Online Learning of Deep Hybrid Architectures for Semi-Supervised Categorization

Alexander G. Ororbia II, David Reitter,
Jian Wu, and C. Lee Giles

College of Information Sciences and Technology,
The Pennsylvania State University, University Park, PA, 16802
{ago109,reitter,jxw394,giles}@psu.edu

Abstract. A hybrid architecture is presented capable of online learning from both labeled and unlabeled samples. It combines both generative and discriminative objectives to derive a new variant of the Deep Belief Network, i.e., the Stacked Boltzmann Experts Network model. The model's training algorithm is built on principles developed from hybrid discriminative Boltzmann machines and composes deep architectures in a greedy fashion. It makes use of its inherent "layer-wise ensemble" nature to perform useful classification work. We (1) compare this architecture against a hybrid denoising autoencoder version of itself as well as several other models and (2) investigate training in the context of an incremental learning procedure. The best-performing hybrid model, the Stacked Boltzmann Experts Network, consistently outperforms all others.

Keywords: restricted boltzmann machines, denoising autoencoders, semi-supervised learning, incremental learning, hybrid architectures

1 Introduction

When it comes to collecting information from unstructured data sources, the challenge is clear for any information harvesting agent: to recognize what is relevant and to categorize what has been found. For applications such as web crawling, models such as the competitive Support Vector Machine are often trained on labeled datasets [6]. However, as the target distribution (such as that of information content from the web) evolves, these models quickly become outdated and require re-training on new datasets. Simply put, while unlabeled data is plentiful, labeled data is not [28]. While incremental approaches such as co-training [15] have been employed to face this challenge, they require careful, time-consuming feature-engineering (to construct multiple views of the data).

To minimize the human effort in gathering data and facilitate scalable learning, a model capable of generalization with only a few labeled examples and vast quantities of easily-acquired unlabeled data is highly desirable. Furthermore, to avoid feature engineering, this model should exploit the representational power afforded by deeper architectures, which have seen success in domains such as computer vision and speech recognition. Such a multi-level model could learn feature abstractions, arguably capturing higher-order feature relationships in an efficient manner. In pursuit of this goal, our

contribution is the development of a novel hybrid Boltzmann-based architecture and its hybrid denoising autoencoder variant as well as their incremental, semi-supervised learning algorithms and prediction mechanisms. The learning process makes use of compound learning objectives, balancing, in a parametric fashion, the dual goals of generative and discriminative modeling of data. We further experiment with relaxing our approach's strict bottom-up scheme to better handle the online data-stream setting.

The rest of this paper is organized in the following manner. First, we review relevant previous work applying deep models to categorization problems in Section 2. Following this, in Section 3, we describe the algorithmic mechanics of our two incremental, semi-supervised deep architectures. Experimental results of using these deep architectures in a variety of data contexts are presented in Section 4. We sum up our work in Section 5 and consider model limitations and potential algorithmic improvements.

2 Related Work

Our algorithms fall in the realm of representation-learning, designed to learn, "...transformations of the data that make it easier to extract useful information when building classifiers or other predictors" [2]. Shallow learning methods, which require extensive prior human knowledge and large, labeled datasets, have been argued to be limited in terms of learning functions that violate restrictive assumptions such as smoothness and locality [4]. Moreover, architectures with a single unobserved layer require an exponentially increasing number of units to accurately learn complex distributions that deeper architectures, composed of multiple layers of non-linearity, potentially can. Deeper models "exploit the unknown structure" of the input data distribution to generate high-level features that are invariant to most variations in the training examples and yet preserve as much information regarding the input as possible [1].

In both large-scale, image-based [18, 39] and language-based problems [31, 24, 14, 26], deep architectures have outperformed popular shallow models. However, these models operate in a multi-stage learning process, where a generative architecture is greedily pre-trained and then used to initialize parameters of a second architecture that is discriminatively fine-tuned. To help deep models deal with potentially uncooperative input distributions or encourage learning of discriminative information earlier in the learning process, some approaches have leveraged auxiliary models in various ways [3, 41, 22]. A few methods have been proposed for adapting deep architecture construction to incremental learning settings [5, 42]. Furthermore, an interesting approach combined the simple idea of *pseudo-labeling* with training deep neural architectures composed of rectified linear activation functions [13], a recent advancement [23].

While fundamentally different compared to purely generative or discriminative ones [21], we hypothesize that deep hybrid models that balance multiple objectives similar to the shallow one in [19] can make good, semi-supervised incremental models for classification. Motivated by this hypothesis, we design two model candidates, building on principles and successes of previous work: the *Stacked Boltzmann Expert Network* (SBEN) and the *Hybrid Stacked Denoising Autoencoders* model (HSDA). Furthermore, we introduce the idea of *layer-wise ensembling*, a simple prediction scheme we shall describe in Section 3.3 to utilize layer-wise information learnt by these models.

3 Deep Hybrid Architectures

In this section, we describe the implementations of our deep hybrid architectures.

3.1 The Stacked Boltzmann Experts Network (SBEN)

Our proposed variant of the Deep Belief Network (DBN), the Stacked Boltzmann Experts Network, follows an approach to construction and training similar to the DBN itself. The key is to, in an efficient, greedy manner, learn a stack of building-block models, and, as a layer is modified, freeze the parameters of lower layers. In practice, this is done by propagating data samples up to the layer targeted for layer-wise training and using the resultant latent representations as observations for constructing a higher level model. In contrast to the DBN, which stacks restricted Boltzmann machines (RBM's) and is often used to initialize a deep multi-layer perceptron (MLP), the SBEN model is constructed by composing hybrid restricted Boltzmann machines and is directly applied to the discriminative task and potentially fine-tuned directly[1].

The hybrid restricted Boltzmann machine (HRBM) [34, 19, 20], originally referred to as the *ClassRBM*, formalized the RBM extended to handle classification tasks directly. The model has been studied and used in a wide variety of applications [25, 12, 8, 38] including the top of a DBN [36], most of which focus on the directly supervised facet of the model. With defined parameters[2] $\Theta = (\mathbf{W}, \mathbf{U}, \mathbf{b}, \mathbf{c}, \mathbf{d})$, the HRBM is designed to model the joint distribution of a binary pattern vector $\mathbf{x} = (x_1, \cdots, x_D)$ and its corresponding target variable $y \in \{1, \cdots, C\}$ utilizing a set of latent variables $\mathbf{h} = (h_1, \cdots, h_H)$. The HRBM assigns a probability to the triplet (y,$\mathbf{x}$,$\mathbf{h}$) using:

$$p(y, \mathbf{x}, \mathbf{h}) = \frac{e^{-E(y,\mathbf{x},\mathbf{h})}}{Z}, \text{ with, } p(y, \mathbf{x}) = \frac{1}{Z} \sum_{\mathbf{h}} e^{-E(y,\mathbf{x},\mathbf{h})} \tag{1}$$

where $Z = \sum_{(y,\mathbf{x},\mathbf{h})} e^{-E(y,\mathbf{x},\mathbf{h})}$ is the partition function meant to ensure that the value assignment is a valid probability distribution. Noting that the $\mathbf{e}_y = (\mathbf{1}_{i=y})_{i=1}^{C}$ is the one-hot vector encoding of y, the model's energy function may be defined as

$$E(y, \mathbf{x}, \mathbf{h}) = -\mathbf{h}^T \mathbf{W} \mathbf{x} - \mathbf{b}^T \mathbf{x} - \mathbf{c}^T \mathbf{h} - \mathbf{d}^T \mathbf{e}_y - \mathbf{h}^T \mathbf{U} \mathbf{e}_y. \tag{2}$$

It is often not possible to compute $p(y, \mathbf{x}, \mathbf{h})$ or the marginal $p(y, \mathbf{x})$ due to the intractable partition function. However, we may leverage block Gibbs sampling to draw samples of the HRBM's latent variable layer given the current state of the visible layer (composed of $\mathbf{x}$ and $\mathbf{e}_y$) and vice versa, owing to the graphical model's bipartite structure (i.e., no intra-layer connections). This yields implementable equations for conditioning on various layers of the model as follows:

[1] We have developed an algorithm to fine-tune the SBEN jointly, but leave usage and evaluation of this for future work.

[2] $\mathbf{W}$ is the input-to-hidden weight matrix, $\mathbf{U}$ the hidden-to-class weight matrix, $\mathbf{b}$ the visible bias vector, $\mathbf{c}$ the hidden unit bias vector, and $\mathbf{d}$ the class unit bias vector.

$$p(\mathbf{h}|y,\mathbf{x}) = \prod_j p(h_j|y,\mathbf{x}), \text{ with } p(h_j = 1|y,\mathbf{x}) = \sigma(c_j + U_{jy} + \sum_i W_{ji}x_i) \quad (3)$$

$$p(\mathbf{x}|\mathbf{h}) = \prod_i p(x_i|\mathbf{h}), \text{ with } p(x_i = 1|\mathbf{h}) = \sigma(b_i + \sum_j W_{ji}h_j) \quad (4)$$

$$p(y|\mathbf{h}) = \frac{e^{d_y + \sum_j U_{jy}h_j}}{\sum_{y^\star} e^{d_{y^\star} + \sum_j U_{jy^\star}h_j}} \quad (5)$$

where $\sigma(v) = 1/(1+e^{-v})$. Furthermore, to perform classification directly using the HRBM, one uses the model's free energy function $F(y,\mathbf{x})$ to compute the conditional

$$p(y|\mathbf{x}) = \frac{e^{-F(y,\mathbf{x})}}{\sum_{y^\star \in \{1,\cdots,C\}} e^{-F(y^\star,\mathbf{x})}} \quad (6)$$

where $-F(y,\mathbf{x}) = (d_y + \sum_j \log(1 + \exp(c_j + U_{jy} + \sum W_{ji}x_i)))$.

The hybrid model is trained leveraging a supervised, compound objective loss function that balances a discriminative objective $\mathcal{L}_{disc}$ and generative objective $\mathcal{L}_{gen}$, defined as follows:

$$\mathcal{L}_{disc}(\mathcal{D}_{train}) = -\sum_{t=1}^{|\mathcal{D}_{train}|} \log p(y|\mathbf{x}_t) \quad (7)$$

$$\mathcal{L}_{gen}(\mathcal{D}_{train}) = -\sum_{t=1}^{|\mathcal{D}_{train}|} \log p(y_t, \mathbf{x}_t) \quad (8)$$

where $\mathcal{D}_{train} = \{(y_t, \mathbf{x}_t)\}$, the labeled training dataset. The gradient for $\mathcal{L}_{disc}$ may be computed directly, following the general form

$$\frac{\partial \log p(y_t|\mathbf{x})}{\partial \theta} = -\mathbb{E}_{\mathbf{h}|y_t,\mathbf{x}_t}\left[\frac{\partial}{\partial \theta}\left(\mathbb{E}(y_t,\mathbf{x}_t,\mathbf{h})\right)\right] + \mathbb{E}_{y,\mathbf{h}|,\mathbf{x}}\left[\frac{\partial}{\partial \theta}\left(\mathbb{E}(y,\mathbf{x},\mathbf{h})\right)\right] \quad (9)$$

implemented via direct formulation (see [20] for details) or a form of *Dropping*, such as *Drop-Out* or *Drop-Connect* [37]. The generative gradient follows the form

$$\frac{\partial \log p(y_t,\mathbf{x})}{\partial \theta} = -\mathbb{E}_{\mathbf{h}|y_t,\mathbf{x}_t}\left[\frac{\partial}{\partial \theta}\left(\mathbb{E}(y_t,\mathbf{x}_t,\mathbf{h})\right)\right] + \mathbb{E}_{y,\mathbf{x},\mathbf{h}}\left[\frac{\partial}{\partial \theta}\left(\mathbb{E}(y,\mathbf{x},\mathbf{h})\right)\right] \quad (10)$$

and, although intractable for any $(y_t, \mathbf{x}_t)$, is approximated via contrastive divergence [17], where the intractable second expectation is replaced by a point estimate using one Gibbs sampling step (after initializing the Markov Chain at the training sample).

In the semi-supervised context, where $\mathcal{D}_{train}$ is small but a large, unlabeled dataset $\mathcal{D}_{unlab}$ is available, the HRBM can be further extended to train with an unsupervised objective $\mathcal{L}_{unsup}$, where negative log-likelihood is optimized according to

$$\mathcal{L}_{unsup}(\mathcal{D}_{unlab}) = -\sum_{t=1}^{|\mathcal{D}_{unlab}|} \log p(\mathbf{x}_t). \quad (11)$$

Algorithm 1 Contrastive Divergence: Single update for HRBM generative objective.

Input: training sample $(y_t, \mathbf{x}_t)$, HRBM current model parameters Θ
// Note that "$a \leftarrow b$" indicates assignment and "$a \sim b$" indicates a is sampled from b
function COMPUTEGENERATIVEGRADIENT($y_t, \mathbf{x}_t, \Theta$)
  if $y_t = \emptyset$ **then**
    $y_t \sim p(y|\mathbf{x})$ ▷ Obtain a pseudo-label for the unlabeled sample.
  // Conduct Positive Phase
  $y^0 \leftarrow y_t, \mathbf{x}^0 \leftarrow \mathbf{x}_t, \widehat{\mathbf{h}}^0 \leftarrow \sigma(\mathbf{c} + W\mathbf{x}^0 + U\mathbf{e}_{y^0})$
  // Conduct Negative Phase
  $\mathbf{h}^0 \sim p(\mathbf{h}|y^0, \mathbf{x}^0), y^1 \sim p(y|\mathbf{h}^0), x^1 \sim p(\mathbf{x}|\mathbf{h}^0)$
  $\widehat{\mathbf{h}}^1 \leftarrow \sigma(\mathbf{c} + W\mathbf{x}^1 + U\mathbf{e}_{y^1})$
  // Compute Gradient Update
  for $\theta \in \Theta$ **do**
    $\bigtriangledown \leftarrow \frac{\partial}{\partial\theta}\mathbb{E}(y^0, \mathbf{x}^0, \widehat{\mathbf{h}}^0) - \frac{\partial}{\partial\theta}\mathbb{E}(y^1, \mathbf{x}^1, \widehat{\mathbf{h}}^1)$
  return $\bigtriangledown$

The gradient for $\mathcal{L}_{unsup}$ can be simply computed using the same contrastive divergence update for $\mathcal{L}_{gen}$ but incorporating an extra step at the beginning by sampling from the model's current estimate of $p(y|\mathbf{u})$ for an unlabeled sample $\mathbf{u}$. This form of the generative update could be viewed as a form of self-training or *Entropy Regularization* [23]. The pseudo-code for the online procedure for computing the generative gradient (either labeled or unlabeled example) for a single HRBM is shown in Algorithm 1.

To train a fully semi-supervised HRBM, one composes the appropriate multi-objective function using a simple weighted summation:

$$\mathcal{L}_{semi}(\mathcal{D}_{train}, \mathcal{D}_{unlab}) = \gamma\mathcal{L}_{disc}(\mathcal{D}_{train}) + \alpha\mathcal{L}_{gen}(\mathcal{D}_{train}) + \beta\mathcal{L}_{unsup}(\mathcal{D}_{unlab}) \quad (12)$$

where α and β are coefficient handles designed to explicitly control the effects that the generative gradients have on the HRBM's learning procedure. We introduced the additional coefficient γ as a means to also directly control the effect of the discriminative gradient in model training. Setting $\gamma = 0$ leads to constructing a purely generative model of $\mathcal{D}_{train}$ and $\mathcal{D}_{unsup}$, and further setting $\beta = 0$ leads to purely supervised generative modeling of labeled dataset $\mathcal{D}_{train}$. If the target task is classification, then γ may be set to any value in $(0, 1]$ (for simplicity, we chose $\gamma = 1$, although future work shall investigate building models with values of this coefficient that shift the balance to models that favor generative features a bit more). These free parameters, though making model selection a bit more challenging, offer an explicit means of controlling the extent to which the final parameters discovered are influenced by generative learning [20], much in contrast to simple generative pre-training of neural architectures.

As mentioned before, to compose our *N*-layer SBEN (or *N*-SBEN), one follows the same greedy, layer-wise procedure of a DBN. However, unlike DBN's, where stacking RBM's warrants only a direct feedforward operation (since RBM's contain only a single set of inputs), modifications must be made to account for the architectural design of the HRBM graphical model. In order to unify the SBEN architecture while respecting

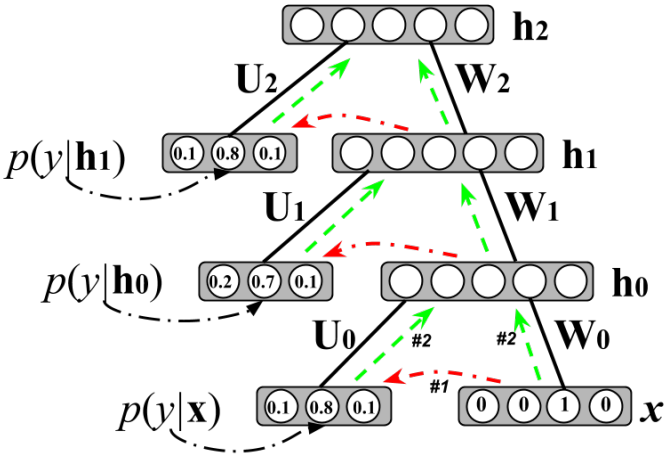

Fig. 1. Architecture of the SBEN model. The flow of data through the system is indicated by the numbered arrows. Given a sample $\mathbf{x}$, the dash-dotted arrow indicates obtaining an estimated label by using the current layer's conditional via Equation 6 (i.e., Step **# 1**, dash-dotted red arrow). The latent representation is computed using this proxy label and the data vector via Equation 3 (i.e.,Step **# 2**, dashed green arrow). This procedure is repeated recursively, replacing $\mathbf{x}$ with $\mathbf{h}_n$.

HRBM building block design, one must combine Equations 3 and 6 to properly compute intermediate data representations during training and prediction. This gives rise to the architecture as depicted in Fig. 1 and its corresponding learning procedure in Algorithm 2, where the representation for the layer above cannot be computed without first obtaining an estimate of the current layer's $p(y|\mathbf{x})$. [3] Each HRBM layer of the SBEN is greedily trained using the frozen latent representations of the one below, generated by using lower level expert's inputs and predictions.

The generative objectives (for both unlabeled and labeled samples) of our model can be viewed as a form of data-dependent regularization acting on the discriminative learning gradient of each layer. One key advantage of SBEN training is that each layer's discriminative progress may be tracked directly, since each layer-wise expert is capable of direct classification using Equation 6 to compute the conditional $p(y|\mathbf{h}_{below})$. Note that setting the number of hidden layers equal to 1 recovers the original HRBM architecture (a *1*-SEBN). One may notice some similarity with the partially supervised, layer-wise procedure of [3] where a simple softmax classifier was loosely coupled with each RBM of a DBN. However, this only served as a temporary mechanism for pre-training whereas the SBEN leverages the more unified framework of the HRBM during and after training. Note that inputs to the SBEN, like the DBN, can be trivially extended [40, 29].

A useful property of the SBEN model is that it also contains a generative model "facet" due to its hybrid nature. One could treat this facet as a directed, top-down gen-

[3] One may sample from this prediction vector as one would for hidden activations, however, we found that simply using this mean in forward propagation step yielded best results.

Algorithm 2 Greedy, layer-wise construction of an N-SBEN, where N is the desired number of layers of latent variables.

Input: $\mathcal{D}_{train}$, $\mathcal{D}_{unlab}$, learning rate λ and hyper-parameters γ, α, β, ***numSteps***, and initial model parameters $\Theta = \{\Theta_1, \Theta_2, ..., \Theta_N\}$
function CONSTRUCTMODEL($\mathcal{D}_{train}$, $\mathcal{D}_{unlab}$, λ, γ, α, β, $numSteps$, Θ)
  $\mathcal{D}^n_{train} \leftarrow \mathcal{D}_{train}$, $\mathcal{D}^n_{unlab} \leftarrow \mathcal{D}_{unlab}$ ▷ Initialize subsets to low-level representations
  for $\Theta_n \in \Theta$ **do**
    $t \leftarrow 0$
    while $t \leq numSteps$ **do**
      $(y_t, \mathbf{x}_t) \sim \mathcal{D}^n_{train}$ ▷ Draw sample from $\mathcal{D}^n_{train}$ without replacement
      $(\mathbf{u}_t) \sim \mathcal{D}^n_{unlab}$ ▷ Draw sample from $\mathcal{D}^n_{unlab}$ without replacement
      $(\bigtriangledown_{disc}, \bigtriangledown_{gen}, \bigtriangledown_{unsup}) \leftarrow$ UPDATELAYER$(y_t, \mathbf{x}_t, \mathbf{u}_t, \Theta_n)$
      $\Theta_n \leftarrow \Theta_n + \lambda(-\gamma \bigtriangledown_{disc} + \alpha \bigtriangledown_{gen} + \beta \bigtriangledown_{unsup})$, $t \leftarrow t + 1$
    $\mathcal{D}^h_{train} \leftarrow \emptyset$, $\mathcal{D}^h_{unlab} \leftarrow \emptyset$
    for $(y_t, \mathbf{x}_t) \in \mathcal{D}^n_{train}$ **do** ▷ Compute latent representation dataset for $\mathcal{D}^n_{train}$
      $\mathcal{D}^h_{train} \leftarrow$ COMPUTELATENTREPRESENTATION$(y_t, \mathbf{x}_t, \Theta_n)$
    for $(\emptyset, \mathbf{u}_t) \in \mathcal{D}^n_{unlab}$ **do** ▷ Compute latent representation dataset for $\mathcal{D}^n_{unlab}$
      $\mathcal{D}^h_{unlab} \leftarrow$ COMPUTELATENTREPRESENTATION$(\emptyset, \mathbf{u}_t, \Theta_n)$
    $\mathcal{D}^n_{train} \leftarrow \mathcal{D}^h_{train}$, $\mathcal{D}^n_{unlab} \leftarrow \mathcal{D}^h_{unlab}$
function UPDATELAYER$(y_t, \mathbf{x}_t, \mathbf{u}_t, \Theta_n)$
  $\bigtriangledown_{disc} \leftarrow$ COMPUTEDISCIMINATIVEGRADIENT$(y_t, \mathbf{x}_t, \Theta_n)$ ▷ See [20] for details
  $\bigtriangledown_{gen} \leftarrow$ COMPUTEGENERATIVEGRADIENT$(y_t, \mathbf{x}_t, \Theta_n)$ ▷ See Algorithm 1
  $\bigtriangledown_{unsup} \leftarrow$ COMPUTEGENERATIVEGRADIENT$(\emptyset, \mathbf{u}_t, \Theta_n)$ ▷ See Algorithm 1
  return $(\bigtriangledown_{disc}, \bigtriangledown_{gen}, \bigtriangledown_{unsup})$
function COMPUTELATENTREPRESENTATION$(y_t, \mathbf{x}_t, \Theta_n)$
  $y^h_t \leftarrow p(y_t|\mathbf{x}_t, \Theta_n)$ ▷ Equation 6 under the layerwise model
  $\mathbf{h}_t \sim p(\mathbf{h}|y^h_t, \mathbf{x}_t, \Theta_n)$ ▷ Equation 3 under the layerwise model
  return $(y_t, \mathbf{h}_t)$

erative network and generate fantasy samples for specific, clamped class units. In addition, one could generate class-specific probabilistic scores for input features by adapting the procedure in [37] and, since each layer contains class units, potentially uncover the SBEN hierarchy extracted from the data.

3.2 Hybrid Stacked Denoising Auto-encoders (HSDA)

The auto-encoder variant of the SBEN is the Hybrid Stacked Denoising Autoencoders model. Instead of building a direct model of the joint distribution of $(y_t, \mathbf{x}_t)$ as in the HRBM, the hybrid Denoising autoencoder (hDA) building block, with parameters $\Theta = (\mathbf{W}, \mathbf{W}', \mathbf{U}, \mathbf{b}, \mathbf{b}', \mathbf{d})$, may be viewed as a fusion of a generative model of $p(\mathbf{x})$, or an encoder-decoder reconstruction model, with a conditional model of $p(y|\mathbf{x})$, or a single-layer MLP. The reconstruction model learns a corrupted version of the feature vector $\mathbf{x}_t$, which is created via stochastic mapping $\widehat{\mathbf{x}}_t \sim q_{\mathcal{D}}(\widehat{\mathbf{x}}_t|\mathbf{x})$, where $q_{\mathcal{D}}$ is a function that stochastically corrupts an input vector (i.e., randomly masking entries by setting them to zero under a given probability). The reconstruction model is defined as

$$\mathbf{h} = f_\theta(\widehat{\mathbf{x}}) = \sigma(\mathbf{W}\widehat{\mathbf{x}} + \mathbf{b}) \quad (13) \qquad \mathbf{z} = g_\theta(\mathbf{h}) = \sigma(\mathbf{W}'\mathbf{h} + \mathbf{b}') \quad (14)$$

where parameters $\mathbf{W}$ and $\mathbf{W}'$ may be "tied" by setting $\mathbf{W}' = \mathbf{W}^T$. The encoder deterministically maps an input to a latent representation $\mathbf{h}$ (Equation 13) and the decoder maps this $\mathbf{h}$ back to a reconstruction $\mathbf{z}$ (Equation 14). The coupled neural network is tasked with mapping input $\mathbf{x}$ to y while sharing its latent layer $\mathbf{h}$ with the encoder-decoder pair. To compute the conditional $p(y|\mathbf{x})$, one uses Equation 13 followed by Equation 5 of the HRBM. While the discriminative objective of an hDA is defined similarly to the HRBM (Equation 7), where gradients of the log loss are computed directly (as in an MLP), the generative objective $\mathcal{L}_{gen}$ proceeds a bit differently:

$$\mathcal{L}_{gen}(\mathcal{D}_{train}) = -\sum_{t=1}^{|\mathcal{D}_{train}|} \mathbf{x}_t \log \mathbf{z}_t + (1 - \mathbf{x}_t) \log(1 - \mathbf{z}_t) \quad (15)$$

This is the cross-entropy of two independent multivariate Bernoulli distributions, or *cross-entropy loss*. Unlike the HRBM, training an hDA under this generative objective is notably simpler since it uses back-propagation combined with the loss as defined in Equaton 15. A full hDA is trained using the weighted, tri-objective framework described in Section 3.1, $\mathcal{L}_{semi}$ (Equation 12), where its unlabeled objective $\mathcal{L}_{unsup}$ uses the same cross-entropy function as $\mathcal{L}_{gen}$ but operates on samples drawn from $\mathcal{D}_{unlab}$. The semi-supervised hDA differs from the HRBM complement in not only gradient calculation but also in that its unsupervised components do not require a corresponding sample of the model's estimate of $p(y_t|\mathbf{u}_t)$ for an unlabeled sample $\mathbf{u}_t$. This is advantageous since generative gradients are computed independently of the existence of a label, saving computational time and avoiding one drawback of self-training schemes: Reinforcement of incorrect predictions through model-generated pseudo-labels.

In the same greedy, layer-wise fashion as the SBEN, the *N*-layer HSDA (*N*-HSDA) may be composed by stacking hDA's. By replacing the procedure for generative gradients (Algorithm 1) and the discriminative gradient with the appropriate autoencoder cross-entropy back-propagation alternatives and substituting Equations 6 and 3 with Equation 13 (for computing hidden activities in *COMPUTELATENTREPRESENTATION* of Algorithm 2), one may build an HSDA using Algorithm 2. The most useful property of the HSDA is that required computation for training and prediction may be reduced since dimensionality of each latent representation in auto-encoder architectures can be gradually decreased for upper levels of the network.

One may notice that the architectures of [42] and [30] may be recovered from our framework by manipulating the coefficients γ, α, and β in the $\mathcal{L}_{semi}$ objective function for the HSDA. Both these studies made use of dual-gradient models, which either focused on a hybrid objective that balanced a discriminative and weighted generative objective on a single sample (where the objective collapsed into a single generative objective when no label was available) [42] or where a generative objective was used as the primary objective and combined with a weighted discriminative objective [30]. Since our HSDA architecture can be viewed as a more general formulation of these original models, it is also amenable to their own particular extensions (such as feature growth/pruning, alternative input units for handling different types of data, etc.).

3.3 Ensembling of Layer-Wise Experts

Both the SBEN and the HSDA models, in addition to unique strengths, possess the interesting property where each layer, or *expert*, of the model is capable of classification given the appropriate latent representation of the data. This implies that the model is ensemble-like in its very nature but differs from standard ensemble methods where many smaller models are horizontally aggregated using well-established schemes such as boosting [33] or majority voting. Traditional feedforward models simply propagate data through the final network to obtain an output prediction from its penultimate layer for a given $\mathbf{x}_t$. In contrast, these hybrid models are capable of a producing a label y_t^n at each level *n* for $\mathbf{x}_t$, resulting from their layer-wise multi-objective training.

To vertically aggregate layer-wise expert outputs, we experimented with a variety of schemes in development, but found that computing a simple mean predictor, $p(y|\mathbf{x})_{ensemble}$ worked best, defined as:

$$p(y|\mathbf{x})_{ensemble} = \frac{1}{N}\sum_{n=1}^{N} p(y|\mathbf{x})_n \tag{16}$$

This ensembling scheme allows for all components of the hybrid model to play a role in classification of unseen samples, perhaps leveraging acquired discriminative "knowledge" at their respective level of abstraction in the model hierarchy to ultimately improve final predictive performance. This scheme exploits our model's inherent layer-wise discriminative ability, which stands as an alternative to coupling helper classifiers as in [3] or the "companion objectives" of [22] to solve potential exploding gradients in deep convolution networks for object detection.

4 Experimental Results

We present experimental results on several classification problems in both optical character recognition and document categorization.

Character recognition Two experiments were conducted. The first experiment uses the Stanford OCR dataset, which contains 52,152 16×8 binary pixel images labeled as 1 of 26 letters of the English alphabet. Training ($\sim 2\%$ of source), validation ($\sim 1.9\%$), unlabeled ($\sim 19.2\%$), and test sets ($\sim 77\%$) are created via a seeded random sampling without replacement procedure that ensured examples from each class appeared in roughly equal quantities in training and validation subsets. The second experiment makes use of a (seeded) stochastic process we implemented that generates 28×28 pixel CAPTCHA images of single characters based on the CAGE model [4], where one of 26 English characters may be generated (26 classes), of either lower or upper-case form in a variety of fonts as well as orientations and scales. We make use of this process in two ways: 1) create a finite dataset of 16,000 samples with ($\sim 3.125\%$ in training, $\sim 3.125\%$ in validation, $\sim 31.125\%$ in unlabeled, and $\sim 62.25\%$ in test) and perform

[4] https://akiraly.github.io/cage/index.html

an experiment similar to the OCR dataset, and 2) use the process as a controllable data-stream, which allows for compact storage of a complex distribution of image samples. The only pre-processing applied to the CAPTCHA samples was pixel binarization.

Text categorization We make use of a pre-processed WEBKB text collection (i.e., font formatting, stop words removed, terms stemmed, and words with length less than 3 removed) [7], which contains pages from a variety of universities (Cornell, Texas, Washington, and Wisconsin and miscellaneous pages from others). The 4-class classification problem as defined by this dataset will be to determine if a web-page can be identified as one belonging to a Student, Faculty, Course, or a Project. The dataset was already pre-partitioned into a training set (2,803 samples) and a test set (1,396 web pages), so using the same sampling scheme as the OCR data, we built from the training split a smaller training ($\sim 20.2\%$) and validation ($\sim 14.2\%$) subset, and put the rest into the unlabeled set ($\sim 62.5\%$), discarding 87 document vectors that contained less than 2 active features. The test set contained 1,344 samples, after discarding 52 samples with less than 2 active features. We simplified the low-level feature representation by using the top 2000 words in the corpus and binarizing the document term vectors.

Models We compare the HSDA and SBEN models to the non-linear, shallow HRBM (which, as described in Section 3.1, is a *1*-SBEN). For a simpler classifier, we implemented an incremental version of Maximum Entropy (which, as explained in [32], is equivalent to a softmax classifier), or *MaxEnt*. Furthermore, we implemented the Pegasos SVM algorithm (*SVM*) [35] and extended it to follow a proper multi-class scheme [9]. This is the online formulation of the Support Vector Machine, trained via sub-gradient descent on the primal objective followed by a projection step (note that for simplicity we built the linear-kernel version of the model, which is quite fast). Evaluating the Pegasos SVM algorithm in the following experiments allows us to compare our deep semi-supervised models against the incremental version of a strong linear-kernel classifier. To provide some context with previously established deep architectures also learnable in a 1-phase fashion like our own, we present results for a simple sparse Rectifier Network, or $Rect.$ [13]. [5] Note that we extended all shallow classifiers and the Rectifier Network to leverage self-training so that they may also learn from unlabeled examples. To do so, we implemented a scheme similar to that of [23] and used a classifier's estimate of $p(y|\mathbf{u})$ for an unlabeled sample. However, a 1-hot proxy encoding using the *argmax* of model's predictor was only created for such a sample if $max[p(y|\mathbf{u})] > \bar{p}$. We found that by explicitly controlling pseudo-labeling through $\bar{p}$ we could more directly improve model performance.

Model Selection Model selection was conducted using a parallelized multi-setting scheme, where a configuration file for each model was specified, describing a set of hyper-parameter combinations to explore (this is akin to a course-grained grid search, where points of model evaluation are set manually a priori). For the *HSDA*, *SBEN*,

[5] Model implementations were computationally verified for correctness when applicable. Since discriminative objectives entailed using an automatic differentiation framework, we checked gradient validity via finite difference approximation.

Table 1. Character identification results on the CAPTCHA simulated dataset. Classification results are reported as 10-trial averages with single standard deviation from the mean.

	Error	**Precision**	**Recall**	**F1-Score**
MaxEnt	0.475 ± 0.010	0.535 ± 0.011	0.524 ± 0.010	0.522 ± 0.010
SVM	0.461 ± 0.011	0.564 ± 0.010	0.537 ± 0.011	0.526 ± 0.011
2-Rect [13, 23]	0.365 ± 0.011	0.651 ± 0.011	0.634 ± 0.011	0.627 ± 0.013
HRBM [20]	0.368 ± 0.009	0.643 ± 0.010	0.631 ± 0.009	0.629 ± 0.009
5-SBEN	$\mathbf{0.324 \pm 0.008}$	$\mathbf{0.681 \pm 0.009}$	$\mathbf{0.675 \pm 0.008}$	$\mathbf{0.671 \pm 0.009}$
5-HSDA	0.359 ± 0.011	0.650 ± 0.011	0.640 ± 0.011	0.633 ± 0.011

Table 2. Character identification results on the Stanford OCR dataset. Classification results are reported as 10-trial averages with single standard deviation from the mean.

	Error	**Precision**	**Recall**	**F1-Score**
MaxEnt	0.425 ± 0.009	0.508 ± 0.006	0.563 ± 0.005	0.512 ± 0.006
SVM	0.428 ± 0.008	0.504 ± 0.004	0.582 ± 0.011	0.510 ± 0.007
3-Rect [13, 23]	0.387 ± 0.009	0.549 ± 0.009	0.592 ± 0.014	0.548 ± 0.011
HRBM [20]	0.399 ± 0.019	0.565 ± 0.009	0.606 ± 0.016	0.552 ± 0.014
3-SBEN	$\mathbf{0.333 \pm 0.009}$	$\mathbf{0.602 \pm 0.009}$	$\mathbf{0.668 \pm 0.009}$	$\mathbf{0.610 \pm 0.012}$
3-HSDA	0.399 ± 0.012	0.546 ± 0.007	0.601 ± 0.012	0.537 ± 0.009

Table 3. Text categorization results on the WEBKB dataset.

	Error	**Precision**	**Recall**	**F1-Score**		**Error**	**Precision**	**Recall**	**F1-Score**
MaxEnt	0.510	0.386	0.387	0.384	*3-SBEN*	**0.210**	**0.788**	0.770	**0.769**
SVM	0.524	0.404	0.378	0.387	*3-HSDA*	0.219	0.757	**0.780**	0.765

HRBM, and *Rect* we varied model architectures, exploring under-complete, complete, and over-complete versions, as well as the learning rate, α, and β coefficients (holding γ fixed at 1.0). If a model was trained using its stochastic form (i.e., HRBM, SBEN, or HSDA), to ensure reproducible model behavior, we ran it in feedforward mean-field, where no sampling steps were taken when data vectors were propagated through a network model when collecting layer-wise predictions (we also found that this yielded lowest generalization error). For the SVM, we tuned its slack variable λ. The rectifier network's training also involved using a L2 regularization penalty (0.002), initialization of hidden biases to small positive values ($|N(0, 0.25)|$) [13], and the use of the improved leaky rectifier unit [27].

For all finite dataset experiments, model performance is reported on the test set using the model with lowest validation-set error found during the training step. [6] Generalization performance was evaluated by calculating classification error, precision, recall,

[6] For the SVM, λ was varied in the interval $[0.0001, 0.5]$ while the learning rate for all other models was varied in $[0.0001, 0.1]$. For HRBM, SBEN, & HSDA, β was explored in the interval $[0.05, 0.1]$, and for HRBM, SBEN, & HSDA, α was explored in $[0.075, 1.025]$. The threshold $\bar{p}$ was varied in $[0.0, 1.0]$ and the number of latent layers N for deeper architectures was explored in $[2, 5]$ where we delineate the optimal number with the prefix "N-".

and F-Measure, where F-Measure was chosen to be the harmonic mean of precision and recall, $F1 = 2(\text{precision} \cdot \text{recall})/(\text{precision} + \text{recall})$.

Since the creation of training, validation, and unlabeled subsets was controlled through a seeded random sampling without replacement process, the procedure described above composes a single trial. For the Standford OCR and CAPTCHA datasets, the results we report are 10-trial averages with a single standard deviation from the mean, where each trial used a unique seed value.

4.1 Finite Dataset Learning Performance

On all of the datasets we experimented with, in the case when all samples are available a priori, ranging from vision-based tasks to text classification, we observe that hybrid incremental architectures have, in general, lower error as compared to non-hybrid ones. In the CAPTCHA experiment (Table 1), we observed that both the SBEN and HSDA models reduced prediction error over the SVM by nearly 30% and 22% respectively. Furthermore, both models consistently improved over the error the HRBM, with the SBEN model reducing error by $\sim 12\%$. In the OCR dataset (Table 2), we see the SBEN improving over the HRBM by more than 16% and the SVM by more than 22%. In this case, the HSDA only marginally improves over the SVM model ($\sim 6\%$) and equal to that of an HRBM, the poor performance we attribute to a coarse search through a meta-parameter space window as opposed to an exhaustive grid search. For WEBKB problem, over the *MaxEnt* model (which slightly outperformed the SVM itself), we see a $\sim 57\%$ improvement in error for the HSDA and $\sim 58\%$ for the SBEN (Table 3). Note that the rectifier network is competitive, however, in both image-based experiments, the SBEN model outperforms it by more than 11% on CAPTCHA and nearly 14% on OCR.

4.2 Incremental Learning Performance

In the online learning setting, samples from $\mathcal{D}_{unlab}$ may not be available at once and instead available at a given rate in a stream for a single time instant (we chose to experiment with one example presented at a given iteration and only constant access to a $|\mathcal{D}_{train}| = 500$). In order to train a deep architecture in this setting, while still exploiting the efficiency of a greedy, layer-wise approach, one may remove the "freezing" step of Algorithm 2 and train all layers dis-jointly in an incremental fashion as opposed to a purely bottom-up approach. Using the same sub-routines as depicted in Algorithm 2, this procedure may be implemented as shown in Algorithm 3, effectively using a single bottom-up pass to modify model parameters. This approach adapts the training of hybrid architectures, such as the SBEN and HSDA, to the online learning setting.

As evidenced by Fig. 2, it is possible to train the layer-wise experts of a multi-level hybrid architecture simultaneously and still obtain a gain in generalization performance over a non-linear, shallow model such as the HRBM. The HRBM settles at an online error of 0.356 whereas the 5-HSDA reaches an error of 0.327 and the 5-SBEN an error of 0.319 in a 10,000 iteration sweep. Online error was evaluated by computing classification error on the next 1,000 unseen samples generated by the CAPTCHA process.

While the simultaneous greedy training used in this experiment allows for construction of a deep hybrid model in unity when faced with a data stream, we note that

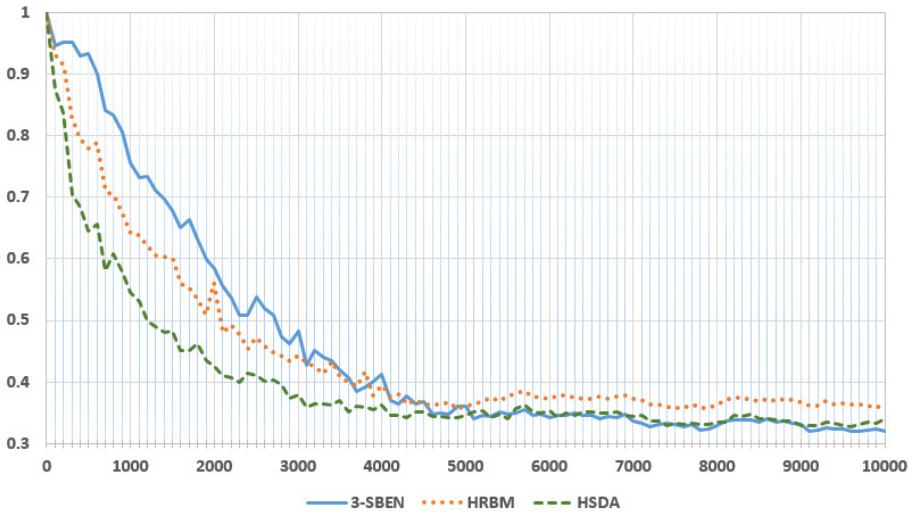

Fig. 2. Online error (y-axis) of 3-SBEN, 3-HSDA, & HRBM (or *1*- SBEN) evaluated every 100 time steps (x-axis). Each curve reported is a 4-trial mean of the lowest validation error model.

Algorithm 3 Online variant of layer-wise construction of a deep hybrid architecture.

Input: $(y_t, \mathbf{x}_t) \in \mathcal{D}_{train}$, $(\mathbf{u}_t) \in \mathcal{D}_{unlab}$, learning rate λ, hyper-parameters γ, α, β, & model parameters $\Theta = \{\Theta_1, \Theta_2, ..., \Theta_N\}$
function CONSTRUCTMODEL($y_t, \mathbf{x}_t, \mathbf{u}_t, \lambda, \gamma, \alpha, \beta, \Theta$)
  $\mathbf{x}_t^h \leftarrow \mathbf{x}_t$, $\mathbf{u}_t^h \leftarrow \mathbf{u}_t$ ▷ Initialize samples to low-level representations
  for $\Theta_n \in \Theta$ **do**
    $(\bigtriangledown_{disc}, \bigtriangledown_{gen}, \bigtriangledown_{unsup}) \leftarrow$ UPDATELAYER$(y_t, \mathbf{x}_t^h, \mathbf{u}_t^h, \Theta_n)$
    $\Theta_n \leftarrow \Theta_n + \lambda(-\gamma \bigtriangledown_{disc} + \alpha \bigtriangledown_{gen} + \beta \bigtriangledown_{unsup}), t \leftarrow t + 1$
    // Compute latent representation of data samples
    $(y_t, \mathbf{x}_t^h) \leftarrow$ COMPUTELATENTREPRESENTATION$(y_t, \mathbf{x}_t^h, \Theta_n)$
    $(\emptyset, \mathbf{u}_t^h) \leftarrow$ COMPUTELATENTREPRESENTATION$(\emptyset, \mathbf{u}_t^h, \Theta_n)$

instability may occur in the form of "shifting representations". This is where an upper level model is dynamically trained on a latent representation of a lower-level model that has not yet settled since it has not yet seen enough samples from the data distribution.

5 Conclusions

We developed two hybrid models, the SBEN and the HSDA, and their training algorithms in the context of incremental, semi-supervised learning. They combine efficient greedy, layer-wise construction of deeper architectures with a multi-objective learning approach. We balance the goal of learning a generative model of the data with extracting discriminative regularity to perform useful classification. More importantly, the framework we describe facilitates more explicit control over the multiple objectives

involved. Additionally, we presented a vertical aggregation scheme, *layer-wise ensembling*, for generating predictions that exploit discriminative knowledge acquired at all levels of abstraction defined by the architecture's hierarchical form. Our framework allows for explicit control over generative and discriminative objectives as well as a natural scheme for tracking layer-wise learning.

Models were evaluated in two problem settings: optical character recognition and text categorization. We compared results against shallow models and found that our hybrid architectures outperform the others in all datasets investigated. We found that the SBEN performed the best, improving classification error by as much 58% (compared to Maximum Entropy on WEBKB). Furthermore, we found that improvement in performance holds when hybrid learning is adapted to an online setting (relaxing the purely bottom-up framework in Section 3.1). We observe that we are able to improve error while significantly minimizing the number of required labeled samples (as low as 2% of total available data in some cases).

The hybrid deep architectures presented in this paper are not without potential limitations. First, there is the danger of "shifting representations" if using Algorithm 3 for online learning. To combat this, samples could be pooled into mini-batch matrices before computing gradients and minimize some of the noise of online error-surface descent. Alternatively, all layer-wise experts could be extended temporally to Conditional RBM-like structures, potentially improving performance as in [43]. Second, additional free parameters were introduced that require tuning, creating a more challenging model selection process for the human user. This may be alleviated with a parallelized, automated approach, however, a model that adapts its objective weights during the learning process would be better, altering its hyper-parameters in response to error progress on data subsets. Our frameworks may be augmented with automatic latent unit growth for both auto-encoder [42] and Boltzmann-like variants [10] or perhaps improved by "tying" all layer-wise expert outputs together in a scheme like that in [11].

The models presented in this paper offer promise in the goal of incrementally building powerful models that reduce expensive labeling and feature engineering effort. They represent a step towards ever-improving models that adapt to "in-the-wild" samples, capable of more fully embracing the "...unreasonable effectiveness of data" [16].

Acknowledgments The first author acknowledges support from Pennsylvania State University & the National Science Foundation under IGERT award # DGE-1144860, Big Data Social Science. We thank Hugo Larochelle and Roberto Calandra for their correspondence and advice, although all shortcomings of the presented work are ours.

References

1. Bengio, Y.: Deep learning of representations for unsupervised and transfer learning. Journal of Machine Learning Research–Proceedings Track (2012)
2. Bengio, Y., Courville, A.C., Vincent, P.: Unsupervised feature learning and deep learning: Review and new perspectives. CoRR abs/1206.5538 (2012)
3. Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H.: Greedy layer-wise training of deep networks. Advances in Neural Information Processing Systems (2007)

4. Bengio, Y., LeCun, Y.: Scaling learning algorithms towards AI. Large-Scale Kernel Machines 34, 1–41 (2007)
5. Calandra, R., Raiko, T., Deisenroth, M.P., Pouzols, F.M.: Learning Deep Belief Networks from non-stationary streams. In: Villa, A.E., Duch, W., rdi, P., Masulli, F., Palm, G. (eds.) Artificial Neural Networks and Machine Learning – ICANN 2012, pp. 379–386. No. 7553 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (2012)
6. Caragea, C., Wu, J., Williams, K., Das, S., Khabsa, M., Teregowda, P., Giles, C.L.: Automatic identification of research articles from crawled documents. Web-Scale Classification: Classifying Big Data from the Web, co-located with WSDM (2014)
7. Cardoso-Cachopo, A.: Improving methods for single-label text categorization. PdD Thesis, Instituto Superior Tecnico, Universidade Tecnica de Lisboa (2007)
8. Chen, G., Srihari, S.H.: Restricted boltzmann machine for classification with hierarchical correlated prior. arXiv preprint arXiv:1406.3407 (2014)
9. Crammer, K., Singer, Y.: On the algorithmic implementation of multiclass kernel-based vector machines. Journal of Machine Learning Research 2, 265–292 (2002)
10. Ct, M.A., Larochelle, H.: An infinite Restricted Boltzmann Machine. arXiv preprint arXiv:1502.02476 (2015)
11. Elfwing, S., Uchibe, E., Doya, K.: Expected energy-based restricted boltzmann machine for classification. Neural Networks 64, 29–38
12. Fiore, U., Palmieri, F., Castiglione, A., De Santis, A.: Network anomaly detection with the Restricted Boltzmann Machine. Neurocomputing 122 (2013)
13. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier networks. In: Proc. 14th International Conference on Artificial Intelligence and Statistics. vol. 15, pp. 315–323 (2011)
14. Glorot, X., Bordes, A., Bengio, Y.: Domain adaptation for large-scale sentiment classification: A deep learning approach. In: Proc. 28th International Conference on Machine Learning (ICML-11). pp. 513–520 (2011)
15. Gollapalli, S.D., Caragea, C., Mitra, P., Giles, C.L.: Researcher homepage classification using unlabeled data. In: Proc. 22nd International Conference on World Wide Web. pp. 471–482. Geneva, Switzerland (2013)
16. Halevy, A., Norvig, P., Pereira, F.: The unreasonable effectiveness of data. Intelligent Systems, IEEE 24(2), 8–12 (2009)
17. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Computation 14(8), 1771–1800 (2002)
18. Hinton, G.E.: What kind of graphical model is the brain? In: Proc. 19th International Joint Conference on Artificial Intelligence. vol. 5, pp. 1765–1775. Edinburgh, Scotland (2005)
19. Larochelle, H., Bengio, Y.: Classification using discriminative Restricted Boltzmann Machines. In: Proc. 25th International Conference on Machine learning. pp. 536–543 (2008)
20. Larochelle, H., Mandel, M., Pascanu, R., Bengio, Y.: Learning algorithms for the classification Restricted Boltzmann Machine. Journal of Machine Learning Research 13, 643–669 (2012)
21. Lasserre, J.A., Bishop, C.M., Minka, T.P.: Principled hybrids of generative and discriminative models. In: Proc. 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. vol. 1, pp. 87–94. IEEE Computer Society, Washington, DC, USA (2006)
22. Lee, C.Y., Xie, S., Gallagher, P., Zhang, Z., Tu, Z.: Deeply-supervised nets. arXiv:1409.5185 [cs, stat] (2014)
23. Lee, D.H.: Pseudo-label: The simple & efficient semi-supervised learning method for deep neural networks. In: Workshop on Challenges in Representation Learning, ICML (2013)
24. Liu, T.: A novel text classification approach based on deep belief network. In: Proc. 17th International Conference on Neural Information Processing: Theory and Algorithms - Volume Part I. pp. 314–321. Springer-Verlag (2010)

25. Louradour, J., Larochelle, H.: Classification of sets using Restricted Boltzmann Machines. arXiv preprint arXiv:1103.4896 (2011)
26. Lu, Z., Li, H.: A deep architecture for matching short texts. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 26, pp. 1367–1375. Curran Associates, Inc. (2013)
27. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: Proc. ICML. vol. 30 (2013)
28. Masud, M.M., Woolam, C., Gao, J., Khan, L., Han, J., Hamlen, K.W., Oza, N.C.: Facing the reality of data stream classification: Coping with scarcity of labeled data. Knowledge and Information Systems 33(1), 213–244 (2012)
29. Nair, V., Hinton, G.E.: Rectified linear units improve Restricted Boltzmann Machines. In: Proc. 27th International Conference on Machine Learning (ICML-10). pp. 807–814 (2010)
30. Ranzato, M.A., Szummer, M.: Semi-supervised learning of compact document representations with deep networks. In: Proc. 25th International Conference on Machine Learning. pp. 792–799. ACM (2008)
31. Salakhutdinov, R., Hinton, G.: Semantic Hashing. International Journal of Approximate Reasoning 50(7), 969–978 (2009)
32. Sarikaya, R., Hinton, G., Deoras, A.: Application of Deep Belief Networks for natural language understanding. IEEE/ACM Transactions on Audio, Speech, and Language Processing 22(4), 778–784 (2014)
33. Schapire, R.E.: The strength of weak learnability. Machine learning 5(2), 197–227 (1990)
34. Schmah, T., Hinton, G.E., Small, S.L., Strother, S., Zemel, R.S.: Generative versus discriminative training of RBMs for classification of fMRI images. In: Advances in neural information processing systems. pp. 1409–1416 (2008)
35. Shalev-Shwartz, S., Singer, Y., Srebro, N., Cotter, A.: Pegasos: Primal estimated sub-gradient solver for SVM. Mathematical programming 127(1), 3–30 (2011)
36. Sun, X., Li, C., Xu, W., Ren, F.: Chinese microblog sentiment classification based on deep belief nets with extended multi-modality features. In: 2014 IEEE International Conference on Data Mining Workshop (ICDMW). pp. 928–935 (2014)
37. Tomczak, J.M.: Prediction of breast cancer recurrence using classification Restricted Boltzmann Machine with dropping. arXiv preprint arXiv:1308.6324 (2013)
38. Tomczak, J.M., Ziba, M.: Classification restricted boltzmann machine for comprehensible credit scoring model. Expert Systems with Applications 42(4), 1789–1796 (2015)
39. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A.: Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of Machine Learning Research 11, 3371–3408 (2010)
40. Welling, M., Rosen-zvi, M., Hinton, G.E.: Exponential family harmoniums with an application to information retrieval. In: Saul, L.K., Weiss, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems 17, pp. 1481–1488. MIT Press (2005)
41. Zhang, J., Tian, G., Mu, Y., Fan, W.: Supervised deep learning with auxiliary networks. In: Proc. 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 353–361. ACM (2014)
42. Zhou, G., Sohn, K., Lee, H.: Online incremental feature learning with denoising autoencoders. In: Proc. 15th International Conference on Artificial Intelligence and Statistics. pp. 1453–1461 (2012)
43. Zhou, J., Luo, H., Luo, Q., Shen, L.: Attentiveness detection using continuous Restricted Boltzmann Machine in e-learning environment. In: Wang, F.L., Fong, J., Zhang, L., Lee, V.S.K. (eds.) Hybrid Learning and Education, pp. 24–34. No. 5685 in Lecture Notes in Computer Science, Springer Berlin Heidelberg (2009)