTER 2

## HISTORICAL CONTEXT OF DEEP LEARNING

---

Until recently, most machine learning and signal processing techniques had exploited shallow-structured architectures. These architectures typically contain at most one or two layers of nonlinear feature transformations. Examples of the shallow architectures are Gaussian mixture models (GMMs), linear or nonlinear dynamical systems, conditional random fields (CRFs), maximum entropy (MaxEnt) models, support vector machines (SVMs), logistic regression, kernel regression, multi-layer perceptrons (MLPs) with a single hidden layer, and extreme learning machines (ELMs). For instance, SVMs use a shallow linear pattern separation model with one or zero feature transformation layer when kernel trick is used or otherwise. (Notable exceptions are the recent kernel methods that have been inspired by and integrated with deep learning; e.g. Cho and Saul, 2009; Deng et al., 2012; Vinyals et al., 2012). Shallow architectures have been shown effective in solving many simple or well-constrained problems, but their limited modeling and representational power can cause difficulties when dealing with more complicated real-world applications involving natural signals such as human speech, natural sound and language, and natural image and visual scenes.

Human information processing mechanisms (e.g., vision and audition), however, suggest the need of deep architectures for extracting complex structure and building internal representation from rich sensory inputs. For example, human speech production and perception systems are both equipped with clearly layered hierarchical structures in transforming the information from the waveform level to the linguistic level (Baker et al., 2009, 2009a; Deng, 1999, 2003). In a similar vein, human visual system is also hierarchical in nature, most in the perception side but interestingly also in the “generation” side (George, 2008; Bouvrie, 2009; Poggio, 2007). It is natural to believe that the state-of-the-art can be advanced in processing these types of natural signals if efficient and effective deep learning algorithms can be developed.

Historically, the concept of deep learning was originated from artificial neural network research. (Hence, one may occasionally hear the discussion of “new-generation neural networks”.) Feed-forward neural networks or MLPs with many hidden layers, which are often referred to as deep neural networks (DNNs), are good examples of the models with a deep architecture. Back-propagation (BP), popularized in 1980’s, has been a well-known algorithm for learning the parameters of these networks. Unfortunately back-propagation alone did not work well in practice then for learning networks with more than a small number of hidden layers (see a review and analysis in (Bengio, 2009; Glorot and Bengio, 2010)). The pervasive presence of local optima in the non-convex objective function of the deep networks is the main source of difficulties in the learning. Back-propagation is based on local gradient descent, and starts usually at some random initial points. It often gets trapped in poor local optima when the batch-mode BP algorithm is used, and the severity increases significantly as the depth of the networks increases. This difficulty is partially responsible for steering away most of the machine learning and signal processing research from neural networks to shallow models that have convex loss functions (e.g., SVMs, CRFs, and MaxEnt models), for which global optimum can be efficiently obtained at the cost of less modeling power.

The optimization difficulty associated with the deep models was empirically alleviated using three techniques: a larger number of hidden units, better learning algorithms, and better parameter initialization techniques.

Using hidden layers with many neurons in a DNN significantly improves the modeling power of the DNN and creates many closely optimal configurations. Even if parameter learning is trapped into a local optimum, the resulting DNN can still perform quite well since the chance of having a poor local optimum is lower than when a small number of neurons are used in the network. Using deep and wide neural networks, however, would cast great demand to the computational power during the training process and this is one of the reasons why it is not until recent years that researchers have started exploring both deep and wide neural networks in a serious manner.

Better learning algorithms also contributed to the success of DNNs. For example, stochastic BP algorithms are in place of the batch-mode BP algorithms for training DNNs nowadays. This is partly because the stochastic gradient descend (SGD) algorithm is the most efficient algorithm when training is carried out on a single machine and the training set is large (Bottou and LeCun, 2004). But more importantly the SGD algorithm can often jump out of the local optimum due to the noisy gradients estimated from a single or a small batch of samples. Other learning algorithms

such as Hessian free (Martens 2010) or Krylov subspace methods (Vinyals and Povey 2011) have shown a similar ability.

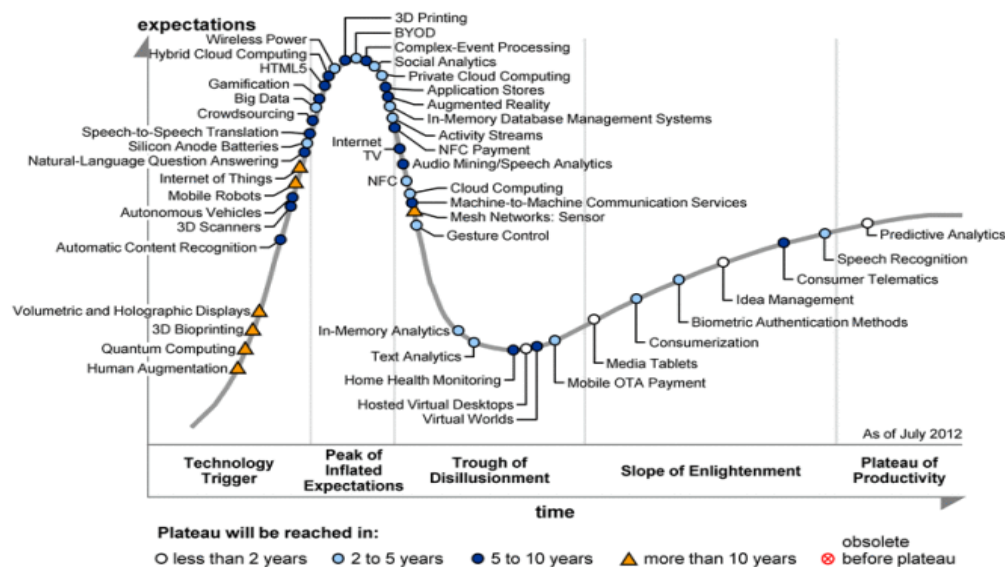
For the highly non-convex optimization problem of DNN learning, it is obvious that better parameter initialization techniques will lead to better models since optimization starts from these initial models. What is not obvious, however, is how to efficiently and effectively initialize DNN parameters until more recently (Hinton et al. 2006; Hinton and Salakhutdinov, 2006; Bengio, 2009; Vincent et al., 2010; Deng et al., 2010; Dahl et al., 2010, 2012; Seide et al. 2011).

The DNN parameter initialization technique that attracted the most attention is the unsupervised pretraining technique proposed in (Hinton et al. 2006; Hinton and Salakhutdinov, 2006). In these papers a class of deep Bayesian probabilistic generative models, called deep belief network (DBN), was introduced. To learn the parameters in the DBN, a greedy, layer-by-layer learning algorithm was developed by treating each pair of layers in the DBN as a Restricted Boltzmann Machine (RBM) (which we will discuss later). This allows for optimizing DBN parameters with computational complexity linear in the depth of the network. It was later found out that the DBN parameters can be directly used as the initial parameters of an MLP or DNN and result in a better MLP or DNN than those randomly initialized after the supervised BP training when the training set is small. As such, DNNs learned with unsupervised DBN pre-training followed by back-propagation fine-tuning is sometimes also called DBNs in the literature (e.g., Dahl et al., 2011; Mohamed et al., 2010, 2012). More recently, researchers have been more careful in distinguishing DNNs from DBNs (Dahl et al., 2012; Hinton et al., 2012), and when DBN is used to initialize the parameters of a DNN, the resulting network is called DBN-DNN (Hinton et al., 2012).

The DBN pretraining procedure is not the only one that allows effective initialization of DNNs. An alternative unsupervised approach that performs equally well is to pretrain DNNs layer by layer by considering each pair of layers as a de-noising auto-encoder regularized by setting a random subset of the inputs to zero (Bengio, 2009; Vincent et al., 2010). Another alternative is to use *contractive* autoencoders for the same purpose by favoring models that is less sensitive to the input variations, i.e., penalizing the gradient of the activities of the hidden units with respect to the inputs (Rifai et al., 2011). Further, Ranzato et al. (2007) developed the Sparse Encoding Symmetric Machine (SESM), which has a very similar architecture to RBMs as building blocks of a DBN. In principle, SESM may also be used to effectively initialize the DNN training. Besides unsupervised pretraining, the supervised pretraining, or sometimes called discriminative pretraining, has also been shown to be effective (Seide et al., 2011; Yu et al., 2011) and in cases where labeled training

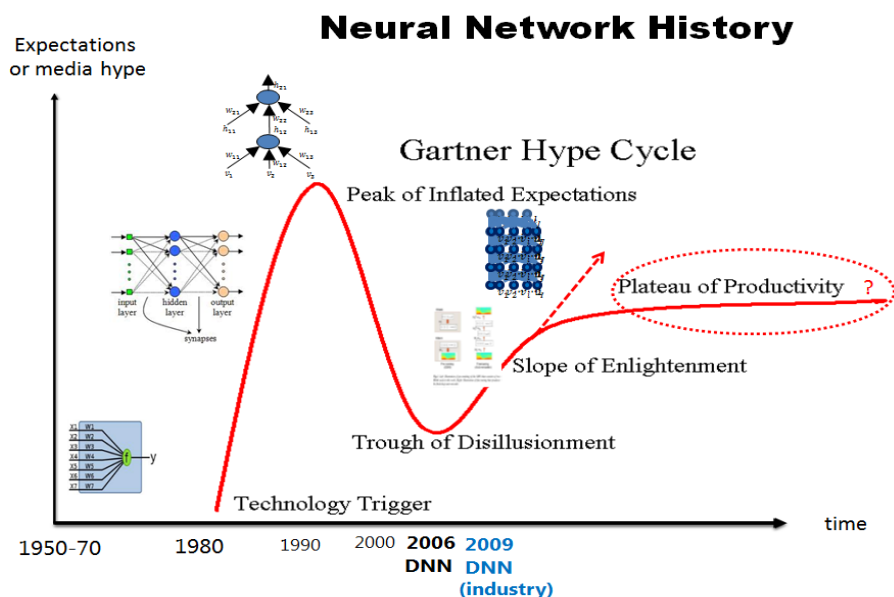
data are abundant performs better than the unsupervised pretraining techniques. The idea of the discriminative pretraining is to start from a one-hidden-layer MLP trained with the BP algorithm. Every time when we want to add a new hidden layer we replace the output layer with a randomly initialized new hidden and output layer and train the whole new MLP (or DNN) using the BP algorithm. Different from the unsupervised pretraining techniques, the discriminative pretraining technique requires labels.

As another way to concisely introduce the DNN, we can review the history of artificial neural network using a “Hype Cycle”, which is a graphic representation of the maturity, adoption and social application of specific technologies. The 2012 version of the Hype Cycles graph compiled by Gartner is shown in Figure 2.1. It intends to show how a technology or application will evolve over time (according to five phases: technology trigger, peak of inflated expectations, trough of disillusionment, slope of enlighten, and plateau of production), and to provide a source of insight to manage its deployment.



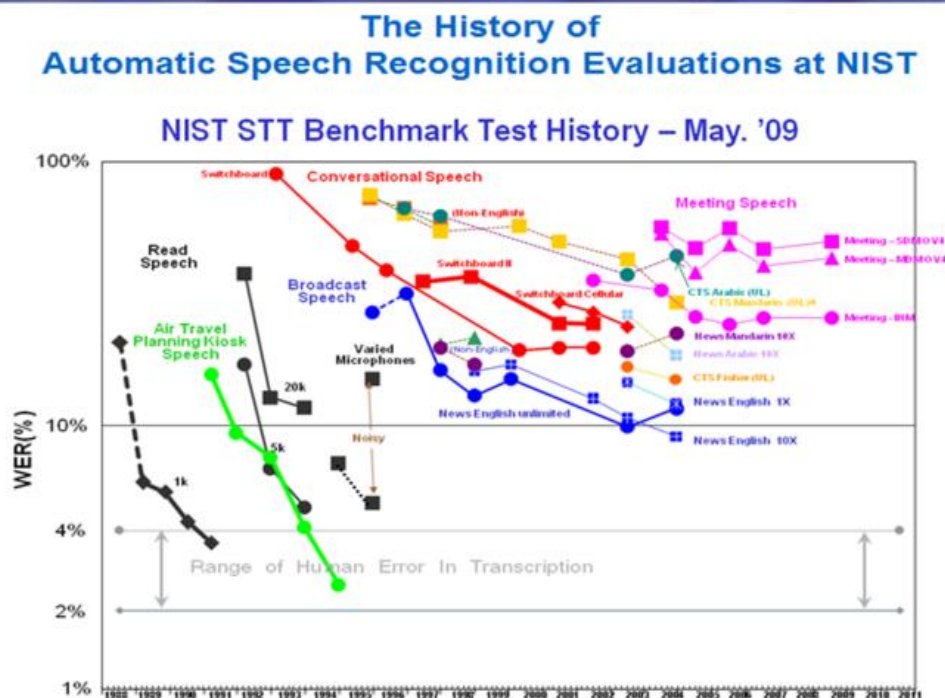
**Figure 2.1.** Gartner Hyper Cycle graph representing five phases of a technology  
[http://en.wikipedia.org/wiki/Hype\\_cycle](http://en.wikipedia.org/wiki/Hype_cycle)

Applying the Gartner Hype Cycle to the artificial neural network development, we created Figure 2.2 to align different generations of neural network with the various phases designated in the Hype Cycle. The peak activities (“expectations” or “media hype” on the vertical axis) occurred late 1980’s and early 1990’s, corresponding to the height of what is often referred to as the “second generation” of neural networks. The deep belief network (DBN) and a fast algorithm for training it were invented in 2006 (Hinton and Salakhudinov, 2006; Hinton et al., 2006). When the DBN was used to initialize the DNN, the learning became highly effective and this has inspired the subsequent fast growing research (“enlightenment” phase shown in Figure 2.2). Applications of the DBN and DNN to industrial speech feature coding and recognition started in 2009 and fast expanded to increasingly larger successes, many of which will be covered in the remainder of this book. The height of the “plateau of productivity” phase, not yet reached, is expected to be higher than in the stereotypical curve (circled with a question mark in Figure 2.2), and is marked by the dashed line that moves straight up.

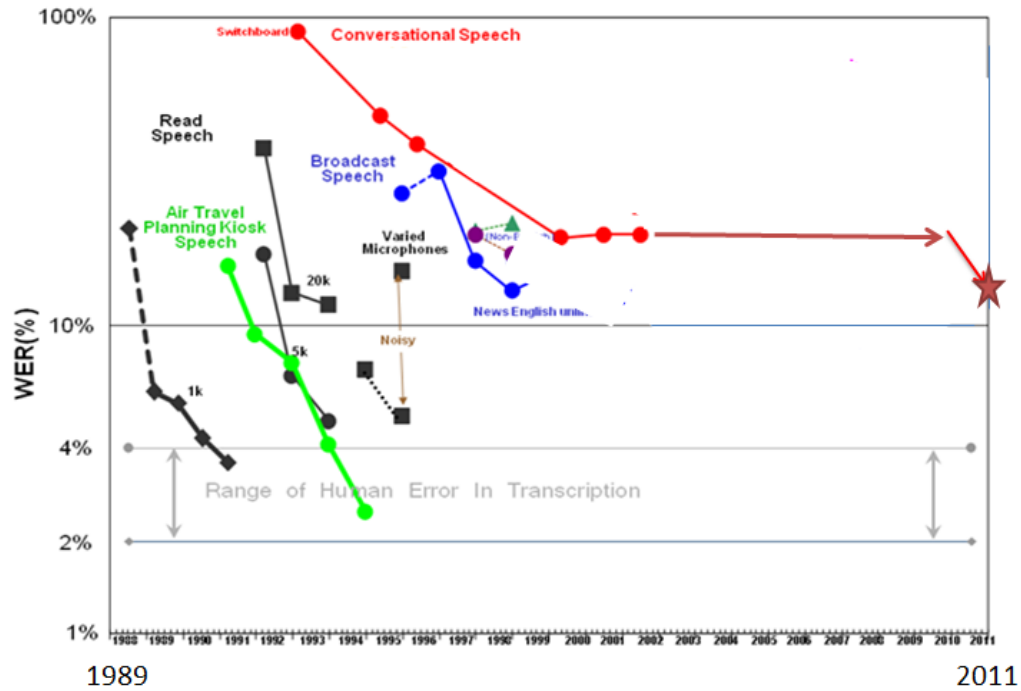


**Figure 2.2:** Applying Gartner Hyper Cycle graph to analyzing the history of artificial neural network technology.

We show in Figure 2.3 the history of speech recognition, which has been compiled by NIST, organized by plotting the word error rate (WER) as a function of time for a number of increasingly difficult speech recognition tasks. Note all WER results were obtained using the GMM-HMM technology. When one particularly difficult task (Switchboard) is extracted from Figure 2.3, we see a flat curve over many years using the GMM-HMM technology but after DNN technology is used the WER drops sharply (marked by star in Figure 2.4).



**Figure 2.3:** The famous NIST plot showing the historical speech recognition error rates achieved by the GMM-HMM approach for a number of increasingly difficult speech recognition tasks.



**Figure 2.4.** Extracting WERs of one task from Figure 2.3 and adding the significantly lower WER (marked by the star) achieved by the DNN technology approach.

In the next Chapter, an overview is provided on the various architectures of deep learning, including and beyond the original DBN proposed in (Hinton et al. 2006).



# CHAPTER 3

## THREE CLASSES OF DEEP LEARNING ARCHITECTURES

---

### 3.1 A Three-Category Classification

As described earlier, deep learning refers to a rather wide class of machine learning techniques and architectures, with the hallmark of using many layers of non-linear information processing that are hierarchical in nature. Depending on how the architectures and techniques are intended for use, e.g., synthesis/generation or recognition/classification, one can broadly categorize most of the work in this area into three classes:

- 1) **Generative deep architectures**, which are intended to capture high-order correlation of the observed or visible data for pattern analysis or synthesis purposes, and/or characterize the joint statistical distributions of the visible data and their associated classes. In the latter case, the use of Bayes rule can turn this type of architecture into a discriminative one.
- 2) **Discriminative deep architectures**, which are intended to directly provide discriminative power for pattern classification purposes, often by characterizing the posterior distributions of classes conditioned on the visible data; and
- 3) **Hybrid deep architectures**, where the goal is discrimination which is assisted (often in a significant way) with the outcomes of generative architectures via better optimization or/and regularization, or where discriminative criteria are used to learn the parameters in any of the deep generative models in category 1) above.

Note the use of “hybrid” in 3) above is different from that used sometimes in the literature, which for example refers to the hybrid systems for speech recognition feeding the output probabilities of a neural network into an HMM (Bengio et al., 1991; Bourlard and Morgan, 1993; Morgan, 2012).

By the commonly adopted machine learning tradition (e.g., Chapter 28 in Murphy, 2012; Deng and Li, 2013), it may be natural to just classify deep learning techniques into deep discriminative models (e.g., DNNs) and deep probabilistic generative models (e.g., DBN, Deep Boltzmann Machine (DBM)). This classification scheme, however, misses a key insight gained in deep learning research about how generative models can greatly improve the training of DNNs and other deep discriminative models via better regularization. Also, deep generative models may not necessarily need to be probabilistic, e.g., deep auto-encoder, stacked denoising auto-encoder, etc. Nevertheless, the traditional two-way classification indeed points to several key differences between deep discriminative models and deep generative probabilistic models. Compared with the two, deep discriminative models such as DNNs are usually more efficient to train and test, more flexible to construct, and more suitable for end-to-end learning of complex systems (e.g., no approximate inference and learning such as loopy belief propagation). The deep probabilistic models, on the other hand, are easier to interpret, easier to embed domain knowledge, easier to compose, and easier to handle uncertainty, but are typically intractable in inference and learning for complex systems. These distinctions are retained also in the proposed three-way classification which is hence adopted throughout this book.

Below we review representative work in each of the above three classes, where several basic definitions are summarized in Table 1. Applications of these deep architectures are deferred to Chapters 7-11.

**TABLE 1.** BASIC DEEP LEARNING TERMINOLOGIES

**Deep belief network (DBN):** probabilistic generative models composed of multiple layers of stochastic, hidden variables. The top two layers have undirected, symmetric connections between them. The lower layers receive top-down, directed connections from the layer above.

**Boltzmann machine (BM):** a network of symmetrically connected, neuron-like units that make stochastic decisions about whether to be on or off.

**Restricted Boltzmann machine (RBM):** a special BM consisting of a layer of visible units and a layer of hidden units with no visible-visible or hidden-hidden connections.

**Deep neural network (DNN):** a multilayer perceptron with many hidden layers, whose weights are fully connected and are often initialized using either an unsupervised or a supervised pretraining technique. (In the literature, DBN is sometimes used to mean DNN)

**Deep auto-encoder:** a DNN whose output target is the data input itself.

**Distributed representation:** a representation of the observed data in such a way that they are modeled as being generated by the interactions of many hidden factors. A particular factor learned from configurations of other factors can often generalize well. Distributed representations form the basis of deep learning.

## 3.2 Generative Architectures

Among the various subclasses of generative deep architecture, the energy-based deep models are the most common (e.g., Ngiam et al., 2011; Bengio, 2009; LeCun et al., 2007). The original form of the deep autoencoder (Hinton and Salakhutdinov, 2006; Deng et al., 2010), which we will give more detail about in Chapter 4, is a typical example of the generative model category. Most other forms of deep autoencoders are also generative in nature, but with quite different properties and implementations. Examples are transforming auto-encoders (Hinton et al., 2010), predictive sparse

coders and their stacked version, and de-noising autoencoders and their stacked versions (Vincent et al., 2010).

Specifically, in de-noising autoencoders, the input vectors are first corrupted; e.g., randomly selecting a percentage of the inputs and setting them to zeros. Then the parameters are adjusted for the hidden encoding nodes to reconstruct the original, uncorrupted input data using criteria such as mean square reconstruction error and KL distance between the original inputs and the reconstructed inputs. The encoded representations transformed from the uncorrupted data are used as the inputs to the next level of the stacked de-noising autoencoder.

Another prominent type of generative model is deep Boltzmann machine or DBM (Salakhutdinov and Hinton, 2009, 2012; Srivastava and Salakhutdinov, 2012). A DBM contains many layers of hidden variables, and has no connections between the variables within the same layer. This is a special case of the general Boltzmann machine (BM), which is a network of symmetrically connected units that are on or off based on a stochastic mechanism. While having very simple learning algorithm, the general BMs are very complex to study and very slow to compute in learning. In a DBM, each layer captures complicated, higher-order correlations between the activities of hidden features in the layer below. DBMs have the potential of learning internal representations that become increasingly complex, highly desirable for solving object and speech recognition problems. Further, the high-level representations can be built from a large supply of unlabeled sensory inputs and very limited labeled data can then be used to only slightly fine-tune the model for a specific task at hand.

When the number of hidden layers of DBM is reduced to one, we have Restricted Boltzmann Machine (RBM). Like DBM, there are no hidden-to-hidden and no visible-to-visible connections. The main virtue of RBM is that via composing many RBMs, many hidden layers can be learned efficiently using the feature activations of one RBM as the training data for the next. Such composition leads to Deep Belief Network (DBN), which we will describe in more detail, together with RBMs, in Chapter 5.

The standard DBN has been extended to the factored higher-order Boltzmann machine in its bottom layer, with strong results for phone recognition obtained (Dahl et. al., 2010). This model, called mean-covariance RBM or mcRBM, recognizes the limitation of the standard RBM in its ability to represent the covariance structure of the data. However, it is very difficult to train mcRBM and to use it at the higher levels of the deep architecture. Further, the strong results

published are not easy to reproduce. In the architecture of (Dahl et. al., 2010), the mcRBM parameters in the full DBN are not fine-tuned using the discriminative information as for the regular RBMs in the higher layers due to the high computational cost.

Another representative deep generative architecture is the sum-product network or SPN (Poon and Domingo, 2011; Gens and Domingo, 2012). An SPN is a directed acyclic graph with the data as leaves, and with sum and product operations as internal nodes in the deep architecture. The “sum” nodes give mixture models, and the “product” nodes build up the feature hierarchy. Properties of “completeness” and “consistency” constrain the SPN in a desirable way. The learning of SPN is carried out using the EM algorithm together with back-propagation. The learning procedure starts with a dense SPN. It then finds a SPN structure by learning its weights, where zero weights indicate removed connections. The main difficulty in learning SPN is that the learning signal (i.e., the gradient) quickly dilutes when it propagates to deep layers. Empirical solutions have been found to mitigate this difficulty as reported in (Poon and Domingo, 2011). It was pointed out, however, despite the many desirable generative properties in the SPN, it is difficult to fine tune the parameters using the discriminative information, limiting its effectiveness in classification tasks. This difficulty has been overcome in the subsequent work reported in (Gens and Domingo, 2012), where an efficient backpropagation-style discriminative training algorithm for SPN was presented. It was pointed out that the standard gradient descent, computed by the derivative of the conditional likelihood, suffers from the same gradient diffusion problem well known in the regular DNNs. The trick to alleviate this problem in SPN is to replace the marginal inference with the most probable state of the hidden variables and only propagate gradients through this “hard” alignment. Excellent results on (small-scale) image recognition tasks are reported.

Recurrent neural networks (RNNs) are another important class of deep generative architectures, where the depth can be as large as the length of the input data sequence. RNNs are very powerful for modeling sequence data (e.g., speech or text), but until recently they had not been widely used partly because they are extremely difficult to train properly due to the well-known “gradient explosion” problem. Recent advances in Hessian-free optimization (Martens, 2010) have partially overcome this difficulty using approximated second-order information or stochastic curvature estimates. In the more recent work (Martens and Sutskever, 2011), RNNs that are trained with Hessian-free optimization are used as a generative deep architecture in the character-level language modeling tasks, where gated connections are introduced to allow the current input characters to predict the transition from one latent state vector to the next. Such generative RNN models are demonstrated to be well capable of generating sequential text characters. More recently,

Bengio et al. (2013) and Sutskever (2013) have explored variations of stochastic gradient descent optimization algorithms in training generative RNNs and shown that these algorithms can outperform Hessian-free optimization methods. Mikolov et al. (2010) have reported excellent results on using RNNs for language modeling, which we will review in Chapter 8.

There has been a long history in speech recognition research where human speech production mechanisms are exploited to construct dynamic and deep structure in probabilistic generative models; for a comprehensive review, see book (Deng, 2006). Specifically, the early work described in (Deng 1992, 1993; Deng et al., 1994; Ostendorf et al., 1996, Deng and Sameti, 1996) generalized and extended the conventional shallow and conditionally independent HMM structure by imposing dynamic constraints, in the form of polynomial trajectory, on the HMM parameters. A variant of this approach has been more recently developed using different learning techniques for time-varying HMM parameters and with the applications extended to speech recognition robustness (Yu and Deng, 2009; Yu et al., 2009). Similar trajectory HMMs also form the basis for parametric speech synthesis (Zen et al., 2011; Zen et al., 2012; Ling et al., 2013; Shannon et al., 2013). Subsequent work added a new hidden layer into the dynamic model so as to explicitly account for the target-directed, articulatory-like properties in human speech generation (Deng and Ramsay, 1997; Deng, 1998; Bridle et al., 1998; Deng, 1999; Picone et al., 1999; Deng, 2003; Minami et al., 2003). More efficient implementation of this deep architecture with hidden dynamics is achieved with non-recursive or finite impulse response (FIR) filters in more recent studies (Deng et al., 2006, 2006a, Deng and Yu, 2007). The above deep-structured generative models of speech can be shown as special cases of the more general dynamic Bayesian network model and even more general dynamic graphical models (Bilmes and Bartels, 2005; Bilmes, 2010). The graphical models can comprise many hidden layers to characterize the complex relationship between the variables in speech generation. Armed with powerful graphical modeling tool, the deep architecture of speech has more recently been successfully applied to solve the very difficult problem of single-channel, multi-talker speech recognition, where the mixed speech is the visible variable while the un-mixed speech becomes represented in a new hidden layer in the deep generative architecture (Rennie et al., 2010; Wohlmayr et al., 2011). Deep generative graphical models are indeed a powerful tool in many applications due to their capability of embedding domain knowledge. However, they are often used with inappropriate approximations in inference, learning, prediction, and topology design, all arising from inherent intractability in these tasks for most real-world applications. This problem has been addressed in the recent work of (Stoyanov et al., 2011), which provides an interesting direction for making deep generative graphical models potentially more useful in practice in the future.

The standard statistical methods used for large-scale speech recognition and understanding combine (shallow) hidden Markov models for speech acoustics with higher layers of structure representing different levels of natural language hierarchy. This combined hierarchical model can be suitably regarded as a deep generative architecture, whose motivation and some technical detail may be found in Chapter 7 of the recent book (Kurzweil, 2012) on “Hierarchical HMM” or HHMM. Related models with greater technical depth and mathematical treatment can be found in (Fine et al., 1998) for HHMM and (Oliver et al., 2004) for Layered HMM. These early deep models were formulated as directed graphical models, missing the key aspect of “distributed representation” embodied in the more recent deep generative architectures of DBN and DBM discussed earlier in this chapter.

Finally, dynamic or temporally recursive generative models based on neural network architectures for non-speech applications can be found in (Taylor et al., 2007) for human motion modeling, and in (Socher et al., 2011) for natural language and natural scene parsing. The latter model is particularly interesting because the learning algorithms are capable of automatically determining the optimal model structure. This contrasts with other deep architectures such as DBN where only the parameters are learned while the architectures need to be pre-defined. Specifically, as reported in (Socher et al., 2011), the recursive structure commonly found in natural scene images and in natural language sentences can be discovered using a max-margin structure prediction architecture. It is shown that the units contained in the images or sentences are identified, and the way in which these units interact with each other to form the whole is also identified.

### 3.3 Discriminative Architectures

Many of the discriminative techniques in signal and information processing are shallow architectures such as HMMs (e.g., Juang et al., 1997; Povey and Woodland, 2002; He et al., 2008; Jiang and Li, 2010; Xiao and Deng, 2010; Gibson and Hain, 2010) and conditional random fields (CRFs) (e.g., Yang and Furui, 2009; Yu et al., 2010; Hifny and Renals, 2009; Heintz et al., 2009; Zweig and Nguyen, 2009; Peng et al., 2009). Since a CRF is defined with the conditional probability on input data as well as on the output labels, it is intrinsically a shallow discriminative architecture. (Interesting equivalence between CRF and discriminatively trained Gaussian models and HMMs can be found in Heigold et al., 2011). More recently, deep-structured CRFs have been developed by stacking the output in each lower layer of the CRF, together with the original input data, onto its higher layer (Yu et al., 2010a). Various versions of deep-structured CRFs are

successfully applied to phone recognition (Yu and Deng, 2010), spoken language identification (Yu et al., 2010a), and natural language processing (Yu et al., 2010). However, at least for the phone recognition task, the performance of deep-structured CRFs, which are purely discriminative (non-generative), has not been able to match that of the hybrid approach involving DBN, which we will take on shortly.

Morgan (2012) gives an excellent review on other major existing discriminative models in speech recognition based mainly on the traditional neural network or MLP architecture using back-propagation learning with random initialization. It argues for the importance of both the increased width of each layer of the neural networks and the increased depth. In particular, a class of deep neural network models forms the basis of the popular “tandem” approach (Morgan et al., 2005), where the output of the discriminatively learned neural network is treated as part of the observation variable in HMMs. For some representative recent work in this area, see (Pinto et al., 2011; Ketabdar and Bourlard, 2010).

In the most recent work of (Deng et al., 2011; Deng et al., 2012a; Tur et al., 2012; Lena et al., 2012; Vinyals et al., 2012), a new deep learning architecture, sometimes called Deep Stacking Network (DSN), together with its tensor variant (Hutchinson et al., 2012, 2013) and its kernel version (Deng et al., 2012), are developed that all focus on discrimination with scalable, parallelizable learning relying on little or no generative component. We will describe this type of discriminative deep architecture in detail in Chapter 6.

Recurrent neural networks (RNNs) have been successfully used as a generative model, as discussed previously. They can also be used as a discriminative model where the output is a label sequence associated with the input data sequence. Note that such discriminative RNNs were applied to speech a long time ago with limited success (e.g., Robinson, 1994). In Robinson’s implementation a separate HMM is used to segment the sequence during training, and to transform the RNN classification results into label sequences. However, the use of HMM for these purposes does not take advantage of the full potential of RNNs.

An interesting method was proposed in (Graves et al., 2006; Graves, 2012) that enables the RNNs themselves to perform sequence classification, removing the need for pre-segmenting the training data and for post-processing the outputs. Underlying this method is the idea of interpreting RNN outputs as the conditional distributions over all possible label sequences given the input sequences. Then, a differentiable objective function can be derived to optimize these conditional distributions



over the correct label sequences, where no segmentation of data is required. The effectiveness of this method is yet to be demonstrated.

Another type of discriminative deep architecture is convolutional neural network (CNN), with each module consisting of a convolutional layer and a pooling layer. These modules are often stacked up with one on top of another, or with a DNN on top of it, to form a deep model. The convolutional layer shares many weights, and the pooling layer subsamples the output of the convolutional layer and reduces the data rate from the layer below. The weight sharing in the convolutional layer, together with appropriately chosen pooling schemes, endows the CNN with some “invariance” properties (e.g., translation invariance). It has been argued that such limited “invariance” or equi-variance is not adequate for complex pattern recognition tasks and more principled ways of handling a wider range of invariance are needed (Hinton et al., 2011). Nevertheless, CNN has been found highly effective and been commonly used in computer vision and image recognition (Bengio and LeCun, 1995; LeCun et al., 1998; Ciresan et al., 2012; Le et al., 2012; Dean et al., 2012; Krizhevsky et al., 2012). More recently, with appropriate changes from the CNN designed for image analysis to that taking into account speech-specific properties, CNN is also found effective for speech recognition (Abdel-Hamid et al., 2012; Sainath et al., 2013; Deng et al., 2013). We will discuss such applications in more detail in Chapter 7.

It is useful to point out that the time-delay neural network (TDNN, Lang et al., 1990) developed for early speech recognition is a special case and predecessor of the CNN when weight sharing is limited to one of the two dimensions, i.e., time dimension. It was not until recently that researchers have discovered that the time-dimension invariance is less important than the frequency-dimension invariance for speech recognition (Abdel-Hamid et al., 2012; Deng et al., 2013). Analysis and the underlying reasons are described in (Deng et al., 2013), together with a new strategy for designing the CNN’s pooling layer demonstrated to be more effective than all previous CNNs in phone recognition.

It is also useful to point out that the model of hierarchical temporal memory (HTM, Hawkins and Blakeslee, 2004; Hawkins et al., 2010; George, 2008) is another variant and extension of the CNN. The extension includes the following aspects: 1) Time or temporal dimension is introduced to serve as the “supervision” information for discrimination (even for static images); 2) Both bottom-up and top-down information flow are used, instead of just bottom-up in the CNN; and 3) A Bayesian probabilistic formalism is used for fusing information and for decision making.

Finally, the learning architecture developed for bottom-up, detection-based speech recognition proposed in (Lee, 2004) and developed further since 2004, notably in (Yu et al., 2012; Siniscalchi et al., 2013, 2013a) using the DBN-DNN technique, can also be categorized in the discriminative deep architecture category. There is no intent and mechanism in this architecture to characterize the joint probability of data and recognition targets of speech attributes and of the higher-level phone and words. The most current implementation of this approach is based on multiple layers of neural networks using back-propagation learning (Yu et al., 2012). One intermediate neural network layer in the implementation of this detection-based framework explicitly represents the speech attributes, which are simplified entities from the “atomic” units of speech developed in the early work of (Deng and Sun, 1994). The simplification lies in the removal of the temporally overlapping properties of the speech attributes or articulatory-like features. Embedding such more realistic properties in the future work is expected to improve the accuracy of speech recognition further.

## 3.4 Hybrid Generative-Discriminative Architectures

The term “hybrid” for this third category refers to the deep architecture that either comprises or makes use of both generative and discriminative model components. In the existing hybrid architectures published in the literature, the generative component is mostly exploited to help with discrimination, which is the final goal of the hybrid architecture. How and why generative modeling can help with discrimination can be examined from two viewpoints:

- The optimization viewpoint where generative models can provide excellent initialization points in highly nonlinear parameter estimation problems (The commonly used term of “pre-training” in deep learning has been introduced for this reason); and/or
- The regularization perspective where generative models can effectively control the complexity of the overall model.

The study reported in (Erhan et al., 2010) provided an insightful analysis and experimental evidence supporting both of the viewpoints above.

The DBN, a generative deep architecture discussed in Chapter 3.1, can be converted and used as the initial model of a DNN with the same network structure, which is further discriminatively trained or fine-tuned). Some explanation of the equivalence relationship can be found in (Mohamed et al, 2012). When DBN is used this way we consider this DBN-DNN model as a hybrid deep model. We will review details of the DNN in the context of RBM/DBN pre-training as well as its interface with the most commonly used shallow generative architecture of HMM (DNN-HMM) in Chapter 5.

Another example of the hybrid deep architecture is developed in (Mohamed et al., 2010), where the DNN weights are also initialized from a generative DBN but are further fine-tuned with a sequence-level discriminative criterion (conditional probability of the label sequence given the input feature sequence) instead of the frame-level criterion (e.g., cross-entropy) commonly used. This is a combination of the static DNN with the shallow discriminative architecture of CRF. It can be shown that such DNN-CRF is equivalent to a hybrid deep architecture of DNN and HMM whose parameters are learned jointly using the full-sequence maximum mutual information (MMI) criterion between the entire label sequence and the input feature sequence. A closely related full-sequence training method is carried out with success for a shallow neural network (Kingsbury, 2009) and for a deep one (Kingsbury et al., 2012).

Here, it is useful to point out a connection between the above pretraining and fine-tuning strategy and the highly popular minimum phone error (MPE) training technique for the HMM (Povey and Woodland, 2002; and He et al., 2008 for an overview). To make MPE training effective the parameters need to be initialized using an algorithm (e.g., Baum-Welch algorithm) that optimizes a generative criterion (e.g. maximum likelihood).

Along the line of using discriminative criteria to train parameters in generative models as in the above HMM training example, we discuss the same method applied to learning other generative architectures. In (Larochelle and Bengio, 2008), the generative model of RBM is learned using the discriminative criterion of posterior class/label probabilities when the label vector is concatenated with the input data vector to form the overall visible layer in the RBM. In this way, RBM can serve as a stand-alone solution to classification problems and the authors derived a discriminative learning algorithm for RBM as a shallow generative model. In the more recent work of (Ranzato et al., 2011), the deep generative model of DBN with gated Markov random field (MRF) at the lowest level is learned for feature extraction and then for recognition of difficult image classes including occlusions. The generative ability of the DBN facilitates the discovery of what

information is captured and what is lost at each level of representation in the deep model, as demonstrated in (Ranzato et al., 2011). A related work on using the discriminative criterion of empirical risk to train deep graphical models can be found in (Stoyanov et al., 2011).

A further example of the hybrid deep architecture is the use of a generative model to pre-train deep convolutional neural networks (deep CNNs) (Lee et al., 2009, 2010, 2011). Like the fully connected DNN discussed earlier, pre-training also helps to improve the performance of deep CNNs over random initialization.

The final example given here for the hybrid deep architecture is based on the idea and work of (Ney, 1999; He and Deng, 2011), where one task of discrimination (e.g., speech recognition) produces the output (text) that serves as the input to the second task of discrimination (e.g., machine translation). The overall system, giving the functionality of speech translation – translating speech in one language into text in another language – is a two-stage deep architecture consisting of both generative and discriminative elements. Both models of speech recognition (e.g., HMM) and of machine translation (e.g., phrasal mapping and non-monotonic alignment) are generative in nature. But their parameters are all learned for discrimination. The framework described in (He and Deng, 2011) enables end-to-end performance optimization in the overall deep architecture using the unified learning framework initially published in (He et al., 2008). This hybrid deep learning approach can be applied to not only speech translation but also all speech-centric and possibly other information processing tasks such as speech information retrieval, speech understanding, cross-lingual speech/text understanding and retrieval, etc. (e.g., Yamin et al., 2008; Tur et al., 2012; He and Deng, 2012, 2013).

# CHAPTER 4

## GENERATIVE: DEEP AUTOENCODER

---

### 4.1 Introduction

Deep autoencoder is a special type of DNN whose output has the same dimension as the input, and is used for learning efficient encoding or representation of the original data at hidden layers. Note that autoencoder is a nonlinear feature extraction method without using class labels. As such the feature extracted aims at conserving information instead of performing classification tasks, although sometimes these two goals are correlated.

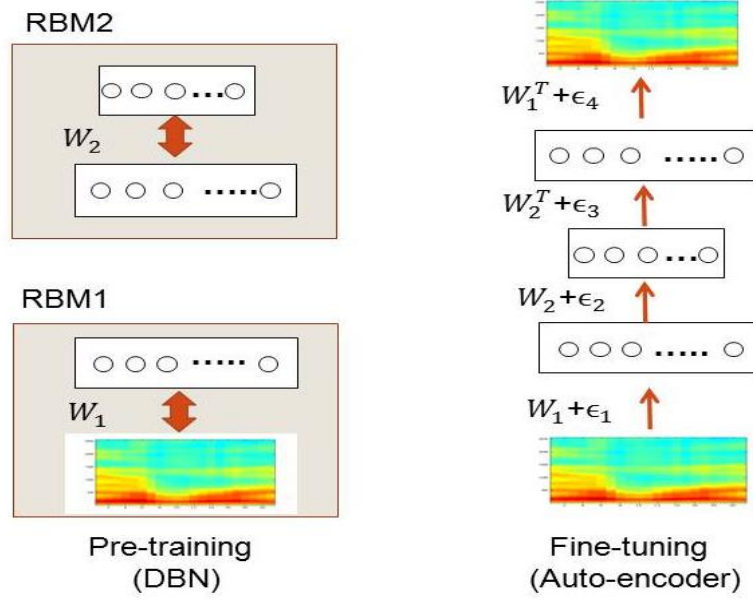
An autoencoder typically has an input layer which represents the original data or feature (e.g., pixels in image or spectra in speech), one or more hidden layers that represent the transformed feature, and an output layer which matches the input layer for reconstruction. When the number of hidden layers is greater than one, the autoencoder is considered to be deep. The dimension of the hidden layers can be either smaller (when the goal is feature compression) or larger (when the goal is mapping the feature to a higher-dimensional space) than the input dimension.

An auto-encoder is often trained using one of the many backpropagation variants (e.g., conjugate gradient method, steepest descent, etc.). Though often reasonably effective, there are fundamental problems when using back-propagation to train networks with many hidden layers. Once the errors get back-propagated to the first few layers, they become minuscule, and training becomes quite ineffective. Though more advanced backpropagation methods (e.g., the conjugate gradient method) help with this to some degree, it still results in very slow learning and poor solutions. As mentioned in the previous chapters this problem can be alleviated by using parameters initialized with some unsupervised pretraining technique such as the DBN pretraining algorithm (Hinton et al, 2006). This strategy has been applied to construct a deep autoencoder to map images to short binary code for fast, content-based image retrieval, to encode documents (called semantic hashing), and to encode spectrogram-like speech features which we review below.

## 4.2 Use of Deep Autoencoder to Extract Speech Features

Here we review a set of work, some of which was published in (Deng et al., 2010), in developing an autoencoder for extracting binary speech codes using unlabeled speech data only. The discrete representations in terms of a binary code extracted by this model can be used in speech information retrieval or as bottleneck features for speech recognition.

A deep generative model of patches of spectrograms that contain 256 frequency bins and 1, 3, 9, or 13 frames is illustrated in Figure 4.1. An undirected graphical model called a Gaussian-Bernoulli RBM is built that has one visible layer of linear variables with Gaussian noise and one hidden layer of 500 to 3000 binary latent variables. After learning the Gaussian-Bernoulli RBM, the activation probabilities of its hidden units are treated as the data for training another Bernoulli-Bernoulli RBM. These two RBM's can then be composed to form a deep belief net (DBN) in which it is easy to infer the states of the second layer of binary hidden units from the input in a single forward pass. The DBN used in this work is illustrated on the left side of Figure 4.1, where the two RBMs are shown in separate boxes. (See more detailed discussions on RBM and DBN in Chapter 5).

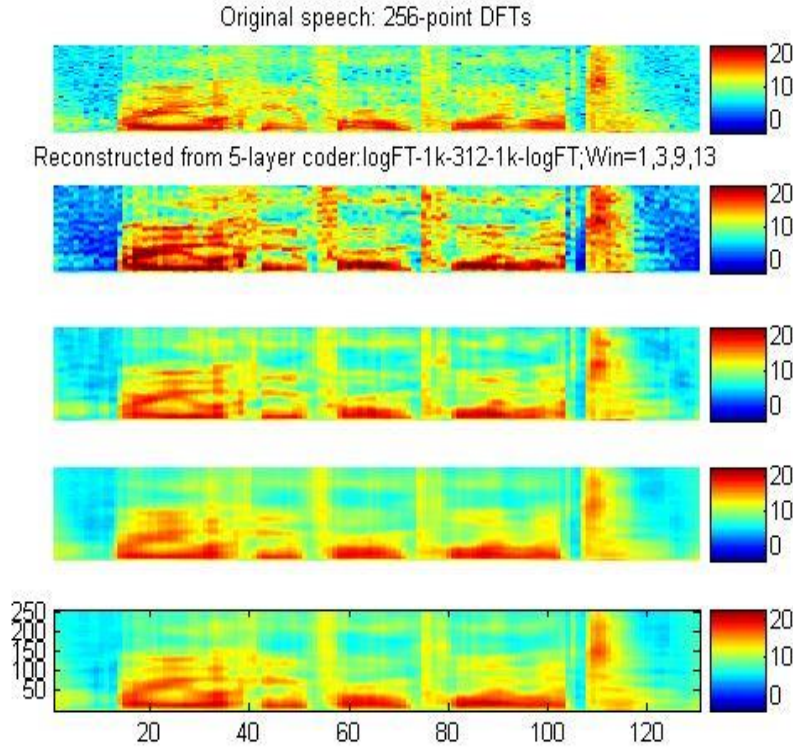


**Figure 4.1:** The architecture of the deep autoencoder used in (Deng et al., 2010) for extracting binary speech codes from high-resolution spectrograms.

The deep autoencoder with three hidden layers is formed by “unrolling” the DBN using its weight matrices. The lower layers of this deep autoencoder use the matrices to encode the input and the upper layers use the matrices in reverse order to decode the input. This deep autoencoder is then fine-tuned using error back-propagation to minimize the reconstruction error, as shown on the right side of Figure 4.1. After learning is complete, any variable-length spectrogram can be encoded and reconstructed as follows. First,  $N$  consecutive overlapping frames of 256-point log power spectra are each normalized to zero-mean and unit-variance to provide the input to the deep autoencoder. The first hidden layer then uses the logistic function to compute real-valued activations. These real values are fed to the next, coding layer to compute “codes”. The real-valued activations of hidden units in the coding layer are quantized to be either zero or one with 0.5 as the threshold. These binary codes are then used to reconstruct the original spectrogram, where individual fixed-frame patches are reconstructed first using the two upper layers of network weights. Finally, overlap-and-add technique is used to reconstruct the full-length speech spectrogram from the outputs

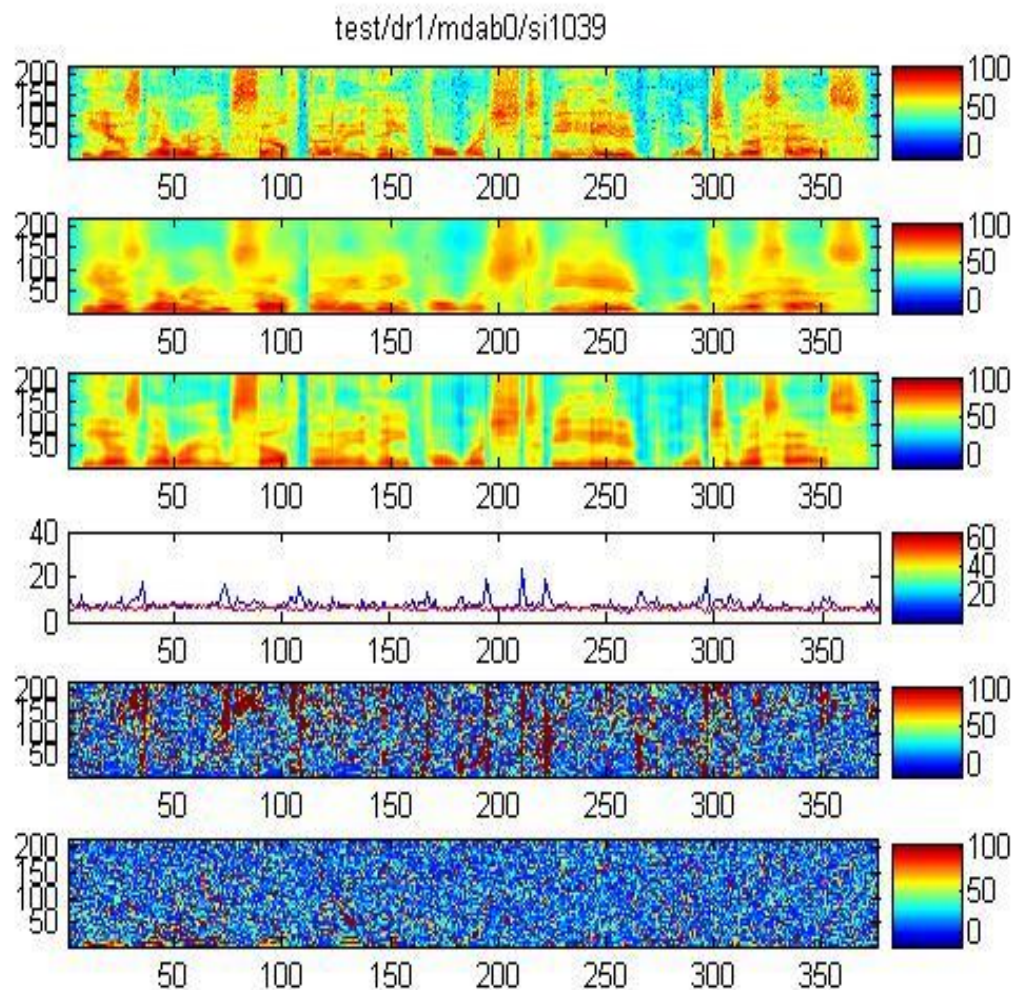
produced by applying the deep autoencoder to every possible window of  $N$  consecutive frames. We show some illustrative encoding and reconstruction examples below.

At the top of Figure 4.2 is the original, un-coded speech, followed by the speech utterances reconstructed from the binary codes (zero or one) at the 312 unit bottleneck code layer with encoding window lengths of  $N=1, 3, 9$ , and  $13$ , respectively. The lower reconstruction errors for  $N=9$  and  $N=13$  are clearly seen.



**Figure 4.2:** Top to Bottom: Original spectrogram; reconstructions using input window sizes of  $N=1, 3, 9$ , and  $13$  while forcing the coding units to be zero or one (i.e., a binary code).

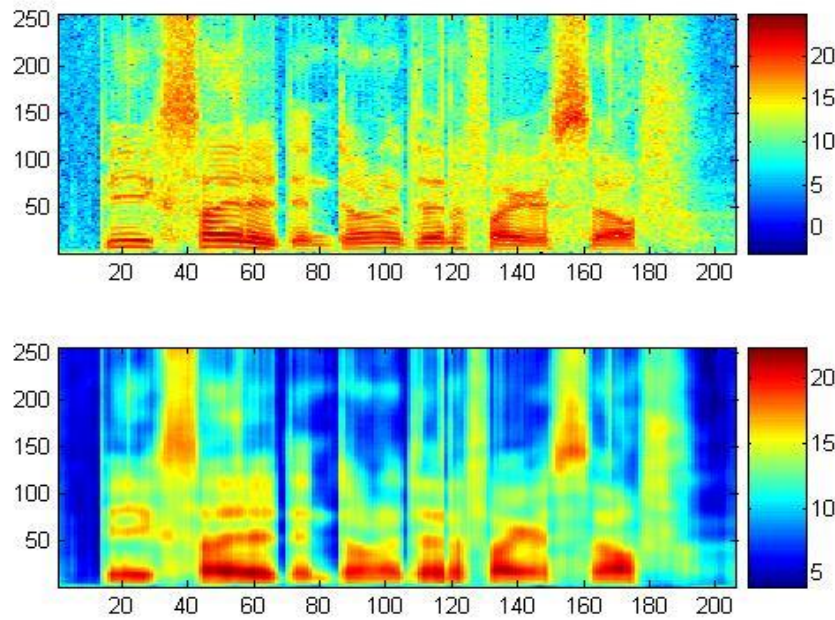




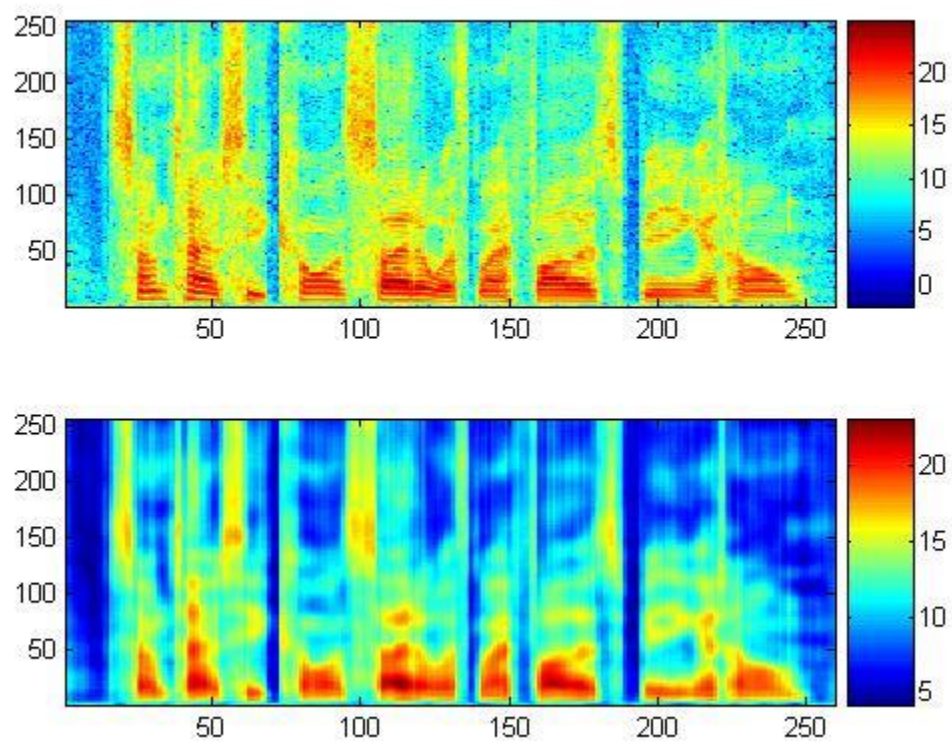
**Figure 4.3:** Top to bottom: Original spectrogram from the test set; reconstruction from the 312-bit VQ coder; reconstruction from the 312-bit auto-encoder; coding errors as a function of time for the VQ coder (blue) and auto-encoder (red); spectrogram of the VQ coder residual; spectrogram of the deep autoencoder's residual.

Encoding error of the deep autoencoder is qualitatively examined in comparison with the more traditional codes via vector quantization (VQ). Figure 4.3 shows various aspects of the encoding errors. At the top is the original speech utterance's spectrogram. The next two spectrograms are the blurry reconstruction from the 312-bit VQ and the much more faithful reconstruction from the 312-bit deep autoencoder. Coding errors from both coders, plotted as a function of time, are shown below the spectrograms, demonstrating that the auto-encoder (red curve) is producing lower errors than the VQ coder (blue curve) throughout the entire span of the utterance. The final two spectrograms show detailed coding error distributions over both time and frequency bins.

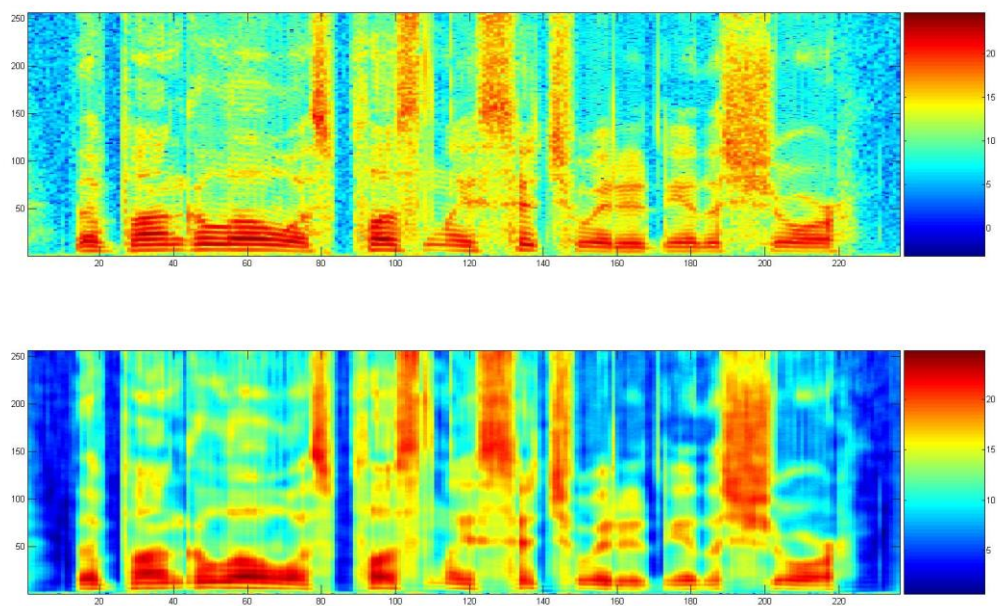
Figures 4.4 to 4.10 show additional examples (unpublished) for the original un-coded speech spectrograms and their reconstructions using the deep autoencoder. They give a diverse number of binary codes for either a single or for three consecutive frames in the spectrogram samples.



**Figure 4.4:** Original speech spectrogram and the reconstructed counterpart. 312 binary codes with one for each single frame.

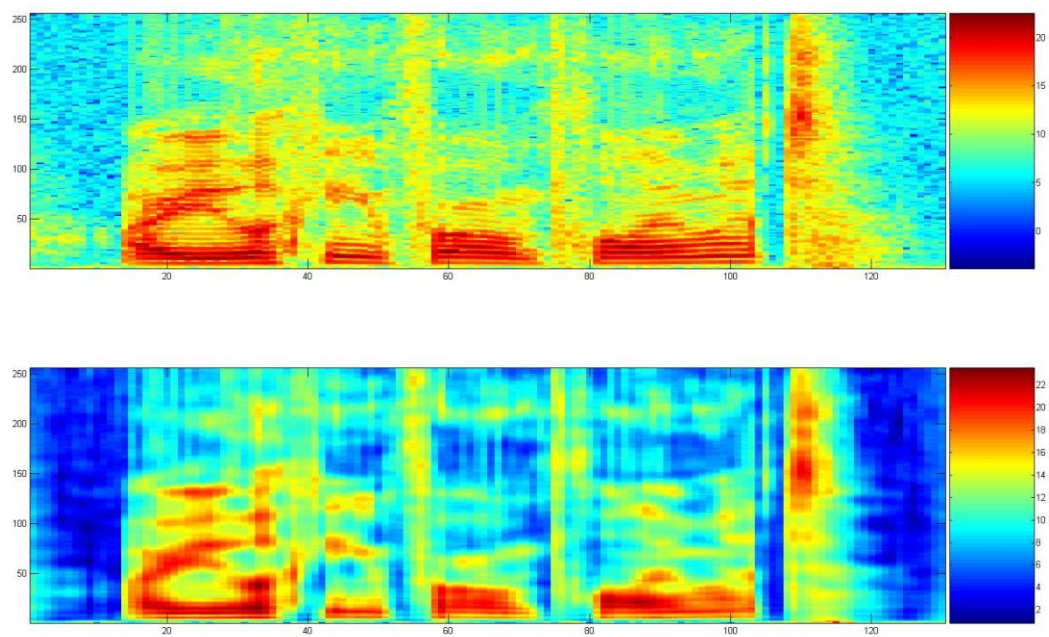


**Figure 4.5:** Same as Figure 4.4 but with a different TIMIT speech utterance.

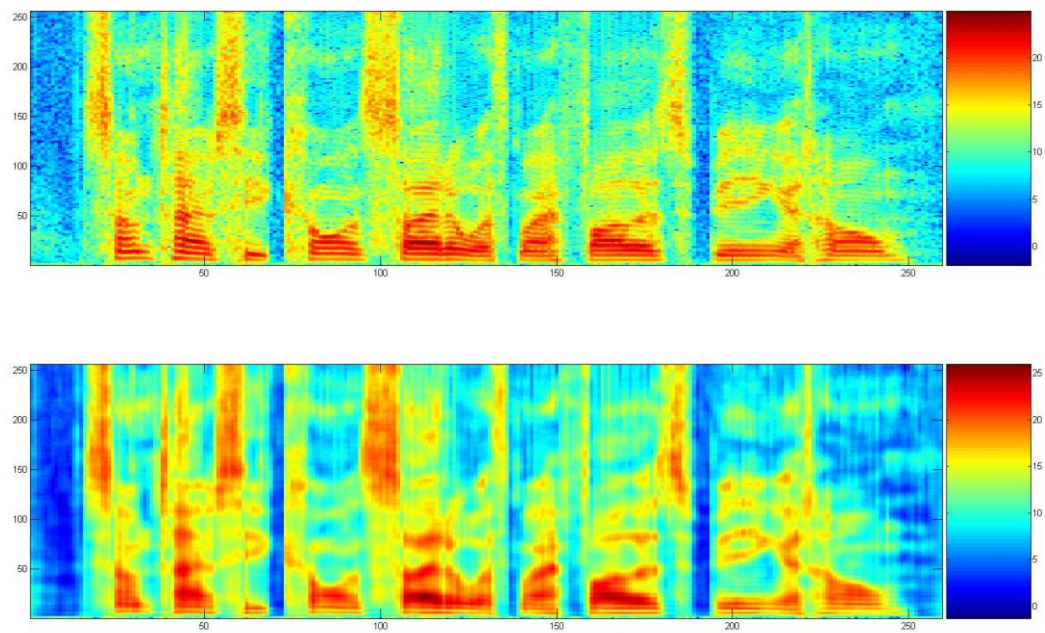


**Figure 4.6:** Original speech spectrogram and the reconstructed counterpart. 936 binary codes for three adjacent frames.

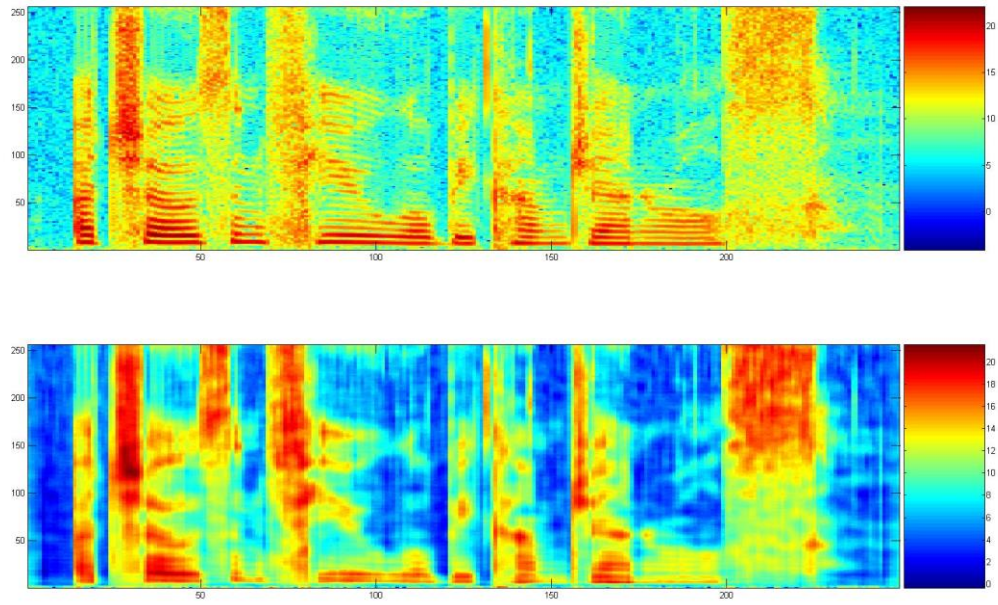




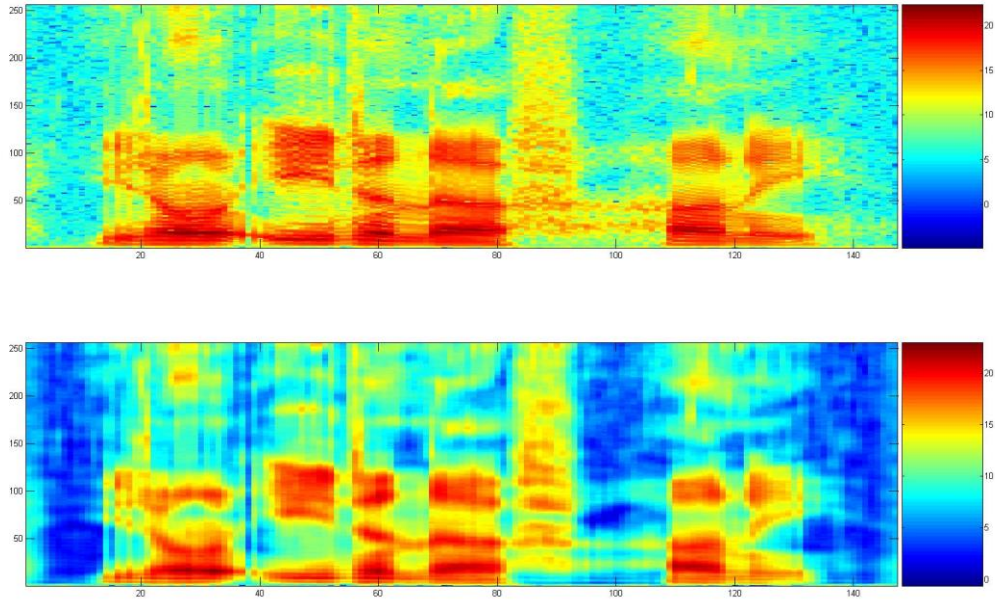
**Figure 4.7:** Same as Figure 4.6 but with a different TIMIT speech utterance.



**Figure 4.8:** Same as Figure 4.6 but with yet another TIMIT speech utterance.



**Figure 4.9:** Original speech spectrogram and the reconstructed counterpart. 2000 binary codes with one for each single frame.



**Figure 4.10:** Same as Figure 4.9 but with a different TIMIT speech utterance.

### 4.3 Stacked Denoising Autoencoder

In most applications the encoding layer has smaller dimension than the input layer in an autoencoder. However, in some applications, it is desirable that the encoding layer is wider than the input layer, in which case some technique is needed to prevent the neural network from learning the trivial identity mapping function. This trivial mapping problem can be prevented by methods such as using sparseness constraints, using the dropout trick (i.e., randomly forcing certain values to be zero) and introducing distortions at the input data (Vincent et al., 2010) or the hidden layers (Hinton et al., 2012).



For example, in the stacked denoising autoencoder (Vincent et al., 2010), random noises are added to the input data. This serves several purposes. First, by forcing the output to match the original undistorted input data the model can avoid learning the trivial identity solution. Second, since the noises are added randomly the model learned would be robust to the same kind of distortions in the test data. Third, since each distorted input sample is different, it essentially greatly increases the training set size and thus can alleviate the overfitting problem. It is interesting to note that when the encoding and decoding weights are forced to be the transpose of each other, such “denoising autoencoder” is approximately equivalent to the RBM when the one-step contrastive divergence algorithm is used to train the RBM (Vincent et al., 2011).

## 4.4 Transforming Autoencoder

The deep autoencoder described above can extract faithful codes for feature vectors due to many layers of nonlinear processing. However, the code extracted this way is transformation-variant. In other words, the extracted code would change unpredictably when the input feature vector is transformed. Sometimes, it is desirable to have the code change predictably to reflect the underlying transformation-invariant property of the perceived content. This is the goal of the transforming auto-encoder proposed in (Hinton et al., 2011) for image recognition.

The building block of the transforming auto-encoder is a “capsule”, which is an independent sub-network that extracts a single parameterized feature representing a single entity, be it visual or audio. A transforming auto-encoder receives both an input vector and a target output vector, which is transformed from the input vector through a simple global transformation mechanism; e.g. translation of an image and frequency shift of speech (the latter due to the vocal tract length difference). An explicit representation of the global transformation is assumed known. The coding layer of the transforming autoencoder consists of the outputs of several capsules.

During the training phase, the different capsules learn to extract different entities in order to minimize the error between the final output and the target.

# CHAPTER 5

## HYBRID: PRE-TRAINED DEEP NEURAL NETWORK

---

In this chapter, we present the most widely used hybrid deep architecture – the pretrained deep neural network (DNN), and discuss the related techniques and building blocks including restricted Boltzmann machine (RBM) and deep belief network. Part of this review is based on the recent publications in (Hinton et al., 2012; Yu and Deng, 2011) and (Dalh et al., 2012).

### 5.1 Restricted Boltzmann Machine

An RBM is a special type of Markov random field that has one layer of (typically Bernoulli) stochastic hidden units and one layer of (typically Bernoulli or Gaussian) stochastic visible or observable units. RBMs can be represented as bipartite graphs, where all visible units are connected to all hidden units, and there are no visible-visible or hidden-hidden connections.

In an RBM, the joint distribution  $p(\mathbf{v}, \mathbf{h}; \theta)$  over the visible units  $\mathbf{v}$  and hidden units  $\mathbf{h}$ , given the model parameters  $\theta$ , is defined in terms of an energy function  $E(\mathbf{v}, \mathbf{h}; \theta)$  of

$$p(\mathbf{v}, \mathbf{h}; \theta) = \frac{\exp(-E(\mathbf{v}, \mathbf{h}; \theta))}{Z},$$

where  $Z = \sum_{\mathbf{v}} \sum_{\mathbf{h}} \exp(-E(\mathbf{v}, \mathbf{h}; \theta))$  is a normalization factor or partition function, and the marginal probability that the model assigns to a visible vector  $\mathbf{v}$  is

$$p(\mathbf{v}; \theta) = \frac{\sum_{\mathbf{h}} \exp(-E(\mathbf{v}, \mathbf{h}; \theta))}{Z}.$$

For a Bernoulli (visible)-Bernoulli (hidden) RBM, the energy function is defined as

$$E(\mathbf{v}, \mathbf{h}; \theta) = - \sum_{i=1}^I \sum_{j=1}^J w_{ij} v_i h_j - \sum_{i=1}^I b_i v_i - \sum_{j=1}^J a_j h_j,$$

where  $w_{ij}$  represents the symmetric interaction term between visible unit  $v_i$  and hidden unit  $h_j$ ,  $b_i$  and  $a_j$  the bias terms, and  $I$  and  $J$  are the numbers of visible and hidden units. The conditional probabilities can be efficiently calculated as

$$p(h_j = 1 | \mathbf{v}; \theta) = \sigma \left( \sum_{i=1}^I w_{ij} v_i + a_j \right),$$

$$p(v_i = 1 | \mathbf{h}; \theta) = \sigma \left( \sum_{j=1}^J w_{ij} h_j + b_i \right),$$

where  $\sigma(x) = 1/(1 + \exp(x))$ .

Similarly, for a Gaussian (visible)-Bernoulli (hidden) RBM, the energy is

$$E(\mathbf{v}, \mathbf{h}; \theta) = - \sum_{i=1}^I \sum_{j=1}^J w_{ij} v_i h_j - \frac{1}{2} \sum_{i=1}^I (v_i - b_i)^2 - \sum_{j=1}^J a_j h_j,$$

The corresponding conditional probabilities become

$$p(h_j = 1 | \mathbf{v}; \theta) = \sigma \left( \sum_{i=1}^I w_{ij} v_i + a_j \right),$$

$$p(v_i|\mathbf{h}; \theta) = \mathcal{N}\left(\sum_{j=1}^J w_{ij} h_j + b_i, 1\right),$$

where  $v_i$  takes real values and follows a Gaussian distribution with mean  $\sum_{j=1}^J w_{ij} h_j + b_i$  and variance one. Gaussian-Bernoulli RBMs can be used to convert real-valued stochastic variables to binary stochastic variables, which can then be further processed using the Bernoulli-Bernoulli RBMs.

The above discussion used two most common conditional distributions for the visible data in the RBM – Gaussian (for continuous-valued data) and binomial (for binary data). More general types of distributions in the RBM can also be used. See (Welling et al., 2005) for the use of general exponential-family distributions for this purpose.

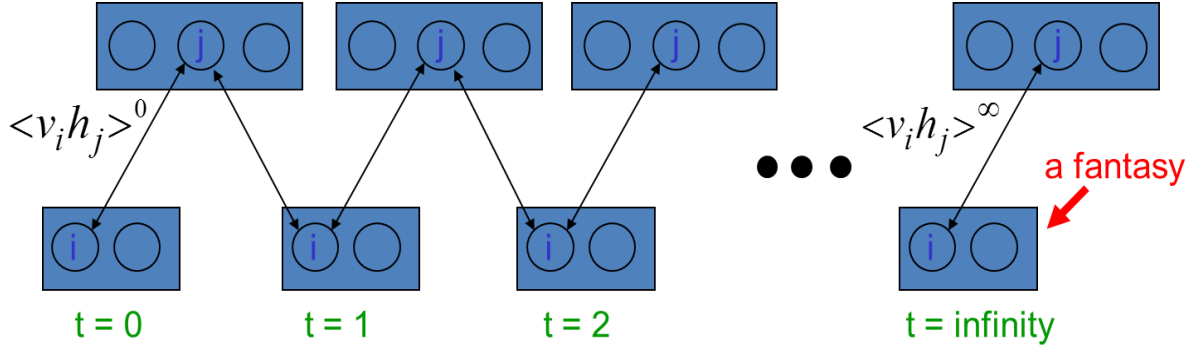
Taking the gradient of the log likelihood  $\log p(\mathbf{v}; \theta)$  we can derive the update rule for the RBM weights as:

$$\Delta w_{ij} = E_{data}(v_i h_j) - E_{model}(v_i h_j),$$

where  $E_{data}(v_i h_j)$  is the expectation observed in the training set and  $E_{model}(v_i h_j)$  is that same expectation under the distribution defined by the model. Unfortunately,  $E_{model}(v_i h_j)$  is intractable to compute so the contrastive divergence (CD) approximation to the gradient is used where  $E_{model}(v_i h_j)$  is replaced by running the Gibbs sampler initialized at the data for one full step. The steps in approximating  $E_{model}(v_i h_j)$  is as follows:

- Initialize  $\mathbf{v}_0$  at data
- Sample  $\mathbf{h}_0 \sim p(\mathbf{h}|\mathbf{v}_0)$
- Sample  $\mathbf{v}_1 \sim p(\mathbf{v}|\mathbf{h}_0)$
- Sample  $\mathbf{h}_1 \sim p(\mathbf{h}|\mathbf{v}_1)$

Then  $(\mathbf{v}_1, \mathbf{h}_1)$  is a sample from the model, as a very rough estimate of  $E_{model}(v_i h_j)$ . Use of  $(\mathbf{v}_1, \mathbf{h}_1)$  to approximate  $E_{model}(v_i h_j)$  gives rise to the algorithm of CD-1. And the sampling process can be pictorially depicted as below in Figure 5.1 below.



**Figure 5.1:** A pictorial view of sampling from a RBM during RBM learning (courtesy of Geoff Hinton).

Careful training of RBMs is essential to the success of applying RBM and related deep learning techniques to solve practical problems. See the Technical Report (Hinton 2010) for a very useful practical guide for training RBMs.

The RBM discussed above is a generative model, which characterizes the input data distribution using hidden variables and there is no label information involved. However, when the label information is available, it can be used together with the data to form the joint “data” set. Then the same CD learning can be applied to optimize the approximate “generative” objective function related to data likelihood. Further, and more interestingly, a “discriminative” objective function can be defined in terms of conditional likelihood of labels. This discriminative RBM can be used to “fine tune” RBM for classification tasks (Larochelle and Bengio, 2008).

Ranzato et al. (2007) proposed an unsupervised learning algorithm called Sparse Encoding Symmetric Machine (SESM), which is quite similar to RBM. They both have a symmetric encoder and decoder, and a logistic non-linearity on the top of the encoder. The main difference is that RBM is trained using (approximate) maximum likelihood, but SESM is trained by simply

minimizing the average energy plus an additional code sparsity term. SESM relies on the sparsity term to prevent flat energy surfaces, while RBM relies on an explicit contrastive term in the loss, an approximation of the log partition function. Another difference is in the coding strategy in that the code units are “noisy” and binary in RBM, while they are quasi-binary and sparse in SESM.

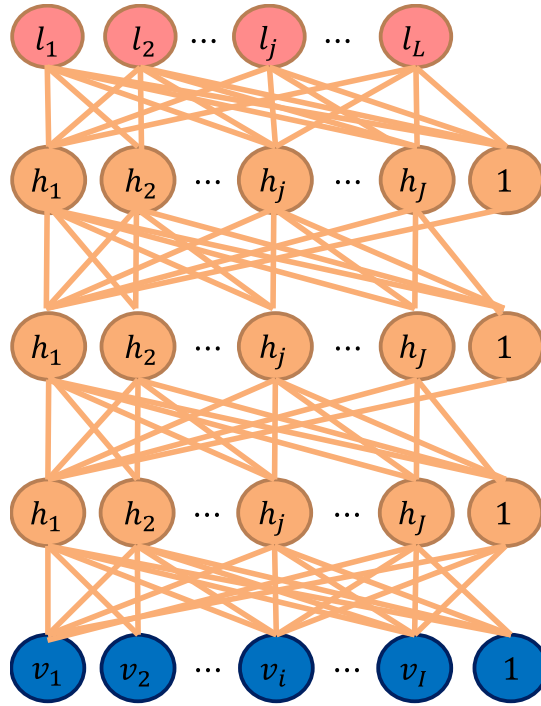
## 5.2 Stacking up RBMs to Form a DBN/DNN

Stacking a number of the RBMs learned layer by layer from bottom up gives rise to a DBN, an example of which is shown in Figure .2. The stacking procedure is as follows. After learning a Gaussian-Bernoulli RBM (for applications with continuous features such as speech) or Bernoulli-Bernoulli RBM (for applications with nominal or binary features such as black-white image or coded text), we treat the activation probabilities of its hidden units as the data for training the Bernoulli-Bernoulli RBM one layer up. The activation probabilities of the second-layer Bernoulli-Bernoulli RBM are then used as the visible data input for the third-layer Bernoulli-Bernoulli RBM, and so on. Some theoretical justification of this efficient layer-by-layer greedy learning strategy is given in (Hinton et al., 2006), where it is shown that the *stacking* procedure above improves a variational lower bound on the likelihood of the training data under the composite model. That is, the greedy procedure above achieves approximate maximum likelihood learning. Note that this learning procedure is unsupervised and requires no class label.

When applied to classification tasks, the generative pre-training can be followed by or combined with other, typically discriminative, learning procedures that fine-tune all of the weights jointly to improve the performance of the network. This discriminative fine-tuning is performed by adding a final layer of variables that represent the desired outputs or labels provided in the training data. Then, the back-propagation algorithm can be used to adjust or fine-tune the network weights in the same way as for the standard feed-forward neural network. What goes to the top, label layer of this DNN depends on the application. For speech recognition applications, the top layer, denoted by “ $l_1, l_2, \dots, l_j, \dots, l_L$ ,” in Figure .2, can represent either syllables, phones, sub-phones, phone states, or other speech units used in the HMM-based speech recognition system.

The generative pre-training described above has produced better phone and speech recognition results than random initialization on a wide variety of tasks, which will be surveyed in Chapter 7. Further research has also shown the effectiveness of other pre-training strategies. As an example, greedy layer-by-layer training may be carried out with an additional discriminative term to the

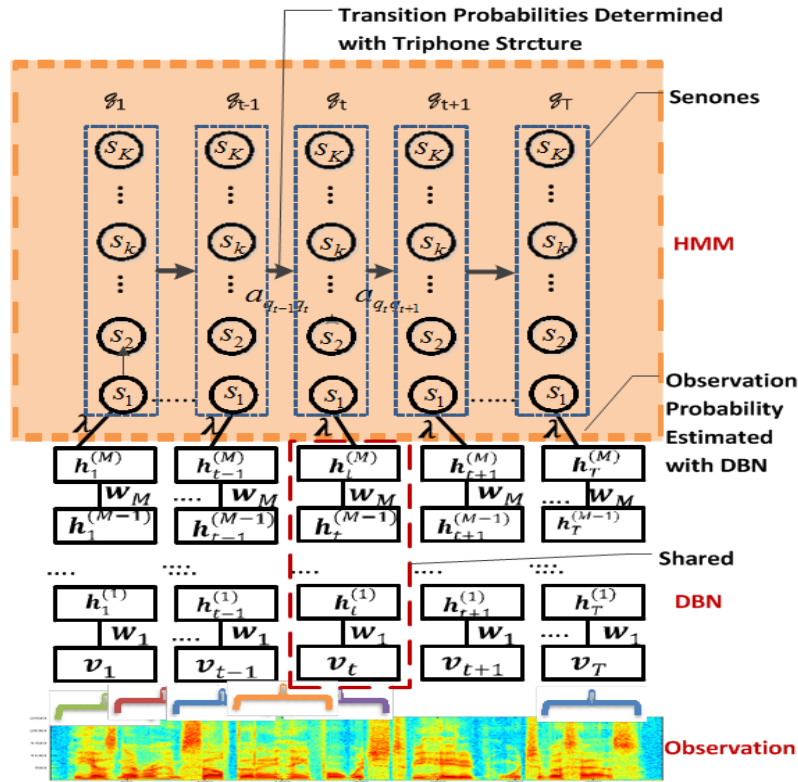
generative cost function at each level. And without generative pre-training, purely discriminative training of DNNs from random initial weights using the traditional stochastic gradient decent method has been shown to work very well when the scales of the initial weights are set carefully and the mini-batch sizes, which trade off noisy gradients with convergence speed, used in stochastic gradient decent are adapted prudently (e.g., with an increasing size over training epochs). Also, randomization order in creating mini-batches needs to be judiciously determined. Importantly, it was found effective to learn a DNN by starting with a shallow neural net with a single hidden layer. Once this has been trained discriminatively (using early stops to avoid overfitting), a second hidden layer is inserted between the first hidden layer and the labeled softmax output units and the expanded deeper network is again trained discriminatively. This can be continued until the desired number of hidden layers is reached, after which a full backpropagation “fine tuning” is applied. This discriminative “pre-training” procedure is found to work well in practice (e.g., Seide et al., 2011; Yu et al, 2011).



**Figure 5.2:** An illustration of a DBN/DNN architecture.

## 5.3 Interfacing DNN with HMM

The DNN discussed above is a static classifier with input vectors having a fixed dimensionality. However, many practical pattern recognition and information processing problems, including speech recognition, machine translation, natural language understanding, video processing and bio-information processing, require sequence recognition. In sequence recognition, sometimes called classification with structured input/output, the dimensionality of both inputs and outputs are variable.



**Figure 5.3:** Interface between DBN/DNN and HMM to form a DBN-HMM or DNN-HMM. This architecture has been successfully used in speech recognition experiments reported in (Dahl et al., 2012).



The HMM, based on dynamic programming operations, is a convenient tool to help port the strength of a static classifier to handle dynamic or sequential patterns. Thus, it is natural to combine DNN and HMM to bridge the gap between the static and sequence pattern recognition. A popular architecture to fulfill this is shown in 5.3. This architecture has been successfully used in speech recognition experiments as reported in (Dahl et al., 2012).

It is important to note that the unique elasticity of temporal dynamic of speech as elaborated in (Deng, 2006) would require temporally-correlated models powerful than HMM for the ultimate success of speech recognition. Integrating such dynamic models having realistic co-articulatory properties with the DNN and possibly other deep learning models to form the coherent dynamic deep architecture is a challenging new research.

# CHAPTER 6

## DISCRIMINATIVE: DEEP STACKING NETWORKS AND VARIANTS

---

### 6.1 Introduction

While the DNN just reviewed has been shown to be extremely powerful in connection with performing recognition and classification tasks including speech recognition and image classification, training a DNN has proven to be difficult computationally. In particular, conventional techniques for training DNN at the fine tuning phase involve the utilization of a stochastic gradient descent learning algorithm, which is extremely difficult to parallelize across machines. This makes learning at large scale practically impossible. For example, it has been possible to use one single, very powerful GPU machine to train DNN-based speech recognizers with dozens to a few hundreds of hours of speech training data with remarkable results. It is very difficult, however, to scale up this success with thousands or more hours of training data.

Here we describe a new deep learning architecture, Deep Stacking Network (DSN), which attacks the learning scalability problem. This chapter is based in part on the recent publications of (Deng and Yu, 2011; Deng et al., 2012a; Hutchinson et al., 2012, 2013) with expanded discussions.

The central idea of DSN design relates to the concept of stacking, as proposed originally in (Wolpert, 1992), where simple modules of functions or classifiers are composed first and then they are “stacked” on top of each other in order to learn complex functions or classifiers. Various ways of implementing stacking operations have been developed in the past, typically making use of supervised information in the simple modules. The new features for the stacked classifier at a higher level of the stacking architecture often come from concatenation of the classifier output of a lower module and the raw input features. In (Cohen and de Carvalho, 2005), the simple module used for stacking was a conditional random field (CRF). This type of deep architecture was further

developed with hidden states added for successful natural language and speech recognition applications where segmentation information is unknown in the training data (Yu et al., 2010). Convolutional neural networks, as in (Jarrett, et al., 2009), can also be considered as a stacking architecture but the supervision information is typically not used until in the final stacking module.

The DSN architecture was originally presented in (Deng and Yu, 2011), which also used the name Deep Convex Network or DCN to emphasize the convex nature of the principal learning algorithm used for learning the network. DSN makes use of supervision information for stacking each of the basic modules, which takes the simplified form of multilayer perceptron. In the basic module, the output units are linear and the hidden units are sigmoidal nonlinear. The linearity in the output units permits highly efficient, parallelizable, and closed-form estimation (a result of convex optimization) for the output network weights given the hidden units' activities. Due to the closed-form constraints between the input and output weights, the input weights can also be elegantly estimated in an efficient, parallelizable, batch-mode manner.

The name “convex” used in (Deng and Yu, 2011) accentuates the role of convex optimization in learning the output network weights given the hidden units' activities in each basic module. It also points to the importance of the closed-form constraints, derived from the convexity, between the input and output weights. Such constraints make the learning of the remaining network parameters (i.e., the input network weights) much easier than otherwise, enabling batch-mode learning of DSN that can be distributed over CPU clusters. And in more recent publications, DSN was used when the key operation of stacking is emphasized.

## 6.2 Architecture of DSN

A DSN, as shown in Figure 6.1, includes a variable number of layered modules, wherein each module is a specialized neural network consisting of a single hidden layer and two trainable sets of weights. In 6.1, only four such modules are illustrated, where each module is shown with a separate color. In practice, up to a few hundreds of modules have been efficiently trained and used in image and speech classification experiments.

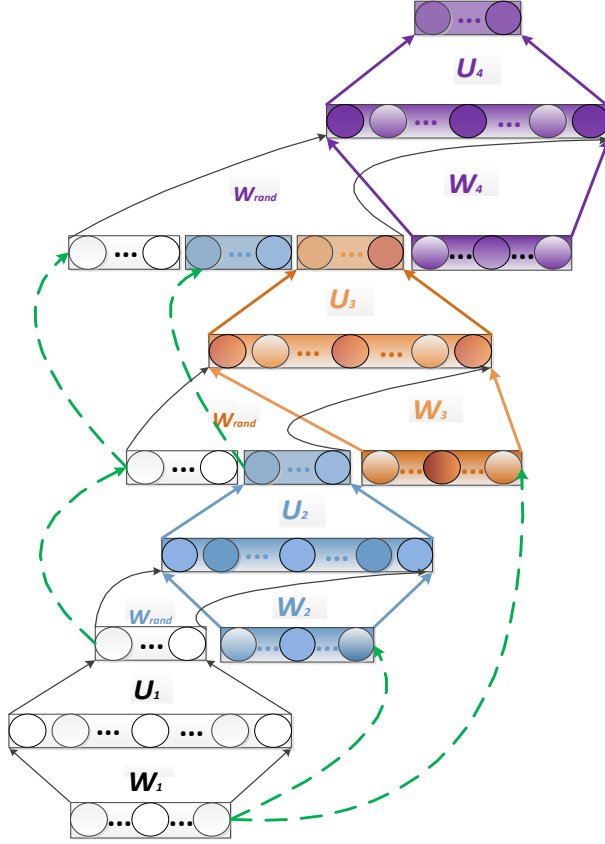
The lowest module in the DSN comprises a linear layer with a set of linear input units, a non-linear layer with a set of non-linear hidden units, and a second linear layer with a set of linear output units. Sigmoidal nonlinearity is typically used in the hidden layer. However, other nonlinearities

can also be used. If the DSN is utilized in connection with recognizing an image, the input units can correspond to a number of pixels (or extracted features) in the image, and can be assigned values based at least in part upon intensity values, RGB values, or the like corresponding to the respective pixels. If the DSN is utilized in connection with speech recognition, the set of input units may correspond to samples of speech waveform, or the extracted features from speech waveforms, such as power spectra or cepstral coefficients. The output units in the linear output layer represent the targets of classification. For instance, if the DSN is configured to perform digit recognition, then the output units may be representative of the values 0, 1, 2, 3, and so forth up to 9 with a 0-1 coding scheme. If the DSN is configured to perform speech recognition, then the output units may be representative of phones, HMM states of phones, or context-dependent HMM states of phones.

The lower-layer weight matrix, which we denote by  $W$ , connects the linear input layer and the hidden nonlinear layer. The upper-layer weight matrix, which we denote by  $U$ , connects the nonlinear hidden layer with the linear output layer. The weight matrix  $U$  can be determined through a closed-form solution given the weight matrix  $W$ .

As indicated above, the DSN includes a set of serially connected, overlapping, and layered modules, wherein each module has the same architecture – a linear input layer followed by a nonlinear hidden layer, which is connected to a linear output layer. Note that the output units of a lower module are a subset of the input units of an adjacent higher module in the DSN. More specifically, in a second module that is directly above the lowest module in the DSN, the input units can include the output units of the lowest module and optionally the raw input feature.

This pattern of including output units in a lower module as a portion of the input units in an adjacent higher module and thereafter learning a weight matrix that describes connection weights between hidden units and linear output units via convex optimization can continue for many modules. A resultant learned DSN may then be deployed in connection with an automatic classification task such as frame-level speech phone or state classification. Connecting DSN's output to an HMM or any dynamic programming device enables continuous speech recognition and other forms of sequential pattern recognition.

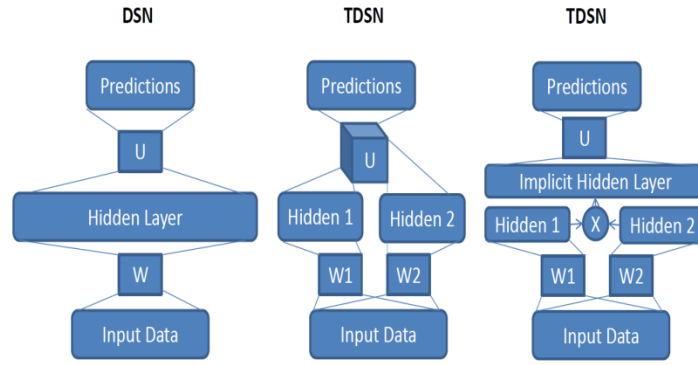


**Figure 6.1:** A DSN architecture. Only four modules are illustrated, each with a distinct color. Dashed lines denote copying layers.

## 6.3 Tensor Deep Stacking Network

The above DSN architecture has recently been generalized to its tensorized version, which we call Tensor DSN (TDSN) (Hutchinson et al., 2012, 2013). It has the same scalability as DSN in terms of parallelizability in learning, but it generalizes DSN by providing higher-order feature interactions missing in DSN.

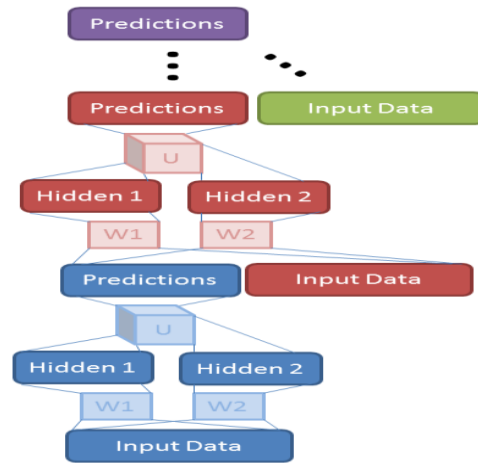
The architecture of TDSN is similar to that of DSN in the way that stacking operation is carried out. That is, modules of the TDSN are stacked up in a similar way to form a deep architecture. The differences of TDSN and DSN lie mainly in how each module is constructed. In DSN, we have one set of hidden units forming a hidden layer, as denoted at the left panel of Figure 6.2. In contrast, each module of a TDSN contains two independent hidden layers, denoted as “Hidden 1” and “Hidden 2” in the middle and right panels of Figure 6.2. As a result of this difference, the upper-layer weights, denoted by “U” in Figure 6.2, changes from a matrix (a two dimensional array) in DSN to a tensor (a three dimensional array) in TDSN, shown as a cube labeled by “U” in the middle panel.



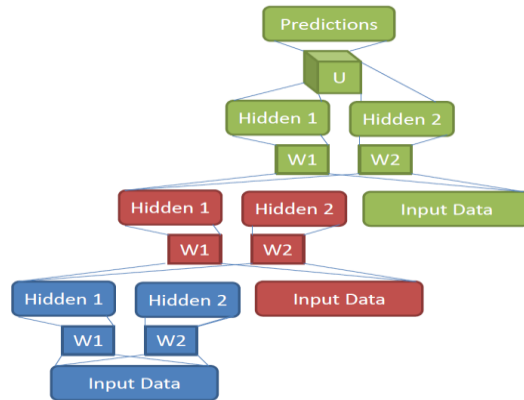
**Figure 6.2:** Comparisons of a single module of a DSN (left) and that of a tensorized-DSN (TDSN). Two equivalent forms of a TDSN module are shown to the right.

The tensor  $U$  has a three-way connection, one to the prediction layer and the remaining to the two separate hidden layers. An equivalent form of this TDSN module is shown in the right panel of Figure 6.2, where the implicit hidden layer is formed by expanding the two separate hidden layers into their outer product. The resulting large vector contains all possible pair-wise products for the two sets of hidden-layer vectors. This turns tensor  $U$  into a matrix again whose dimensions are 1) size of the prediction layer; and 2) product of the two hidden layers’ sizes. Such equivalence enables the same convex optimization for learning  $U$  developed for DSN to be applied to learning tensor  $U$ . Importantly, higher-order hidden feature interactions are enabled in TDSN via the outer product construction for the large, implicit hidden layer.

Stacking TDSN modules to form a deep architecture pursues in a similar way to DSN by concatenating various vectors. Two examples are shown in Figure 6.3 and Figure 6.4. Note stacking by concatenating hidden layers with input (Figure 6.4) would be difficult for DSN since its hidden layer tends to be too large for practical purposes.



**Figure 6.3:** Stacking of TDSN modules by concatenating prediction vector with input vector.



**Figure 6.4:** Stacking of TDSN modules by concatenating two hidden-layers' vectors with the input vector.

# CHAPTER 7

## SELECTED APPLICATIONS IN SPEECH RECOGNITION

---

The traditional neural network (e.g., MLP) has been in use for speech recognition for many years. When used alone, its performance is typically lower than the state-of-the-art HMM systems with observation probabilities approximated with Gaussian mixture models (GMMs). Since a few years ago, the deep learning technique has been successfully applied to phone recognition (Mohamed et al., 2009, 2010, 2012; Sivaram and Hermansky, 2012; Deng et al., 2013) and large vocabulary speech recognition tasks (Yu et al., 2010, 2012; Seide et al., 2011; Dahl et al., 2011, 2012; Kubo et al., 2012; Deng et al., 2013a) by integrating the powerful discriminative training ability of the DNNs with the sequential modeling ability of the HMMs.

More specifically, the work of (Mohamed et al., 2009) was presented at the NIPS-2009 Workshop (Deng et al., 2009), and carried out and assisted by University of Toronto and MSR speech researchers. In this work, a five-layer DNN (called the DBN in the paper) was used to replace the Gaussian mixture component of the GMM-HMM, and the monophone state was used as the modeling unit. Although monophones are generally accepted as a weaker phonetic representation than triphones, the DBN-HMM approach with monophones was shown to achieve higher phone recognition accuracy than the state-of-the-art triphone GMM-HMM systems. Further, the DNN results were found to be slightly superior to the then-best-performing single system based on the hidden trajectory model (HTM) in the literature (Deng et al., 2006, 2007) evaluated on the same TIMIT task. At the Microsoft Research Redmond lab, the error patterns produced by these two separate systems (DNN vs. HTM) were carefully analyzed and found to be very different, reflecting distinct core capabilities of the two approaches and igniting further studies on the DNN approach described below.

The technique of (Mohamed et al., 2009) was improved in the later work reported in (Mohamed et al., 2010) by using the CRF instead of the HMM to model the sequential speech data and by



applying the maximum mutual information (MMI) training technique successfully developed in speech recognition to the resultant DBN-CRF training. The sequential discriminative learning technique developed jointly optimizes the DBN weights, transition weights, and phone language model and achieved higher accuracy than the DBN-HMM phone recognizer with the frame-discriminative training criterion implicit in the DBN’s fine tuning procedure implemented in (Mohamed et al., 2009).

Here we elaborate on the method of (Mohamed et al., 2010). The early phonetic recognition experiments made use of the standard frame-based objective function in static pattern classification, cross-entropy, to optimize the DNN weights. The transition parameters and language model scores were obtained from an HMM and were trained independently of the DNN weights. However, it has been known during the long history of HMM research that sequence classification criteria can be very helpful in improving speech and phone recognition accuracy. This is because the sequence classification criteria are more directly correlated with the performance measure (e.g., the overall word or phone error rate) than cross entropy.

The benefit of using such sequence classification criteria was shown on shallow neural networks in (Kingsbury 2009; Prabhavalkar and Fosler-Lussier, 2010). In more recent work of (Mohamed et al., 2010), one popular type of sequence classification criterion, maximum mutual information or MMI, was successfully applied to learn DNN weights for the TIMIT phone recognition task. Use of cross entropy to train DNN for phone sequence recognition does not explicitly take into account the fact that the neighboring frames have smaller distances between the assigned probability distributions over phone class labels. To overcome this deficiency, one can optimize the conditional probability of the whole sequence of labels, given the whole visible feature utterance or equivalent the hidden feature sequence extracted by DNN. To optimize the log conditional probability on the training data, we take the gradient over the activation parameters, transition parameters and lower-layer weights, and then pursue back-propagation of the error defined at the sentence level.

In implementing the above learning algorithm for MMI-DNN, the DNN weights can be initialized using the frame-based training with cross entropy as the objective. The transition parameters can be initialized from the combination of the HMM transition matrices and the “phone language” model scores, and can be further optimized by tuning the transition features while fixing the DNN weights before the joint optimization. Using the joint optimization with careful scheduling, it was

shown that MMI-DNN can outperform the DNN trained with cross entropy by approximately 5% relative.

In (Dahl et al., 2011, 2012), the DNN-HMM was extended from the monophone phonetic representation to the triphone or context-dependent counterpart and from phone recognition to large vocabulary speech recognition. Experiments on the Bing mobile voice search dataset collected under the real usage scenario demonstrate that the triphone DNN-HMM significantly outperforms the state-of-the-art HMM system. Three factors additional to the DNN contribute to the success: the use of triphones as the DNN modeling units, the use of the best available tri-phone GMM-HMM to generate the alignment with each state in the tri-phones, and the tuning of the transition probabilities. Experiments also indicate that the decoding time of a five-layer DNN-HMM is almost the same as that of the state-of-the-art triphone GMM-HMM.

In other large vocabulary speech recognition tasks including those defined on Google voice search and YouTube, and in the Switchboard and Broadcast News databases, DNN-HMM also produces strong results (Seide et al., 2011; Sainath et al., 2011; Jaitly et al., 2012; Hinton et al., 2012). For example, on the Switchboard benchmark, the context-dependent DNN-HMM (CD-DNN-HMM) cut error by one third compared to the state-of-the-art GMM-HMM system (Seide et al., 2011).

A similar strength of DNN (called DBN in the paper) was reported in (Vinyals and Ravuri, 2011) in a somewhat different setup called Tandem approach and for mis-matched noisy speech. It was reported that DNNs outperform the MLPs with a single hidden layer under the clean condition, but the gains slowly diminish as the noise level is increased. Furthermore, using MFCCs in conjunction with the posteriors computed from DNNs outperforms using single DNNs alone in low to moderate noise conditions under the Tandem architecture.

The more recent work of (Deng and Yu, 2011; Yu and Deng, 2011; Deng, Yu, and Platt, 2012), which we reviewed in Chapter 6, makes use of the discriminative DSN architecture to perform frame-level phone classification as well as phone recognition. Higher accuracy than DBN/DNN is reported, especially after a discriminative “fine-tuning” technique developed in (Yu and Deng, 2011) is exploited. The effectiveness of DSN is particularly enabled by convex optimization for discriminative “pre-training”, and is recently found to beat strong baseline results also in a speech understanding task (Tur et al, 2012).

The basic DSN architecture has recently been extended to its tensor version, TDSN, as reviewed in Chapter 6, to incorporate correlation structure in the hidden layer activities with promising results on phone recognition and computational properties reported in (Hutchinson et al., 2012). While the work reported in these papers has not fully developed parallel implementation of the basic learning algorithm in the DSN and TDSN architectures, research is currently underway to enable high scalability of learning DSN and TDSN via parallelization, with initial results shown in (Deng et al., 2012b).

In the work reported in (Lee et al., 2009) and the follow-up work, the convolutional structure is imposed on the RBM while building up a DBN. Convolution is made in time by sharing weights between hidden units in an attempt to detect the same “invariant” feature over different times. Then a max-pooling operation is performed where the maximal activations over small temporal neighborhoods of hidden units are obtained, proving some local temporal invariance. The resulting convolutional DBN is applied to audio and speech data for a number of tasks including music artist and genre classification, speaker identification, speaker gender classification, and phone classification, with promising results presented.

As discussed in Chapter 3.2, the concept of convolution in time in the above work was originated in TDNN as a shallow neural net (Lang et al., 1990) developed in early speech recognition. Only recently and when deep architectures (e.g. deep CNN) are used, it has been found that frequency-dimension weight sharing is more effective for high-performance phone recognition than time domain as in the previous TDNN (Abdel-Hamid et al., 2012; Deng et al., 2013). These studies also show that designing the pooling in deep CNN to properly trade-off between invariance to vocal tract length and discrimination between speech sounds (together with a regularization technique of “dropout” (Hinton et al., 2012) leads to even better phone recognition performance. This set of work also points to the direction of trading-off between trajectory discrimination and invariance expressed in the whole dynamic pattern of speech defined in mixed time and frequency domains using convolution and pooling. Moreover, the most recent work of (Sainath et al., 2013) shows that CNNs are also useful for large vocabulary continuous speech recognition and further demonstrates that multiple convolutional layers provide even more improvement when the convolutional layers use a large number of convolution kernels or feature maps.

The more recent work reported in (Jaitly and Hinton, 2011) makes use of speech sound waves as the raw input feature to an RBM with a convolutional structure as the classifier. With the use of rectifier linear units in the hidden layer (Glorot et al., 2011), it is possible to automatically

normalize the amplitude variation in the waveform signal, thus overcoming the difficulty encountered in the earlier attempt of using the same raw feature in the HMM-based approach (Sheikhzadeh and Deng, 1994).

In addition to the RBM, DNN, and DSN, other deep models have also been developed and reported in the literature for speech processing and related applications. For example, the deep-structured CRF, which stacks many layers of CRFs, have been successfully used in the task of language identification (Yu et al., 2010), phone recognition (Yu and Deng, 2010), sequential labeling in natural language processing (Yu et al., 2010a), and confidence calibration in speech recognition (Yu et al., 2010b). Further, while RNN has early success in phone recognition (Robinson, 1994), it was not easy to duplicate due to the intricacy in training, let alone to scale up for larger speech recognition tasks. Learning algorithms for the RNN have been dramatically improved since then, and much better results have been obtained recently using the RNN (Graves, et al, 2006; 2013). The RNN has also been recently applied to music processing applications (Bengio et al., 2013), where the use of rectified linear hidden units instead of logistic or tanh nonlinearities is explored in the RNN. Rectified linear units compute  $y = \max(x, 0)$ , and lead to sparser gradients, less diffusion of credit and blame in the RNN, and faster training.

# CHAPTER 8

## SELECTED APPLICATIONS IN LANGUAGE MODELING

---

Research in language, document, and text processing has seen increasing popularity recently in the signal processing community, and has been designated as one of the main focus areas by the society's audio, speech, and language processing technical committee. There has been a long history (e.g., Bengio et al., 2000; Zamora et al., 2009) of using (shallow) neural networks in language modeling (LM) – an important component in speech recognition, machine translation, text information retrieval, and in natural language processing. Recently, deep neural networks have been attracting more and more attention in statistical language modeling.

An LM is a function that captures the salient statistical characteristics of the distribution of sequences of words in a natural language. It allows one to make probabilistic predictions of the next word given preceding ones. A neural network LM is one that exploits the neural network ability to learn distributed representations to reduce the impact of the curse of dimensionality.

A distributed representation of a symbol is a vector of features which characterize the meaning of the symbol. Each element in the vector participates in representing the meaning. With a neural network LM, one relies on the learning algorithm to discover meaningful, continuous-valued features. The basic idea is to learn to associate each word in the dictionary with a continuous-valued vector representation, where each word corresponds to a point in a feature space. One can imagine that each dimension of that space corresponds to a semantic or grammatical characteristic of words. The hope is that functionally similar words get to be closer to each other in that space, at least along some directions. A sequence of words can thus be transformed into a sequence of these learned feature vectors. The neural network learns to map that sequence of feature vectors to the probability distribution over the next word in the sequence.

The distributed representation approach to LM has the advantage that it allows the model to generalize well to sequences that are not in the set of training word sequences, but that are similar in terms of their features, i.e., their distributed representation. Because neural networks tend to map nearby inputs to nearby outputs, the predictions corresponding to word sequences with similar features are mapped to similar predictions.

The above ideas of neural network LM have been implemented in various studies, some involving deep architecture. In (Mnih and Hinton, 2007), temporally factored RBM was used for language modeling. Unlike the traditional N-gram model the factored RBM uses distributed representations not only for context words but also for the words being predicted. This approach is generalized to deeper structures as reported in (Mnih and Hinton, 2008).

More recent work on neural network LM with deep architectures can be found in (Le et al., 2010, 2011; Mikolov et al., 2010; Mikolov et al., 2011; Mikolov, 2012). In particular, the work described in (Mikolov et al., 2011) and (Mikolov, 2012) makes use RNNs to build large scale language models. It achieves stability and fast convergence in training, helped by capping the growing gradient in training RNNs. It also develops adaptation schemes for the RNN-based LM by sorting the training data with respect to their relevance and by training the model during processing of the test data. Empirical comparisons with other state-of-the-art LM show much better performance of RNN especially in the perplexity measure. A separate work on applying RNN as an LM on the unit of characters instead of words can be found in (Sutskever et al., 2011). Very interesting properties such as predicting long-term dependency (e.g. making open and closing quotes in a paragraph) are demonstrated. But its usefulness in practical applications is not clear because word is such a powerful representation for natural language and changing word to character in LM limits most practical application scenarios.

Further, the use of hierarchical Bayesian priors in building up deep and recursive structure in LM appeared in (Huang and Renals, 2010). Specifically, Pitman-Yor process is exploited as the Bayesian prior, from which a deep (four layers) probabilistic generative model is built. It offers a principled approach to LM smoothing by incorporating the power-law distribution for natural language. As discussed in Chapter 3, this type of prior knowledge embedding is more readily achievable in the probabilistic modeling setup than in the neural network one.

# CHAPTER 9

## SELECTED APPLICATIONS IN NATURAL LANGUAGE PROCESSING

---

In the well-known and sometimes debatable work on natural language processing, Collobert and Weston (2008) developed and employed a convolutional DBN as the common model to simultaneously solve a number of classic problems including part-of-speech tagging, chunking, named entity tagging, semantic role identification, and similar word identification. More recent work reported in (Collobert, 2010) further developed a fast purely discriminative approach for parsing based on the deep recurrent convolutional architecture called Graph Transformer Network. Collobert et al., (2011) provides a comprehensive review on this line of work, specifically on ways of applying a unified neural network architectures and related deep learning algorithms to solve natural language processing problems from “scratch”. The theme of this line of work is to avoid task-specific, “man-made” feature engineering while providing versatility and unified features constructed automatically from deep learning applicable to all natural language processing tasks. The system described in (Collobert et al., 2011) automatically learns internal representation from vast amounts of mostly unlabeled training data.

One most important aspects of the work described in (Collobert and Weston, 2008) and (Collobert et al., 2011) is the transformation of raw word representations in terms of sparse vectors with a very high dimension (vocabulary size or its square or even its cubic) into low-dimensional, real-valued vectors for processing by subsequent neural network layers. This is known as “word embedding”, widely used in natural language processing and language modeling nowadays. Unsupervised learning is used where “context” of the word is used as the learning signal in neural networks. An excellent tutorial was recently given (Socher et al, 2012) to explain how the neural network is trained to perform word embedding originally proposed in (Collobert and Weston, 2008). More recent work proposes new ways of doing word embedding that better capture the semantics of words by incorporating both local and global document context and better account for homonymy and polysemy by learning multiple embeddings per word (Huang et al., 2012). Also, there is strong evidence that the use of the RNN can also provide empirically good

performance in word embedding (Mikolov, 2012). Further, very significant improvements in perplexity are reported using the RNN (Mikolov, 2012).

Another study on applying multi-task learning techniques with DBN is provided in (Deselaers et al., 2009) to attack a machine transliteration problem. This type of deep architectures and learning may be generalized to the more difficult machine translation problem.

In the work of (Sarikaya et al., 2011), DNNs (called DBNs in the paper) are used to perform a natural language call-routing task. The DNNs use unsupervised learning to discover multiple layers of features that are then used in a feed-forward neural network and fine-tuned to optimize discrimination. Unsupervised feature discovery is found to make DBNs far less prone to overfitting than the neural networks initialized with random weights. It also makes it easier to train neural networks with many hidden layers. DBNs are found to produce better classification results than several other widely used learning techniques, e.g., Maximum Entropy and Boosting based classifiers.

An interesting recent work on applying deep learning to natural language processing appears in (Socher et al., 2011), where a recursive neural network is used to build a deep architecture. The network is shown to be capable of successful merging of natural language words based on the learned semantic transformations of their original features. This deep learning approach provides an excellent performance on natural language parsing. The same approach is also demonstrated by the same authors to be successful in parsing natural scene images. In related studies, a similar recursive deep architecture is used for paraphrase detection (Socher et al., 2011a), and for predicting sentiment distributions from text (Socher et al., 2011b).



# CHAPTER 10

## SELECTED APPLICATIONS IN INFORMATION RETRIEVAL

---

Here we discuss very interesting applications of the DBN and the related deep auto-encoder to document indexing and information retrieval as published in (Salakhutdinov and Hinton, 2007; Hinton and Salakhutdinov, 2010). It is shown that the hidden variables in the final layer of a DBN not only are easy to infer but also give a much better representation of each document, based on the word-count features, than the widely used latent semantic analysis and the traditional TF-IDF approach for information retrieval. Using the compact code produced by deep networks, documents are mapped to memory addresses in such a way that semantically similar text documents are located at nearby address to facilitate rapid document retrieval. And the mapping from a word-count vector to its compact code is highly efficient, requiring only a matrix multiplication and a subsequent sigmoid function evaluation for each hidden layer in the encoder part of the network.

A deep generative model of DBN is exploited for the above purpose as discussed in (Hinton and Salakhutdinov, 2010). Briefly, the lowest layer of the DBN represents the word-count vector of a document and the top layer represents a learned binary code for that document. The top two layers of the DBN form an undirected associative memory and the remaining layers form a Bayesian (also called belief) network with directed, top-down connections. This DBN, composed of a set of stacked RBMs as we reviewed in Chapter 5, produces a feed-forward “encoder” network that converts word-count vectors to compact codes. By composing the RBMs in the opposite order, a “decoder” network is constructed that maps compact code vectors into reconstructed word-count vectors. Combining the encoder and decoder, one obtains a deep auto-encoder (subject to further fine-tuning as discussed in Chapter 4) for document coding and subsequent retrieval.

After the deep model is trained, the retrieval process starts with mapping each query document into a 128-bit binary code by performing a forward pass through the model with thresholding.

Then the Hamming distance between the query binary code and all other documents' 128-bit binary codes are computed efficiently.

The same idea above for coding text documents for information retrieval has been explored for audio document retrieval and speech feature coding problems with some initial exploration reported in (Deng et al., 2010) discussed in Chapter 4.

More recently, the deep stacking network (DSN) discussed in Chapter 6 has been explored for information retrieval with insightful results (Deng et al., 2013c). The experimental results suggest that the classification error rate using the binary decision of “relevant” vs. “non-relevant” from the DSN, which is closely correlated with the DSN training objective, is also generally correlated well with the NDCG as the common information retrieval quality measure. The exception is found in the region of high information retrieval quality.

# CHAPTER 11

## SELECTED APPLICATIONS IN IMAGE, VISION, & MULTIMODAL/MULTITASK PROCESSING

---

The original DBN and deep auto-encoder were developed and demonstrated with success on the simple image recognition and dimensionality reduction (coding) tasks (MNIST) in (Hinton and Salakhutdinov, 2006). It is interesting to note that the gain of coding efficiency using the DBN-based auto-encoder on the image data over the conventional method of principal component analysis as demonstrated in (Hinton and Salakhutdinov, 2006) is very similar to the gain reported in (Deng et al., 2010) on the speech data over the traditional technique of vector quantization.

In (Nair and Hinton, 2009), a modified DBN is developed where the top-layer model uses a third-order Boltzmann machine. This type of DBN is applied to the NORB database – a 3-dimensional object recognition task. An error rate close to the best published result on this task is reported. In particular, it is shown that the DBN substantially outperforms shallow models such as SVMs.

In (Tang and Elasmith, 2010), two strategies to improve the robustness of the DBN are developed. First, sparse connections in the first layer of the DBN are used as a way to regularize the model. Second, a probabilistic de-noising algorithm is developed. Both techniques are shown to be effective in improving robustness against occlusion and random noise in a noisy image recognition task.

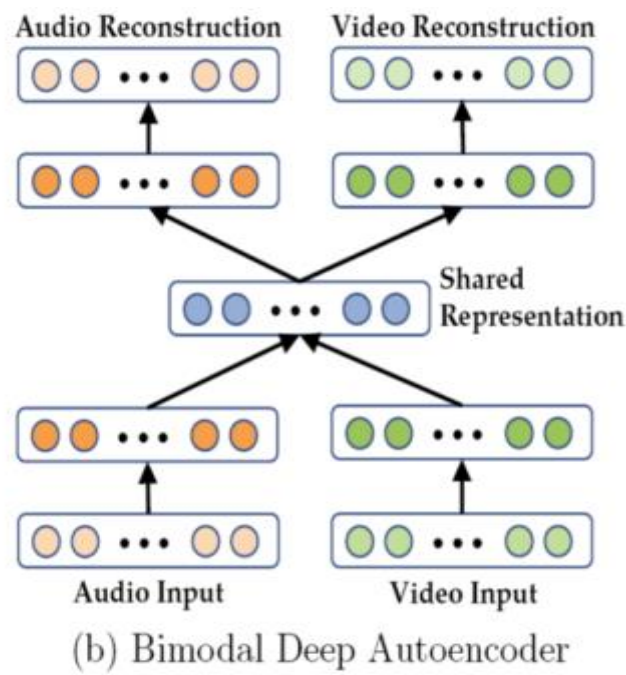
DBNs have also been successfully applied to create compact but meaningful representations of images (Taralba et al., 2008) for retrieval purposes. On this large collection image retrieval task, deep learning approaches also produced strong results.

Deep architectures with convolution structure have been found highly effective and been commonly used in computer vision and image recognition (Bengio and LeCun, 1995; LeCun et al., 1998; Jarrett et al., 2009; Kavukcuoglu et al., 2010; Ciresan et al., 2012; Le et al., 2012; Dean et al., 2012; Krizhevsky et al., 2012). The most notable advance was recently achieved in the 2012 ImageNet LSVRC contest, where 1000 different image classes are the targets with 1.2 million high-resolution images in the training set. On the test set consisting of 150k images, the deep CNN approach described in (Krizhevsky et al., 2012) achieved the error rates considerably lower than the previous state-of-the-art. Very large deep CNNs are used, consisting of 60 million weights, and 650,000 neurons, and five convolutional layers together with max-pooling layers. Additional three fully-connected layers as in the DNN described previously are used on top of the deep CNN layers. Although all the above structures were developed separately in earlier work, their best combination accounted for part of the success. Additional factors contributing to the final success are: 1) a powerful regularization technique called “dropout” (see details in Hinton et al., 2012); and 2) use of non-saturating neurons or rectified linear units (ReLU) that compute  $y = \max(x, 0)$ , significantly speeding up the training process especially with a very efficient GPU implementation.

The use of a temporally conditional DBN for video sequence and human motion synthesis is reported in (Taylor et al., 2007). The conditional DBN makes the DBN weights associated with a fixed time window conditioned on the data from previous time steps. The computational tool offered in this type of temporal DBN and the related recurrent networks may provide the opportunity to improve the DBN-HMMs towards efficient integration of temporal-centric human speech production mechanisms into DBN-based speech production model.

A very interesting study appeared in (Ngiam et al., 2011), where the authors propose and evaluate a novel application of deep networks to learn features over both audio and video modalities. A similar deep auto-encoder architecture that we described in Chapter 4 and in (Deng et al, 2010) is used and it can be considered as a generalization from a single modality to two modalities, as shown in Figure 11.1. Cross modality feature learning has been demonstrated – better features for video can be learned if both audio and video information sources are available at feature learning time. The authors further show how to learn a shared audio and video representation, and evaluate it on a fixed task, where the classifier is trained with audio-only data but tested with video-only data and vice versa. The work concludes that deep learning architectures are generally effective in learning multimodal features from unlabeled data and in improving single modality features through cross modality learning. One exception is the cross-modality setting using the CUAVE dataset. The results presented in (Ngiam et al., 2011) show that there is an improvement by learning

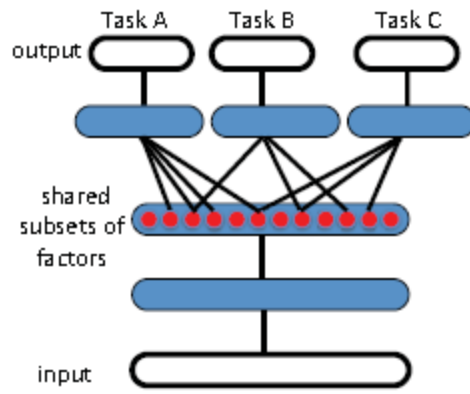
video features with both video and audio compared to learning features with only video data. However, the same paper also shows that a model of (Papandreou, 2009) in which a sophisticated signal processing technique for extracting visual features, together with the uncertainty-compensation method developed originally from robust speech recognition (Deng et al., 2005), gives the best classification accuracy in the cross-modeling learning task, beating the features derived from the generative deep architecture designed for this task.



**Figure 11.1:** The architecture of a deep autoencoder for multi-modal audio-visual features (Ngiam et al., 2011).

While the deep generative architecture for multimodal learning described in (Ngiam et al., 2011) is based on non-probabilistic auto-encoder neural nets, a probabilistic version based on deep Boltzmann machine (DBM) has appeared more recently for the same multimodal application. In (Srivastava and Salakhutdinov, 2012), a DBM is used to extract a unified representation integrating separate modalities, useful for both classification and information retrieval tasks. Rather than using the “bottleneck” layers in the deep auto-encoder to represent multimodal inputs, here a probability density is defined on the joint space of multimodal inputs, and states of suitably defined latent variables are used for the representation. The advantage of this probabilistic formulation, lacking in the deep auto-encoder, is that the missing modality’s information can be filled in naturally by sampling from its conditional distribution. For the bi-modal data consisting of image and text, the multimodal DBM is shown to outperform deep multimodal auto-encoder as well as multimodal DBN in classification and information retrieval tasks.

The DNN architecture discussed above for multimodal processing and learning can be regarded as a special case of more general multitask learning and even more general transfer learning (Caruana, 1997; Bengio et al., 2013). Transfer learning, encompassing both adaptive and multitask learning (Deng and Li, 2013), refers to the ability of a learning architecture and technique to exploit common hidden explanatory factors among different learning tasks. Such exploitation permits sharing of aspects of diverse types of input data sets, thus allowing the possibility of transferring knowledge across seemingly different learning tasks. As argued in (Bengio et al., 2013), the learning architecture shown in Figure 11.2 and the associated learning algorithms have an advantage for such tasks because they learn representations that capture underlying factors, a subset of which may be relevant for each particular task. This advantage has been recently demonstrated by a number of experiments which we briefly review below; see other somewhat older experiments in Bengio et al., 2013).

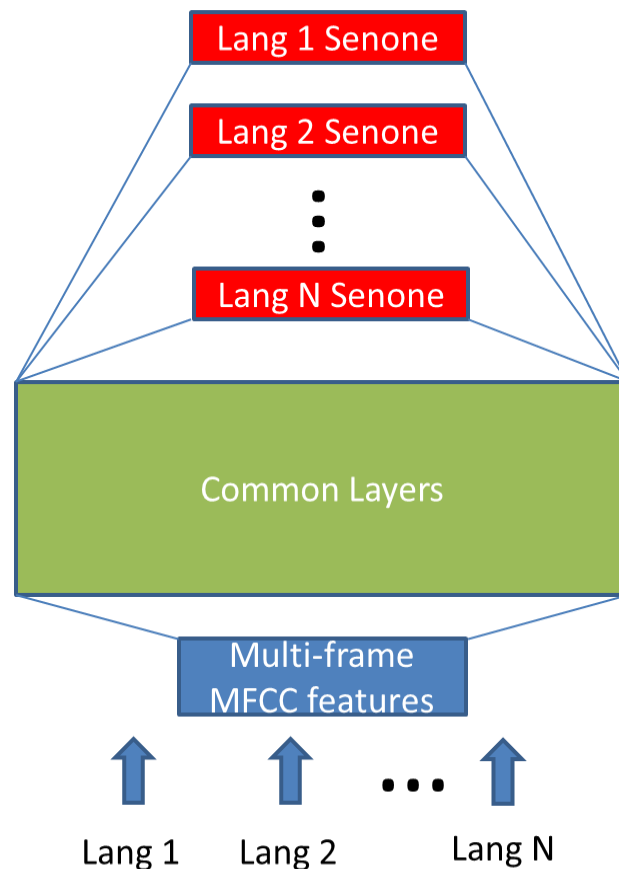


**Figure 11.2:** A DNN architecture for multitask learning that is aimed to discover hidden explanatory factors sharing among three tasks A, B, and C (adopted from Bengio et al., 2013).

In the very recent papers of (Huang et al., 2013; Deng et al., 2013a) and (Heigold et al., 2013), two research groups independently developed closely related DNN architectures with multitask learning capabilities for multilingual speech recognition. See Figure 11.3 for an illustration of this type of architecture. The idea behind these architectures is that the hidden layers in the DNN, when learned appropriately, serve as increasingly complex feature transformations sharing common hidden factors across the acoustic data in different languages. Then, the final softmax layer represented by a log-linear classifier makes use of the most abstract feature vectors represented in the top-most the hidden layer. While the log-linear classifier is necessarily separate for different languages, the feature transformations can be shared across languages. Excellent multilingual speech recognition results are reported, far exceeding the earlier results using the GMM-HMM based approaches (e.g., Lin et al., 2009; Yu et al., 2009).

The implication of this set of work is significant and far reaching. It points to the possibility of quickly building a high-performance DNN-based system for a new language from an existing multilingual DNN. This huge benefit would require only a small amount of training data from the target language, although having more data would further improve the performance. This

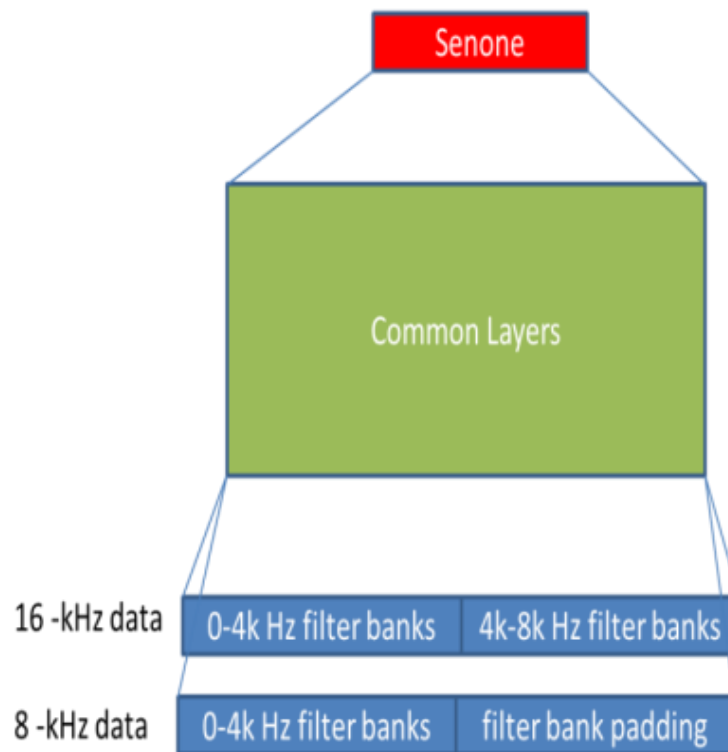
multitasking learning approach can completely eliminate the unsupervised pre-training stage, and can train the DNN with much fewer epochs. Extension of this set of work would be to efficiently build a language-universal speech recognition system. Such a system can not only recognize many languages and improve the accuracy for each individual language, but also expand the languages supported by simply stacking softmax layers in the DNN for new languages.



**Figure 11.3:** A DNN architecture for multilingual speech recognition (Huang et al., 2013; Deng et al., 2013a)



A closely related DNN architecture, as shown in Figure 11.4), with multitask learning capabilities was also recently applied to another acoustic modeling problem --- learning joint representations for two separate sets of acoustic data (Li et al., 2012; Deng et al., 2013a). The set that consists of the speech data with 16k sampling rate is of wideband and high quality, which is often collected from increasingly popular smart phones under the voice search scenario. Another, narrowband data set has a lower sampling rate of 8k, often collected using the telephony speech recognition systems.



**Figure 11.4:** DNN training and testing with mixed-band acoustic data with 16-kHz and 8-kHz sampling rates (Li et al., 2012).

# CHAPTER 12

## EPILOGUES

---

This monograph presented a brief history of deep learning and developed a categorization scheme to analyze the existing deep architectures in the literature into generative, discriminative, and hybrid classes. The deep auto-encoder, DSN, and DBN-DNN architectures, one in each of the three classes, are discussed and analyzed in detail, as they appear to be popular and promising approaches with author's personal research experiences. Applications of deep learning in five broad areas of information processing are also reviewed.

The literature on deep learning is vast, mostly coming from the machine learning community. The signal processing community embraced deep learning only within the past four years or so and the momentum is growing fast. This monograph is written mainly from the signal processing perspective. Beyond just surveying existing deep learning work, a classificatory scheme based on the architecture and the nature of learning algorithms is developed and in-depth analysis with concrete examples conducted. This will hopefully provide insight for readers to better understand the capability of the various deep learning systems discussed in the book, the connection among different but similar deep learning methods, and how to design proper deep learning algorithms under different circumstances.

Throughout this review, the important message is conveyed that building and learning deep hierarchies of features are highly desirable. We have discussed the difficulty of learning parameters in all layers in one shot due to pervasive local optimum and gradient dilution. The generative, pre-training method in the hybrid architecture of DBN-DNN, which we reviewed in detail in Chapter 5, appears to have offered a useful, albeit empirical, solution to poor local optima in optimization and to regularization for the deep model containing massive parameters even though a solid theoretical foundation is still lacking.

Deep learning is an emerging technology. Despite the empirical promising results reported so far, much needs to be developed. Importantly, it has not been the experience of deep learning researchers that a single deep learning technique can be successful for all classification tasks. For

example, while the popular learning strategy of generative pre-training followed by discriminative fine-tuning seems to work well empirically for many tasks, it failed to work for some other tasks that have been explored (e.g., language identification or speaker recognition; unpublished). For these tasks, the features extracted at the generative pre-training phase seem to describe the underlining speech variations well but do not contain sufficient information to distinguish between different languages. A learning strategy that can extract discriminative yet also invariant features is expected to provide better solutions. Extracting discriminative features may also greatly reduce the model size needed in the many current deep learning systems. Also, domain knowledge such as what kind of invariance is useful for a specific task in hand (e.g., vision, speech, or language) and what kind of regularization in terms of parameter constraints is key to the success of applying deep learning methods. Further, new types of deep neural network architectures and learning beyond the several popular ones discussed in this book are currently under active development (Deng et al., 2013b), holding the promise to improve the performance of deep learning models in the applications in signal and information processing.

Recent published work shows that there is vast room to improve the current optimization techniques for learning deep architectures (Martens, 2010; Le et al., 2011; Martens and Sutskever, 2011; Dean et al., 2012; Sutskever, 2013). To what extent pre-training is essential to learning the full set of parameters in deep architectures has been currently under investigation, especially when very large amounts of labeled training data are available which reduces or even obliterates the need for model regularization. Some preliminary results have been discussed in this book and in (Yu et al. 2010; Seide et al. 2011; Hinton et al., 2012).

In recent years, machine learning is becoming increasingly dependent on large-scale data sets. For instance, many of the recent successes of deep learning as discussed earlier in this book have relied on access to massive data sets and massive computing power. It would become increasingly difficult to explore the new algorithmic space without the access to large, real-world data sets and without the related engineering expertise. How well deep learning algorithms behave would be depend on the amount of data and computing power available. As we showed with speech recognition examples, a deep learning algorithm that appears to be performing not so great on small data sets can begin to perform considerably better when these limitations are removed, one of main reasons for the recent resurgence in neural network research. As an example, the DBN pretraining that ignited a new era of (deep) machine learning research appears unnecessary if enough data and computing power are used.

As a consequence, effective and scalable parallel algorithms are critical for training deep models with large data sets, as in many common information processing applications such as speech recognition and machine translation. The popular mini-batch stochastic gradient technique is known to be difficult to be parallelized over computers. The common practice nowadays is to use GPGPUs to speed up the learning process, although recent advance in developing asynchronous stochastic gradient learning has shown promises by using large-scale CPU clusters (e.g. Le et al., 2011; Dean et al., 2012). In this interesting computing architecture, many different replicas of the DNN to compute gradients on different subsets of the training data in parallel. These gradients are communicated to a central parameter server that updates the shared weights. Even though each replica typically computes gradients using parameter values not immediately updated, stochastic gradient descent is robust to the slight errors this has introduced. To make deep learning techniques scalable to very large training data, theoretically sound parallel learning algorithms together with novel architectures need to be further developed (e.g., Bottou and LeCun, 2004; Dean et al., 2012; Hutchinson et al., 2013; Sutskever, 2013; Bengio et al., 2013).

One major barrier to the application of DNNs and related deep models is that it currently requires considerable skill and experience to choose sensible values for hyper-parameters such as the learning rate schedule, the strength of the regularizer, the number of layers and the number of units per layer, etc. Sensible values for one hyper-parameter may depend on the values chosen for other hyper-parameters and hyper-parameter tuning in DNNs is especially expensive. Some interesting methods for solving the problem have been developed recently, including random sampling (Bergstra et al., 2012) and Bayesian optimization procedure (Snoek et al., 2012). Further research is needed in this important area.

Finally, solid theoretical foundations of deep learning need to be established in a myriad of aspects. As an example, the success of deep learning in unsupervised learning has not been demonstrated as much as for supervised learning; yet the essence and major motivation of deep learning lie right in unsupervised learning for automatically discovering data representation. The issues involve appropriate objectives for learning effective feature representations and the right deep learning architectures/algorithms for distributed representations to effectively disentangle the hidden explanatory factors of variation in the data. For example, most of the successful deep learning techniques have so far dealt with unstructured or “flat” classification problems. Although speech recognition is a sequential classification problem, a separate HMM is used to handle the sequence structure and the DNN is only used to produce the frame-level, unstructured posterior distributions. Recent proposals have called for moving beyond the “flat” representations and incorporating

structures in both the deep learning architectures and input and output representations (Socher, 2012). Also, deep learning researchers have been advised by neuroscientists to seriously consider a broader set of issues and learning architectures so as to gain insight into biologically plausible representations in the brain that may be useful for practical applications (Olshausen, 2012). How can computational neuroscience models about hierarchical brain structure and learning style help improve engineering deep learning architectures and algorithms? All the issues discussed here will need intensive research in order to further push the frontier of deep learning.

# BIBLIOGRAPHY

---

- Abdel-Hamid, O., Mohamed, A., Jiang, H., and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," Proc. ICASSP, 2012.
- Arel, I., Rose, C., and Karnowski, T. "Deep Machine Learning - A New Frontier in Artificial Intelligence," IEEE Computational Intelligence Mag., Nov., 2010.
- Baker, J., Deng, L., Glass, J., Khudanpur, S., Lee, C.-H., Morgan, N., and O'Shaughnessy, D. "Research developments and directions in speech recognition and understanding," IEEE Sig. Proc. Mag., vol. 26, no. 3, May 2009, pp. 75-80.
- Baker, J., Deng, L., Glass, J., Khudanpur, S., Lee, C.-H., Morgan, N., and O'Shaughnessy, D. "Updated MINS report on speech recognition and understanding," IEEE Sig. Proc. Mag., vol. 26, no. 4, July 2009a.
- Bengio, Y., Boulanger, N., and Pascanu. R. "Advances in optimizing recurrent networks," Proc. ICASSP, 2013.
- Bengio, Y., Courville, A., and Vincent, P. "Representation learning: A review and new perspectives," IEEE Trans. PAMI, 2013a.
- Bengio, Y. "Learning deep architectures for AI," in Foundations and Trends in Machine Learning, Vol. 2, No. 1, 2009, pp. 1-127.
- Bengio, Y., Ducharme, R., Vincent, P. and Jauvin, C. "A neural probabilistic language model," Proc. NIPS, 2000, pp. 933-938.
- Bengio, Y., De Mori, R., Flammia, G. and Kompe, F. "Global optimization of a neural network—Hidden Markov model hybrid," in Proc. Eurospeech, 1991.
- Bergstra, J. and Bengio, Y. "Random search for hyper-parameter optimization," J. Machine Learning Research," Vol. 3, pp. 281-305, 2012.
- Bottou, L. and LeCun. Y. "Large scale online learning," Proc. NIPS, 2004.
- Bilmes, J. "Dynamic graphical models," IEEE Signal Processing Mag., vol. 33, pp. 29–42, 2010.
- Bilmes, J. and Bartels, C. "Graphical model architectures for speech recognition," IEEE Signal Processing Mag., vol. 22, pp. 89–100, 2005.
- Bouclard, H. and Morgan, N., Connectionist Speech Recognition: A Hybrid Approach, Norwell, MA: Kluwer, 1993.
- Bouvrie, J. "Hierarchical Learning: Theory with Applications in Speech and Vision," Ph.D. thesis, MIT, 2009.

- Bridle, J., L. Deng, J. Picone, H. Richards, J. Ma, T. Kamm, M. Schuster, S. Pike, and R. Reagan, "An investigation of segmental hidden dynamic models of speech coarticulation for automatic speech recognition," Final Report for 1998 Workshop on Language Engineering, CLSP, Johns Hopkins, 1998.
- Caruana, R. "Multitask Learning," *Machine Learning*, Vol. 28, pp. 41-75, Kluwer Academic Publishers, 1997.
- Cho, Y. and Saul L. "Kernel methods for deep learning," Proc. NIPS, pp. 342–350, 2009.
- Ciresan, D., Giusti, A., Gambardella, L., and Schmidhuber, J. "Deep neural networks segment neuronal membranes in electron microscopy images," Proc. NIPS, 2012.
- Cohen, W. and R. V. de Carvalho. "Stacked sequential learning," Proc. IJCAI, pp. 671–676, 2005.
- Collobert, R. "Deep learning for efficient discriminative parsing," Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2010.
- Collobert, R. and Weston J. "A unified architecture for natural language processing: Deep neural networks with multitask learning," Proc. ICML, 2008.
- Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. "Natural language processing (almost) from scratch," J. Machine Learning Research, Vo. 12, pp. 2493-2537, 2011.
- Dahl, G., Yu, D., Deng, L., and Acero, A. "Context-dependent DBN-HMMs in large vocabulary continuous speech recognition," Proc. ICASSP, 2011.
- Dahl, G., Yu, D., Deng, L., and Acero, A. "Context-dependent, pre-trained deep neural networks for large vocabulary speech recognition," IEEE Trans. Audio, Speech, & Language Proc., Vol. 20 (1), pp. 30-42, January 2012.
- Dahl, G., Ranzato, M., Mohamed, A. and Hinton, G. "Phone recognition with the mean-covariance restricted Boltzmann machine," Proc. NIPS, vol. 23, 2010, 469-477.
- Dean, J., Corrado, G., R. Monga, K. Chen, M. Devin, Q. Le, M. Mao, M. Ranzato, A. Senior, P. Tucker, Yang, K., and Ng, A. "Large Scale Distributed Deep Networks," Proc. NIPS, 2012.
- Deng, L. and Li, X. "Machine learning paradigms in speech recognition: An overview," IEEE Trans. Audio, Speech, & Language, May 2013.
- Deng, L., Abdel-Hamid, O., and Yu, D. "A deep convolutional neural network using heterogeneous pooling for trading acoustic invariance with phonetic confusion," Proc. ICASSP, 2013.
- Deng, L., Li, J., Huang, K., Yao, D. Yu, F. Seide, M. Seltzer, G. Zweig, X. He, J. Williams, Y. Gong, and A. Acero. "Recent advances in deep learning for speech research at Microsoft," Proc. ICASSP, 2013a.
- Deng, L., Hinton, G., and Kingsbury, B. "New types of deep neural network leaning for speech recognition and related applications: An overview," Proc. ICASSP, 2013b.

- Deng, L., He, X., and J. Gao, J. “Deep stacking networks for information retrieval,” Proc. ICASSP, 2013c.
- Deng, L., Tur, G, He, X, and Hakkani-Tur, D. “Use of kernel deep convex networks and end-to-end learning for spoken language understanding,” Proc. IEEE Workshop on Spoken Language Technologies, December 2012.
- Deng, L., Yu, D., and Platt, J. “Scalable stacking and learning for building deep architectures,” Proc. ICASSP, 2012a.
- Deng, L., Hutchinson, B., and Yu, D. “Parallel training of deep stacking networks,” Proc. Interspeech, 2012b.
- Deng, L. “An Overview of Deep-Structured Learning for Information Processing,” Proceedings of Asian-Pacific Signal & Information Processing Annual Summit and Conference (APSIPA-ASC), October 2011.
- Deng, L. and Yu, D. “Deep Convex Network: A scalable architecture for speech pattern classification,” Proc. Interspeech, 2011.
- Deng, L., Seltzer, M., Yu, D., Acero, A., Mohamed, A., and Hinton, G. “Binary coding of speech spectrograms using a deep auto-encoder,” Proc. Interspeech, 2010.
- Deng, L., Yu, D., and Hinton, G. “Deep Learning for Speech Recognition and Related Applications” NIPS Workshop, 2009.
- Deng, L. and Yu, D. “Use of differential cepstra as acoustic features in hidden trajectory modeling for phonetic recognition,” Proc. ICASSP, 2007.
- Deng, L. DYNAMIC SPEECH MODELS – Theory, Algorithm, and Application, Morgan & Claypool, December 2006.
- Deng, L., Yu, D. and Acero, A. “Structured speech modeling,” IEEE Trans. on Audio, Speech and Language Processing, vol. 14, no. 5, pp. 1492-1504, September 2006
- Deng, L., Yu, D. and Acero, A. “A bidirectional target filtering model of speech coarticulation: Two-stage implementation for phonetic recognition,” IEEE Transactions on Audio and Speech Processing, vol. 14, no. 1, pp. 256-265, January 2006a.
- Deng, L., Wu, J., Droppo, J., and Acero, A. “Dynamic Compensation of HMM Variances Using the Feature Enhancement Uncertainty Computed From a Parametric Model of Speech Distortion,” IEEE Transactions on Speech and Audio Processing, vol. 13, no. 3, pp. 412–421, 2005.
- Deng, L. and O’Shaughnessy, D. SPEECH PROCESSING – A Dynamic and Optimization-Oriented Approach, Marcel Dekker, 2003.
- Deng, L. “Switching dynamic system models for speech articulation and acoustics,” in Mathematical Foundations of Speech and Language Processing, pp. 115–134. Springer-Verlag, New York, 2003.



- Deng, L. "Computational Models for Speech Production," in Computational Models of Speech Pattern Processing, pp. 199-213, Springer Verlag, 1999.
- Deng, L., Ramsay, G., and Sun, D. "Production models as a structural basis for automatic speech recognition," Speech Communication, vol. 33, no. 2-3, pp. 93-111, Aug 1997.
- Deng, L. and Sameti, H. "Transitional speech units and their representation by regressive Markov states: Applications to speech recognition," IEEE Transactions on speech and audio processing, vol. 4, no. 4, pp. 301-306, July 1996.
- Deng, L., Aksmanovic, M., Sun, D., and Wu, J. "Speech recognition using hidden Markov models with polynomial regression functions as nonstationary states," IEEE Transactions on Speech and Audio Processing, vol. 2, no. 4, pp. 507-520, 1994.
- Deng L. and Sun, D. "A statistical approach to automatic speech recognition using the atomic speech units constructed from overlapping articulatory features," Journal of the Acoustical Society of America, vol. 85, no. 5, pp. 2702-2719, 1994.
- Deng, L. "A stochastic model of speech incorporating hierarchical nonstationarity," IEEE Transactions on Speech and Audio Processing, vol. 1, no. 4, pp. 471-475, 1993.
- Deng, L. "A generalized hidden Markov model with state-conditioned trend functions of time for the speech signal," Signal Processing, vol. 27, no. 1, pp. 65-78, 1992.
- Deselaers, T., Hasan, S., Bender, O. and Ney, H. "A deep learning approach to machine transliteration," Proc. 4th Workshop on Statistical Machine Translation , pp. 233-241, Athens, Greece, March 2009.
- Erhan, D., Bengio, Y., Courville, A., Manzagol, P., Vencent, P., and Bengio, S. "Why does unsupervised pre-training help deep learning?" J. Machine Learning Research, pp. 201-208, 2010.
- Fine, S., Singer, Y. and Tishby, N. "The hierarchical hidden Markov model: Analysis and applications," Machine Learning, vol. 32, p. 41-62, 1998.
- Gens, R. and Domingo, P. "Discriminative learning of sum-product networks," NIPS, 2012.
- George, D. "How the Brain Might Work: A Hierarchical and Temporal Model for Learning and Recognition," Ph.D. thesis, Stanford University, 2008.
- Gibson, M. and Hain, T. "Error approximation and minimum phone error acoustic model estimation," IEEE Trans. Audio, Speech, and Language Proc., vol. 18, no. 6, August 2010, pp. 1269-1279.
- Glorot, X., Bordes, A., and Bengio, Y. "Deep sparse rectifier neural networks," Proc. AISTAT, April 2011.
- Glorot, X. and Bengio, Y. "Understanding the difficulty of training deep feed-forward neural networks" Proc. AISTAT, 2010.

- Graves, A., Fernandez, S., Gomez, F., and Schmidhuber, J. "Connectionist temporal classification: Labeling unsegmented sequence data with recurrent neural networks," Proc. ICML, 2006.
- Graves, A. "Sequence Transduction with Recurrent Neural Networks," Representation Learning Worksp, ICML 2012.
- Graves, A., Mahamed, A., and Hinton, G. "Speech recognition with deep recurrent neural networks," Proc. ICASSP, 2013.
- Hawkins, J. and Blakeslee, S. On Intelligence: How a New Understanding of the Brain will lead to the Creation of Truly Intelligent Machines, Times Books, New York, 2004.
- Hawkins, G., Ahmad, S. and Dubinsky, D. "Hierarchical Temporal Memory Including HTM Cortical Learning Algorithms," Numenta Tech. Report, December 10, 2010.
- He, X., Deng, L., Chou, W. "Discriminative learning in sequential pattern recognition – A unifying review for optimization-oriented speech recognition," IEEE Sig. Proc. Mag., vol. 25, 2008, pp. 14-36.
- He, X. and Deng, L. "Speech recognition, machine translation, and speech translation – A unifying discriminative framework," IEEE Sig. Proc. Magazine, Vol. 28, November, 2011.
- He, X. and Deng, L. "Optimization in speech-centric information processing: Criteria and techniques," Proc. ICASSP, 2012.
- He, X. and Deng, L. "Speech-centric information processing: An optimization-oriented approach," Proc. of the IEEE, 2013.
- Heigold, G., Vanhoucke, V., Senior, A. Nguyen, P., Ranzato, M., Devin, M., and Dean, J. "Multilingual acoustic models using distributed deep neural networks," Proc. ICASSP, 2013.
- Heigold, G., Ney, H., Lehn, P., Gass, T., Schluter, R. "Equivalence of generative and log-linear models," IEEE Trans. Audio, Speech, and Language Proc., vol. 19, no. 5, February 2011, pp. 1138-1148.
- Heintz, I., Fosler-Lussier, E., and Brew, C. "Discriminative input stream combination for conditional random field phone recognition," IEEE Trans. Audio, Speech, and Language Proc., vol. 17, no. 8, Nov. 2009, pp. 1533-1546.
- Hifny, Y. and Renals, S. "Speech recognition using augmented conditional random fields," IEEE Trans. Audio, Speech, and Language Proc., vol. 17, no. 2, February 2009, pp. 354-365.
- Hinton, G. and Salakhutdinov, R. "Discovering binary codes for documents by learning deep generative models," Topics in Cognitive Science, pp. 1-18, 2010.
- Hinton, G., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. "Improving neural networks by preventing co-adaptation of feature detectors," arXiv: 1207.0580v1, 2012.
- Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T., and Kingsbury, B., "Deep Neural Networks for Acoustic Modeling in

- Speech Recognition,” IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 82-97, November 2012.
- Hinton, G., Krizhevsky, A., and Wang, S. “Transforming auto-encoders,” Proc. Intern. Conf. Artificial Neural Networks, 2011.
- Hinton, G. “A practical guide to training restricted Boltzmann machines,” UTML Tech Report 2010-003, Univ. Toronto, August 2010.
- Hinton, G., Osindero, S., and Teh, Y. “A fast learning algorithm for deep belief nets,” Neural Computation, vol. 18, pp. 1527-1554, 2006.
- Hinton, G. and Salakhutdinov, R. “Reducing the dimensionality of data with neural networks,” Science, vol. 313. no. 5786, pp. 504 - 507, July 2006.
- Hinton, G. “A better way to learn features,” Communications of the ACM,” Vol. 54, No. 10, October, 2011, pp. 94.
- Huang, J., Li, J., Deng, L., and Yu, D. “Cross-language knowledge transfer using multilingual deep neural networks with shared hidden layers,” Proc. ICASSP, 2013.
- Huang, S. and Renals, S. “Hierarchical Bayesian language models for conversational speech recognition,” IEEE Trans. Audio, Speech, and Language Proc., vol. 18, no. 8, November 2010, pp. 1941-1954.
- Huang, E., Socher, R., Manning, C, and Ng, A. “Improving Word Representations via Global Context and Multiple Word Prototypes,” Proc. ACL, 2012.
- Hutchinson, B., Deng, L., and Yu, D. “A deep architecture with bilinear modeling of hidden representations: Applications to phonetic recognition,” Proc. ICASSP, 2012.
- Hutchinson, B., Deng, L., and Yu, D. “Tensor deep stacking networks,” IEEE Trans. Pattern Analysis and Machine Intelligence, 2013.
- Jaitly, N. and Hinton, G. “Learning a better representation of speech sound waves using restricted Boltzmann machines,” Proc. ICASSP, 2011.
- Jaitly, N., Nguyen, P., and Vanhoucke, V. “Application of pre-trained deep neural networks to large vocabulary speech recognition,” Proc. Interspeech, 2012.
- Jarrett, K., Kavukcuoglu, K. and LeCun, Y. “What is the best multistage architecture for object recognition?” Proc. Intl. Conf. Computer Vision, pp. 2146–2153, 2009.
- Jiang, H. and Li, X. “Parameter estimation of statistical models using convex optimization: An advanced method of discriminative training for speech and language processing,” IEEE Signal Processing Magazine, vol. 27, no. 3, pp. 115–127, 2010.
- Juang, B.-H., Chou, W., and Lee, C.-H. “Minimum classification error rate methods for speech recognition,” IEEE Trans. On Speech and Audio Processing, vol. 5, pp. 257–265, 1997.
- Kavukcuoglu, K., Sermanet, P., Boureau, Y., Gregor, K., Mathieu M., and LeCun, Y. “Learning Convolutional Feature Hierarchies for Visual Recognition,” Proc. NIPS, 2010.

- Ketabdar, H. and Bourlard, H. "Enhanced phone posteriors for improving speech recognition systems," *IEEE Trans. Audio, Speech, and Language Proc.*, vol. 18, no. 6, August 2010, pp. 1094-1106.
- Kingsbury, B. "Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling," *Proc. ICASSP*, 2009.
- Kingsbury, B., Sainath, T., and Soltau, H. "Scalable minimum Bayes risk training of deep neural network acoustic models using distributed Hessian-free optimization," *Proc. Interspeech*, 2012.
- Krizhevsky, A., Sutskever, I. and Hinton, G. "ImageNet classification with deep convolutional neural Networks," *Proc. NIPS* 2012.
- Kubo, Y., Hori, T., and Nakamura, A. "Integrating deep neural networks into structural classification approach based on weighted finite-state transducers," *Proc. Interspeech*, 2012.
- Kurzweil R. *How to Create a Mind*. Viking Books, Dec., 2012.
- Lang, K., Waibel, A., and Hinton, G. "A time-delay neural network architecture for isolated word recognition," *Neural Networks*, Vol. 3(1), pp. 23-43, 1990.
- Larochelle, H. and Bengio, Y. "Classification using discriminative restricted Boltzmann machines," *Proc. ICML*, 2008.
- Le, H., Allauzen, A., Wisniewski, G., and Yvon, F. "Training continuous space language models: Some practical issues," in *Proc. of EMNLP*, 2010, pp. 778-788.
- Le, H., Oparin, I., Allauzen, A., Gauvain, J., and Yvon, F. "Structured output layer neural network language model," *Proc. ICASSP*, 2011.
- Le, Q., Ngiam, J., Coates, A., Lahiri, A., Prochnow, B., and Ng, A. "On optimization methods for deep learning," *Proc. ICML*, 2011.
- Le, Q., Ranzato, M., Monga, R., Devin, M., Corrado, G., Chen, K., Dean, J., Ng, A. "Building High-Level Features Using Large Scale Unsupervised Learning," *Proc. ICML* 2012.
- LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, Vol. 86, pp. 2278-2324, 1998.
- LeCun, Y. and Bengio, Y. "Convolutional networks for images, speech, and time series," in *The Handbook of Brain Theory and Neural Networks* (M. Arbib, ed.), pp. 255- 258, Cambridge, Massachusetts: MIT Press, 1995.
- LeCun, Y., Chopra S., Ranzato, M., and Huang, F. "Energy-based models in document recognition and computer vision," *Proc. Intern. Conf. Document Analysis and Recognition (ICDAR)*, 2007.
- Lee, C.-H. "From knowledge-ignorant to knowledge-rich modeling: A new speech research paradigm for next-generation automatic speech recognition," *Proc. ICSLP*, 2004, p. 109-111.

- Lee, H., Grosse, R., Ranganath, R., and Ng, A. "Unsupervised learning of hierarchical representations with convolutional deep belief networks," *Communications of the ACM*, Vol. 54, No. 10, October, 2011, pp. 95-103.
- Lee, H., Grosse, R., Ranganath, R., and Ng, A. "Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations," *Proc. ICML*, 2009.
- Lee, H., Largman, Y., Pham, P., Ng, A. "Unsupervised feature learning for audio classification using convolutional deep belief networks," *Proc. NIPS*, 2010.
- Lena, P., Nagata, K., and Baldi, P. "Deep spatiotemporal architectures and learning for protein structure prediction," *Proc. NIPS*, 2012.
- Li, J., Yu, D., Huang, J., and Gong, Y. "Improving wideband speech recognition using mixed-bandwidth training data in CD-DNN-HMM," *Proc. IEEE SLT* 2012.
- Lin, H., Deng, L., Yu, D., Gong, Y., Acero, A., and C-H Lee, "A study on multilingual acoustic modeling for large vocabulary ASR," *Proc. ICASSP*, 2009.
- Ling, Z., Richmond, K., and Yamagishi, J. "Articulatory control of HMM-based parametric speech synthesis using feature-space-switched multiple regression," *IEEE Trans. Audio, Speech, and Language Proc.*, Vol. 21, Jan, 2013.
- Markoff, J. "Scientists See Promise in Deep-Learning Programs," *New York Times*, Nov 24, 2012.
- Martens, J. "Deep learning with Hessian-free optimization," *Proc. ICML*, 2010.
- Martens, J. and Sutskever, I. "Learning recurrent neural networks with Hessian-free optimization," *Proc. ICML*, 2011.
- Mikolov, T. "Statistical Language Models based on Neural Networks," PhD thesis, Brno University of Technology, 2012.
- Mikolov, T., Deoras, A., Povey, D., Burget, L., and Cernocky, J. "Strategies for training large scale neural network language models," *Proc. IEEE ASRU*, 2011.
- Mikolov, T., Karafiat, M., Burget, L., Cernocky, J., and Khudanpur, S. "Recurrent neural network based language model," *Proc. ICASSP*, 2010, 1045–1048.
- Minami, Y., McDermott, E. Nakamura, A. and Katagiri, S. "A recognition method with parametric trajectory synthesized using direct relations between static and dynamic feature vector time series," *Proc. ICASSP*, pp. 957-960, 2002.
- Mnih, A. and Hinton G. "Three new graphical models for statistical language modeling," *Proc. ICML*, 2007, pp. 641-648.
- Mnih, A. and Hinton G. "A scalable hierarchical distributed language model" *Proc. NIPS*, 2008, pp. 1081-1088.
- Mohamed, A., Dahl, G. and Hinton, G. "Acoustic Modeling Using Deep Belief Networks", *IEEE Trans. Audio, Speech, & Language Proc.* Vol. 20 (1), January 2012.

- Mohamed, A., Hinton, G., and Penn, G., "Understanding how deep belief networks perform acoustic modelling," Proc. ICASSP, 2012a.
- Mohamed, A., Yu, D., and Deng, L. "Investigation of full-sequence training of deep belief networks for speech recognition," Proc. Interspeech, Sept. 2010.
- Mohamed, A., Dahl, G., and Hinton, G. "Deep belief networks for phone recognition," in Proc. NIPS Workshop Deep Learning for Speech Recognition and Related Applications, 2009.
- Morgan, N. "Deep and Wide: Multiple Layers in Automatic Speech Recognition," IEEE Trans. Audio, Speech, & Language Proc. Vol. 20 (1), January 2012.
- Morgan, N., Q. Zhu, A. Stolcke, K. Sonmez, S. Sivasdas, T. Shinozaki, M. Ostendorf, P. Jain, H. Hermansky, D. Ellis, G. Doddington, B. Chen, O. Cretin, H. Bourlard, , and M. Athineos, "Pushing the envelope - aside [speech recognition]," IEEE Signal Processing Magazine, vol. 22, no. 5, pp. 81–88, Sep 2005.
- Murphy, K. Machine Learning – A Probabilistic Perspective, The MIT Press, 2012.
- Nair, V. and Hinton, G. "3-d object recognition with deep belief nets," Proc. NIPS, 2009.
- Ney, H. "Speech translation: Coupling of recognition and translation," Proc. ICASSP, 1999.
- Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., and Ng, A. "Multimodal deep learning," Proc. ICML, 2011.
- Ngiam, J., Chen, Z., Koh, P., and Ng, A. "Learning deep energy models," Proc. ICML, 2011.
- Oliver, N., Garg, A., and Horvitz, E. "Layered Representations for Learning and Inferring Office Activity from Multiple Sensory Channels," Computer Vision and Image Understanding," vol. 96, pp. 163-180, 2004.
- Olshausen, B. "Can 'Deep Learning' offer deep insights about Visual Representation?" NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2012.
- Ostendorf, V. Digalakis, and O. Kimball, "From HMM's to segment models: A unified view of stochastic modeling for speech recognition," IEEE Trans. Speech and Audio Proc., vol. 4, no. 5, September 1996.
- Papandreou, G., Katsamanis, A., Pitsikalis, V., and Maragos, P. "Adaptive multimodal fusion by uncertainty compensation with application to audiovisual speech recognition," IEEE Trans. Audio, Speech, and Lang. Processing, Vol.17(3), pp. 423-435, 2009.
- Peng, J., Bo, L., and Xu, J. "Conditional neural fields," Proc. NIPS, 2009.
- Picone, P., S. Pike, R. Regan, T. Kamm, J. bridle, L. Deng, Z. Ma, H. Richards, and M. Schuster, "Initial evaluation of hidden dynamic models on conversational speech," Proc. ICASSP, 1999.
- Pinto, J., Garimella, S., Magimai-Doss, M., Hermansky, H., and Bourlard, H. "Analysis of MLP-based hierarchical phone posterior probability estimators," IEEE Trans. Audio, Speech, and Language Proc., vol. 19, no. 2, Feb. 2011.

- Poggio, T. "How the Brain Might Work: The Role of Information and Learning in Understanding and Replicating Intelligence," In: *Information: Science and Technology for the New Century*, Editors: G. Jacovitt, A. Pettorossi, R. Consolo and V. Senni, Lateran University Press, pp. 45-61, 2007.
- Poon, H. and Domingos, P. "Sum-product networks: A new deep architecture," *Proc. Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, 2011. Barcelona, Spain.
- Povey, D. and Woodland, P. "Minimum phone error and I-smoothing for improved discriminative training," *Proc. ICASSP*, 2002, pp. 105–108.
- Prabhavalkar, R. and Fosler-Lussier, E. "Backpropagation training for multilayer conditional random field based phone recognition", *Proc. ICASSP 2010*, pp. 5534-5537.
- Ranzato, M., Chopra, S. and LeCun, Y., and Huang, F.-J. "Energy-based models in document recognition and computer vision," *Proc. International Conference on Document Analysis and Recognition (ICDAR)*, 2007.
- Ranzato, M., Boureau, Y., and LeCun, Y. "Sparse Feature Learning for Deep Belief Networks," *Proc. NIPS*, 2007.
- Ranzato, M., Susskind, J., Mnih, V., and Hinton, G. "On deep generative models with applications to recognition," *Proc. CVPR*, 2011.
- Rennie, S., Hershey, H., and Olsen, P. "Single-channel multi-talker speech recognition — Graphical modeling approaches," *IEEE Signal Processing Mag.*, vol. 33, pp. 66–80, 2010.
- Rifai, S., Vincent, P., X. Muller, X. Glorot, and Y. Bengio, "Contractive autoencoders: Explicit invariance during feature extraction," *Proc. ICML*, 2011, pp. 833-840.
- Robinson, A. "An application of recurrent nets to phone probability estimation," *IEEE Trans. Neural Networks*, Vol. 5, pp. 298-305, 1994.
- Sainath, T., Ramabhadran, B., Picheny, M., Nahamoo, D., and Kanevsky, D., "Exemplar-Based Sparse Representation Features: From TIMIT to LVCSR," *IEEE Transactions on Speech and Audio Processing*, November 2011.
- Sainath, T., Kingbury, B., Ramabhadran, B., Novak, P., and Mohamed, A. "Making deep belief networks effective for large vocabulary continuous speech recognition," *Proc. IEEE ASRU*, 2011.
- Sainath, T., Mohamed, A., Kingsbury, B., and Ramabhadran, B. "Convolutional neural networks for LVCSR," *Proc. ICASSP*, 2013.
- Salakhutdinov R. and Hinton, G. "Semantic hashing," *Proc. SIGIR Workshop on Information Retrieval and Applications of Graphical Models*, 2007.
- Salakhutdinov R. and Hinton, G. "Deep Boltzmann machines," *Proc. AISTATS*, 2009.
- Salakhutdinov R. and Hinton, G. "A better way to pretrain deep Boltzmann machines," *Proc. NIPS*, 2012.

- Sarikaya, R., Hinton, G., Ramabhadran, B. "Deep belief nets for natural language call-routing," Proc. ICASSP, pp. 5680-5683, 2011.
- Seide, F., Li, G., Chen, X., and Yu, D. "Feature engineering in context-dependent deep neural networks for conversational speech transcription," Proc. ASRU 2011, pp. 24-29.
- Seide, F., Li, G., and Yu, D. "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks," Interspeech 2011, pp. 437-440.
- Shannon, M., Zen, H., and Byrne W. "Autoregressive models for statistical parametric speech synthesis," IEEE Trans. Audio, Speech, Language Proc., Vol. 21, No. 3, 2013, pp. 587-597.
- Sheikhzadeh, H. and Deng, L. "Waveform-based speech recognition using hidden filter models: Parameter selection and sensitivity to power normalization," IEEE Trans. on Speech and Audio Processing, Vol. 2, pp. 80-91, 1994.
- Siniscalchi, M., Yu, D., Deng, L., and Lee, C.-H. "Exploiting deep neural networks for detection-based speech recognition," Neurocomputing, 2013.
- Siniscalchi, M., Svendsen, T., and Lee, C.-H. "A bottom-up modular search approach to large vocabulary continuous speech recognition," IEEE Trans. Audio, Speech, Language Proc., Vol. 21, 2013a.
- Sivaram G. and Hermansky, H. "Sparse multilayer perceptron for phoneme recognition," IEEE Trans. Audio, Speech, & Language Proc. Vol. 20 (1), January 2012.
- Snoek, J., Larochelle, H., and Adams, R. "Practical Bayesian Optimization of Machine Learning Algorithms," Proc. NIPS, 2012.
- Socher, R. "New Directions in Deep Learning: Structured Models, Tasks, and Datasets," NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2012.
- Socher, R., Lin, C., Ng, A., and Manning, C. "Learning continuous phrase representations and syntactic parsing with recursive neural networks," Proc. ICML, 2011.
- Socher, R., Pennington, J., Huang, E., Ng, A., and Manning, C. "Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions," Proc. EMNLP, 2011a.
- Socher, R., Pennington, J., Huang, E., Ng, A., and Manning, C. "Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection, Proc. NIPS 2011b.
- Socher, R., Bengio, Y., and Manning, C. "Deep learning for NLP," Tutorial at ACL, 2012, <http://www.socher.org/index.php/DeepLearningTutorial/DeepLearningTutorial>.
- Stoyanov, V., Ropson, A. and Eisner, J. "Empirical Risk Minimization of Graphical Model Parameters Given Approximate Inference, Decoding, and Model Structure," Proc. AISTAT, 2011.
- Srivastava, N. and Salakhutdinov R. "Multimodal learning with deep Boltzmann machines," Proc. NIPS, 2012.
- Sutskever, I. "Training Recurrent Neural Networks," Ph.D. Thesis, University of Toronto, 2013.



- Sutskever, I., Martens J., and Hinton, G. "Generating text with recurrent neural networks," Proc. ICML, 2011.
- Taylor, G., Hinton, G. E., and Roweis, S. "Modeling human motion using binary latent variables." Proc. NIPS, 2007.
- Tang, Y. and Eliasmith, C. "Deep networks for robust visual recognition," Proc. ICML, 2010.
- Taralba, A, Fergus R, and Weiss, Y. "Small codes and large image databases for recognition," Proc. CVPR, 2008.
- Tur, G., Deng, L., Hakkani-Tür, D., and X. He. "Towards deep understanding: Deep convex networks for semantic utterance classification," Proc. ICASSP, 2012.
- Vincent, P. "A connection between score matching and denoising autoencoder", Neural Computation, Vol. 23, No. 7, pp. 1661-1674, 2011.
- Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P. "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," J. Machine Learning Research, Vol. 11, 2010, pp. 3371-3408.
- Vinyals, O., & Povey, D. "Krylov Subspace Descent for Deep Learning," Proc. AISTAT, 2012.
- Vinyals, O., Jia, Y., Deng, L., and Darrell, T. "Learning with recursive perceptual representations," Proc. NIPS, 2012.
- Vinyals O., and Ravuri, S. "Comparing multilayer perceptron to deep belief network tandem features for robust ASR," Proc. ICASSP, 2011.
- Welling, M., Rosen-Zvi, M., and Hinton, G. "Exponential family harmoniums with an application to information retrieval," Proc. NIPS, Vol. 20, 2005.
- Wohlmayr, M., Stark, M., Pernkopf, F. "A probabilistic interaction model for multi-pitch tracking with factorial hidden Markov model," IEEE Trans. Audio, Speech, and Language Proc., vol. 19, no. 4, May. 2011.
- Wolpert, D. "Stacked generalization," Neural Networks, 5(2), pp. 241-259, 1992.
- Xiao, L. and Deng, L. "A geometric perspective of large-margin training of Gaussian models," IEEE Signal Processing Magazine, vol. 27, no. 6, pp. 118-123, IEEE, November 2010.
- Yamin, S., Deng, L., Wang, Y., and Acero, A. "An integrative and discriminative technique for spoken utterance classification," IEEE Trans. Audio, Speech, and Language Proc., 2008.
- Yang, D., Furui, S. "Combining a two-step CRF model and a joint source channel model for machine transliteration," Proc. ACL, Uppsala, Sweden, 2010, pp. 275-280.
- Yu, D., Deng, L., and Seide, F. "The deep tensor neural network with applications to large vocabulary speech recognition," IEEE Trans. Audio, Speech, Lang. Proc., 2013.
- Yu, D. and Deng, L. "Efficient and effective algorithms for training single-hidden-layer neural networks," Pattern Recognition Letters, 2012.

- Yu, D., Seide, F., Li, G., Deng, L. "Exploiting sparseness in deep neural networks for large vocabulary speech recognition," Proc. ICASSP 2012.
- Yu, D., Siniscalchi, S., Deng, L., and Lee, C. "Boosting attribute and phone estimation accuracies with deep neural networks for detection-based speech recognition", Proc. ICASSP 2012.
- Yu, D., Chen, X., and Deng, L., "Factorized deep neural networks for adaptive speech recognition," International Workshop on Statistical Machine Learning for Speech Processing, March 2012.
- Yu, D. and Deng, L. "Deep learning and its applications to signal and information processing," IEEE Signal Processing Magazine, January 2011, pp. 145-154.
- Yu, D. and Deng, L. "Accelerated parallelizable neural networks learning algorithms for speech recognition," Proc. Interspeech 2011.
- Yu, D., Deng, L., Li, G., and F. Seide. "Discriminative pretraining of deep neural networks," U.S. Patent Filing, Nov. 2011.
- Yu, D. and Deng, L. "Deep-structured hidden conditional random fields for phonetic recognition," Proc. Interspeech, Sept. 2010.
- Yu, D., Wang, S., Karam, Z., Deng, L. "Language recognition using deep-structured conditional random fields," Proc. ICASSP, 2010, pp. 5030-5033.
- Yu, D., Wang, S., Deng, L., "Sequential labeling using deep-structured conditional random fields", J. of Selected Topics in Signal Processing, 2010a.
- Yu, D., Li, J.-Y., and Deng, L. "Calibration of confidence measures in speech recognition," IEEE Trans. Audio, Speech and Language, 2010b.
- Yu, D., Deng, L., and Dahl, G.E., "Roles of Pre-Training and Fine-Tuning in Context-Dependent DBN-HMMs for Real-World Speech Recognition," NIPS 2010 Workshop on Deep Learning and Unsupervised Feature Learning, Dec. 2010c.
- Yu, D., Deng, D., Wang, S., "Learning in the Deep-Structured Conditional Random Fields," NIPS 2009 Workshop on Deep Learning for Speech Recognition and Related Applications, 2009.
- Yu, D., Deng, L., Gong, Y. and Acero, A. "A novel framework and training algorithm for variable-parameter hidden Markov models," IEEE Transactions on Audio, Speech and Language Processing, vol. 17, no. 7, September 2009, pp. 1348-1360.
- Yu, D., Deng, L., Liu, P., Wu, J., Gong, Y., and Acero, A. "Cross-lingual speech recognition under runtime resource constraints," Proc. ICASSP, 2009.
- Yu, D. and Deng, L. "Solving nonlinear estimation problems using Splines," IEEE Signal Processing Magazine, vol. 26, no. 4, pp. 86-90, July 2009.
- Zamora-Martínez, F., Castro-Bleda, M., España-Boquera, S. "Fast evaluation of connectionist language models," Intern. Conf. Artificial Neural Networks, 2009, pp. 144-151.

- Zen, H., Nankaku, Y., and Tokuda, K. "Continuous stochastic feature mapping based on trajectory HMMs," IEEE Trans. Audio, Speech, and Language Proc., vol. 19, no. 2, Feb. 2011, pp. 417-430.
- Zen, H. Gales, M. J. F. Nankaku, Y. Tokuda, K. "Product of experts for statistical parametric speech synthesis," IEEE Trans. Audio, Speech, and Language Proc., vol. 20, no. 3, March, 2012, pp. 794-805.
- Zweig, G. and Nguyen, P. "A segmental CRF approach to large vocabulary continuous speech recognition," Proc. ASRU, 2009.