A Deep Learning Approach to Unsupervised Ensemble Learning

Uri Shaham URI.SHAHAM@YALE.EDU
Xiuyuan Cheng XIUYUAN.CHENG@YALE.EDU
Omer Dror OMERDR@GMAIL.COM
Ariel Jaffe ARIEL.JY@GMAIL.COM
Boaz Nadler BOAZ.NADLER@WEIZMANN.AC.IL
Joseph Chang JOSEPH.CHANG@YALE.EDU
Yuval Kluger YUVAL.KLUGER@YALE.EDU

Abstract

We show how deep learning methods can be applied in the context of crowdsourcing and unsupervised ensemble learning. First, we prove that the popular model of Dawid and Skene, which assumes that all classifiers are conditionally independent, is *equivalent* to a Restricted Boltzmann Machine (RBM) with a single hidden node. Hence, under this model, the posterior probabilities of the true labels can be instead estimated via a trained RBM. Next, to address the more general case, where classifiers may strongly violate the conditional independence assumption, we propose to apply RBM-based Deep Neural Net (DNN). Experimental results on various simulated and real-world datasets demonstrate that our proposed DNN approach outperforms other state-of-the-art methods, in particular when the data violates the conditional independence assumption.

1. Introduction

In recent years, crowdsourcing applications gained significant popularity, and consequently much academic attention. At the same time, deep learning has become a major tool in machine learning and artificial intelligence, demonstrating impressive performance in several applications, including computer vision, speech recognition and natural language processing.

The goal of this paper is to show that deep learning methods can also be applied to the areas of crowdsourcing and unsupervised ensemble learning, and provide state-of-the-art results. In unsupervised ensemble learning, one is given the predictions of d classifiers on a set of n instances and the goal is to recover the true, unknown label of each instance. Dawid & Skene (1979) were among the first to consider such a setup. They assumed that the classifiers are conditionally independent given the true labels. We refer to this model as the *DS* model and also as the *Conditional Independence model*.

Despite its simplicity, computing the maximum likelihood estimates of the classifiers' accuracies and the true labels in the DS model is a non-convex optimization problem. In their paper, Dawid and Skene estimated these quantities by the EM algorithm, which is only guaranteed to converge to a local optimum. In recent years, several authors developed computationally efficient spectral methods that are asymptotically consistent under the DS model, see Zhang et al. (2014); Parisi et al. (2014); Jain & Oh (2013); Jaffe et al. (2014) and references therein.

The model of Dawid and Skene relied on two key assumptions that typically do not hold in practice: (i) that classifiers make perfectly independent errors; and (ii) that these errors are uniformly distributed across all instances. To address the second issue above, several authors proposed richer models, that include parameters such as instance difficulty and varying skills of annotators across different regions of the input space, see for example Raykar et al. (2010), Whitehill et al. (2009) and Welinder et al. (2010).

In contrast, relatively few works considered relaxations of the conditional independence assumption: Platanios et al. (2014) proposed to estimate the accuracies of possibly dependent classifiers, via their agreement rates over classifier groups of different sizes. Donmez et al. (2010) proposed a model with pairwise interactions between all classifiers. Closest to our approach is the work of Jaffe et al. (2015), who assumed that some of the classifiers may be conditionally dependent, yet their dependency structure can be accurately described by a tree of depth 2.

In this manuscript, we propose a deep learning approach

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

to unsupervised ensemble learning problems with possibly dependent classifiers, where the conditional independence assumption is strongly violated. We make the following contributions. First, we show that the DS model has an equivalent parametrization in terms of a Restricted Boltzmann Machine (RBM) with a single hidden node. Hence, under this model, the posterior probability of the true labels can be estimated from a trained RBM. Next, to tackle violations of conditional independence, we show how a RBM-based Deep Neural Net (DNN) can be applied to unsupervised ensemble learning, and propose a heuristic for determining the DNN architecture. Experimentally, we compare our approach to several state-of-the-art methods that are based on the conditional independence assumption and relaxations of it. We show that our DNN approach often performs better than the other methods on both simulated and real world datasets. Remarkably, we demonstrate that in some cases, while the raw representation of the data contains correlated features, the learned features in the last hidden layer are almost perfectly uncorrelated.

The structure of this manuscript is as follows: in Section 2 we give a formal definition of the problem. A brief background on RBMs is given in Section 3. In Section 4 we show how RBMs can be used to predict the true labels, under the assumption of conditional independence. In Section 5 we describe how to estimate the labels using a RBM-based DNN. Experimental results are reported in Section 6. The manuscript concludes with a brief summary in Section 7. Proofs appear in the appendix.

1.1. Notation

Throughout this manuscript, X, H, Y are random variables, p_θ, p_λ are probability densities, parametrized by θ, λ, respectively. We think of p_θ as the distribution generating the data and of p_λ as the RBM model distribution. When the context is clear, we occasionally write $p(x)$ as a shorthand for $p(X = x)$. The dimensions of the input data and the sample size are denoted by d and n, respectively. We use $\sigma(\cdot)$ to denote the sigmoid function $\sigma(z) = \frac{1}{1+e^{-z}}$.

2. Problem Setup

Let $X \in \{0,1\}^d$, $Y \in \{0,1\}$ be random variables. We refer to Y as the label of X. The pair (X, Y) has a joint distribution, parametrized by θ which is given by $p_\theta(X, Y) = p_\theta(Y)p_\theta(X|Y)$. The joint distribution $p_\theta(X, Y)$ is not known to us, and neither are the marginals $p_\theta(X), p_\theta(Y)$. Let $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})$ be n i.i.d samples from $p_\theta(X, Y)$. In unsupervised ensemble learning, we observe $x^{(1)}, \ldots, x^{(n)}$ and the learning task is to recover $y^{(1)}, \ldots, y^{(n)}$. In this application, the binary vector $X = (X_1, \ldots, X_d)^T$ contains the predictions of d classifiers or annotators on an instance, whose label Y is unobserved.

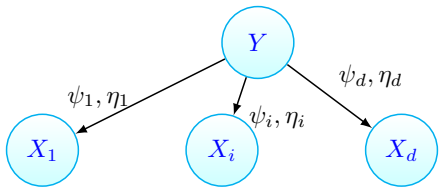

Figure 1. The conditional independence model, studied by Dawid & Skene (1979).

2.1. The Conditional Independence Model

In their seminal paper, Dawid & Skene (1979), assumed that the conditional distribution $p_\theta(X|Y)$ factorizes, i.e.,

$$p_\theta(X|Y) \equiv \prod_{i=1}^{d} p_\theta(X_i|Y). \tag{1}$$

Eq. (1), also known as the *conditional independence model*, is depicted in Figure 1. It is fully parametrized by $\theta = (\{\psi_i : i = 1, ..., d\}, \{\eta_i : i = 1, ..., d\}, \pi)$, where

$$\psi_i = \Pr(X_i = 1|Y = 1),\ \eta_i = \Pr(X_i = 0|Y = 0),$$
$$\pi = \Pr(Y = 1).$$

ψ_i, η_i are often referred to as sensitivity and specificity, respectively. Under the interpretation of the X_i's being classifiers, the sensitivity and specificity quantify the competence of the classifiers or annotators and the conditional independence assumption means that all d classifiers make independent errors.

The conditional independence model is often overly simplistic. In this manuscript we propose to apply deep learning techniques, specifically RBM-based DNNs, for unsupervised ensemble learning problems, where the conditional independence is not likely to hold. The following section gives essential background on RBMs, section 4 shows that a RBM with a single hidden node is equivalent to the conditional independence model, and section 5 presents our RBM-based DNN approach.

3. Restricted Boltzmann Machines

A Restricted Boltzmann Machine (RBM) is an undirected bipartite graphical model, consisting of a set X of d visible binary random variables and a set H of m hidden binary random variables, arranged in two layers, which are fully connected to each other. An illustration of a RBM is depicted in Figure 2. A RBM is parametrized by $\lambda = (W, a, b)$, where W is the weight matrix of the connections between the visible and hidden units, and a, b are the bias vectors of the visible and hidden layers, respectively. Each configuration $(X = x, H = h)$ of a RBM is

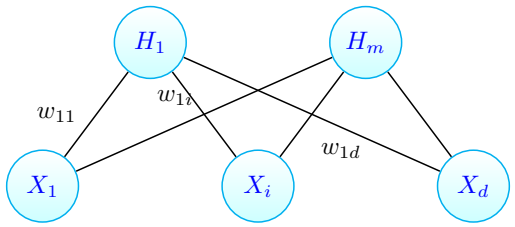

Figure 2. A RBM with d visible and m hidden units.

associated with the following energy

$$E_\lambda(x,h) = -(a^T x + b^T h + x^T W h) \tag{2}$$

which defines the probability of the configuration

$$p_\lambda(X = x, H = h) = \frac{e^{-E_\lambda(x,h)}}{Z},$$

where $Z \equiv \sum_{x,h} e^{-E_\lambda(x,h)}$ is the *partition function*. The bipartite structure of the RBM implies factorial conditional probabilities

$$p_\lambda(X|H) = \prod_i p_\lambda(X_i|H),\ p_\lambda(H|X) = \prod_j p_\lambda(H_j|X),$$

given by

$$p_\lambda(X_i = 1|H) = \sigma(a_i + W_{i.}H)$$
$$p_\lambda(H_j = 1|X) = \sigma(b_j + X^T W_{.j}),$$

where $\sigma(z)$ is the sigmoid function, $W_{i.}$ is the i-th row of W and $W_{.j}$ is its j-th column.

Given iid training data $x^{(1)}, .., x^{(n)} \sim p_\theta(X)$, the RBM parameters $\lambda = (W, a, b)$ are typically tuned to maximize the log-likelihood of the training data, where the likelihood that the RBM associates with a vector x is given by $p_\lambda(X = x) = \sum_h p_\lambda(X = x, H = h)$.

A popular approach to learn the RBM parameters is via gradient-based optimization, where the gradients are approximated using contrastive divergence (Hinton et al., 2006; Bengio, 2009).

4. RBM in the Conditional Independence Case

In this section we show that given observed data $x^{(1)}, \ldots, x^{(n)} \in \{0,1\}^d$ from the conditional independence model of Eq. (1), the posterior probabilities of the true, unknown labels $y^{(1)}, \ldots, y^{(n)}$ can be consistently estimated via a RBM with a single hidden node.

We begin by showing that there is a bijective map from the parameters λ of a RBM with a single hidden node to the parameters θ of the conditional independence model, such that the joint distribution specified by the RBM is *equivalent* to that of the conditional independence model.

Lemma 4.1. *The joint probability $p_\lambda(X = x, H = y)$ of a RBM with parameters $\lambda = (a, b, W)$ is equivalent to the joint probability $p_\theta(X = x, Y = y)$ of a conditional independence model with parameters $\theta = (\{\psi_i\}, \{\eta_i\}, \pi)$ given by*

$$\psi_i \equiv \sigma(a_i + W_i),\ \eta_i \equiv 1 - \sigma(a_i)$$
$$\pi \equiv \frac{\sum_{x\in\{0,1\}^d} e^{a^T x + b + x^T W}}{\sum_{x\in\{0,1\}^d} \left(e^{a^T x} + e^{a^T x + b + x^T W}\right)}$$

Furthermore, the map $\lambda \mapsto \theta$ is a bijection.

We are now ready to prove the main result of this section, namely, that the posterior distribution of the true labels $y^{(1)}, \ldots, y^{(n)}$ can be consistently estimated by a RBM with a single hidden node. To do so, we rely on a special case of a result proved by Chang (1996), that provides conditions under which the parameters of the conditional independence model are identifiable.

Lemma 4.2. *Let $x^{(1)}, ..., x^{(n)}$ be observed data from the conditional independence model, specified by p_θ. Assume that θ is such that for each $i = 1, \ldots, d$, X_i is not independent of Y (i.e., each classifier is not just a random guess), and that $d \geq 3$. Let $\hat{\lambda}_{MLE}$ be a maximum likelihood parameter estimate of a RBM with a single hidden node. Then the RBM posterior probability $p_{\hat{\lambda}_{MLE}}(H = 1|X = x)$ converges to the true posterior $p_\theta(Y = 1|X = x)$, as $n \to \infty$.*

Remark 4.3. *The identifiability of the parameters is up to a single global $0/1$ label flip. This means that one recovers either $p_\theta(Y = y|X)$ or $p_\theta(Y = 1 - y|X)$. Assuming that on average, the X_i's are more accurate than a random guess, this sign ambiguity can be resolved by comparing the predictions to the majority vote decision.*

Remark 4.4. *Lemma 4.2 assumes that we found the MLE of the RBM parameters. Obtaining such a MLE is problematic for two main reasons. First, RBMs are typically trained to maximize a proxy for the likelihood, as the true likelihood is not tractable. Second, the RBM likelihood function is not concave, hence there are no guarantees that after training a RBM one obtains the maximum likelihood parameter $\hat{\lambda}_{MLE}$.*

5. RBM-based Deep Neural Net

In many practical settings, the variables $X_1, \ldots, X_d$ are not conditionally independent. Fitting a conditionally independent model to such data may yield highly sub-optimal predictions for the true labels y_i. To tackle this general case, we propose to train a RBM-based Deep Neural Net (DNN) and use it to estimate the posterior probabilities $p_\theta(Y|X)$. In such a DNN, the hidden layer of each RBM is the input for the successive RBM. As suggested by Hinton et al. (2006), the RBMs are trained one at a time, bottom to top,

i.e., the DNN is trained in a layer-wise fashion. Specifically, given training data $x^{(1)}, \ldots, x^{(n)} \in \{0,1\}^d$, we start by training the bottom RBM, and then obtain the first layer hidden representation of the data by sampling $h^{(i)}$ from the conditional RBM distribution $p_\lambda(H|X = x^{(i)})$. The vectors $h^{(1)}, \ldots, h^{(n)}$ are then used as a training set for the second RBM and so on.

In the case considered in this manuscript, where the true label y is binary, the upper-most RBM in the DNN has a single hidden unit, from which the posterior probability $p_\theta(Y|X)$ can be estimated. Such a DNN is depicted in Figure 3.

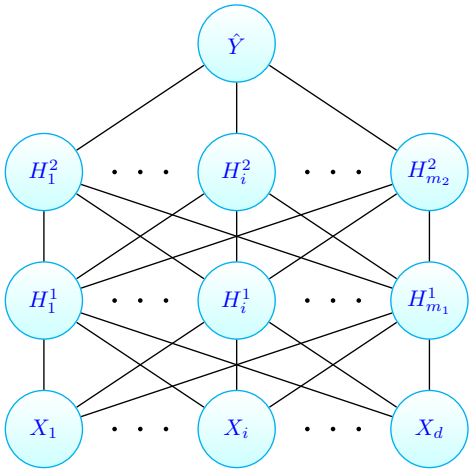

Figure 3. A sketch of RBM-based DNN with two hidden layers.

5.1. Motivation

Deep learning algorithms have recently achieved state-of-the-art performance in a wide range of applications (LeCun et al., 2015). While a rigorous theoretical understanding of deep nets is still lacking, many researchers believe that a key property in their success is their ability to disentangle factors of variation in the inputs; see for example Bengio et al. (2013), Tishby & Zaslavsky (2015), and Mehta & Schwab (2014). That is, as one moves through the net, the hidden units become less statistically dependent. We have seen in Section 4 that given a representation in which the units are independent conditional on the true label, a single node RBM gives a consistent estimation of the true label posterior probability. Propagating the data through several RBM layers can hence be seen as a processing of the data, which reduces the conditional dependence of the units while preserving most of the information on the true label Y. In Section 6 we will demonstrate cases where such decoupling does indeed happen in practice, i.e., although the original input variables X_i's are not conditionally independent given the true label Y, after training, the units in the uppermost hidden layer are, remarkably, approximately conditionally independent. Thus, the assumptions of the conditional independence model apply (with respect to the uppermost hidden layer H^{last}), and therefore one is able to consistently estimate the label posterior probability, $\Pr(Y|H^{\text{last}})$, as in Section 4.

Another motivation for using deep nets with several hidden layers for unsupervised ensemble learning is their rich expressive power. In our setting, we wish to approximate the posterior probability $p(Y|X)$, which in general may be a complicated nonlinear function of X. When $p(Y|X)$ cannot be accurately estimated by a RBM with a single hidden node (i.e., when the conditional independence assumption of Dawid and Skene does not hold), a better approximation may be obtained from a deeper network. Several works show that there exist functions that are significantly more efficiently represented by deeper networks, compared to shallower ones, where efficiency corresponds to the number of units. For example, Montufar et al. (2014) show that deep networks with piece-wise linear activations can represent functions with greater number of linear regions compared to shallow networks with the same number of units. In a recent work, Eldan & Shamir (2015) give an example for a radial function that can be efficiently computed by a 3-layer network, while requiring exponentially many units to be approximated accurately by a 2-layer network.

Finally, we would like to emphasize that a RBM-based DNN is a discriminative model to estimate the posterior $p(Y|X)$. In general, it may not correspond to any generative model (Arora et al., 2015). Indeed, there is no guarantee that the marginal distributions implied by two adjacent RBMs match. Yet, it can be shown (see Appendix C) that stacking RBMs is a variational inference procedure assuming a specific class of data generation models. The nature of approximation of a top down generative model, where the data X is generated from a label Y, by a RBM-based DNN is explored in Appendix D.

5.2. Predicting the Label from a Trained DNN

Given a trained DNN and a sample $x \sim p_\theta(X)$, the label y is estimated by propagating x through the network. Specifically, the units of each layer can be set by either (i) sampling from the conditional distribution given the layer below, i.e., $h_j \sim p_\lambda(h_j|x)$, or (ii) by MAP estimate, setting each hidden unit $h_j = \arg\max_{h_j \in \{0,1\}} p_\lambda(h_j|x)$. Since the first option is stochastic, one may propagate x through the net multiple times and average the outputs $p(y|x)$ to obtain an approximation of $\mathbb{E}(Y|X = x)$. Experimentally, we found both options to be equally effective, while each option slightly outperforms the other in some cases.

5.3. Choosing the DNN Architecture

The specific DNN architecture (i.e., number and sizes of layers) might have a dramatic effect on the quality of pre-

dictions. To determine the number of units in each layer we employed the following procedure: we first train a RBM with d hidden units. Next, we compute the singular value decomposition of the weight matrix W, and determine its rank (i.e., the number of sufficiently large singular values). Given that the rank is some $m \leq d$, we re-train the RBM, setting the number of hidden units to be m. If $m > 1$, we add another layer on top of the current layer, and proceed recursively. The process stops when $m = 1$, so that the last layer of the DNN contains a single node. We refer to this method as *the SVD approach*. In our experiments, as a rule of thumb, we set m to be the minimal number of singular values (in descending order) whose cumulative sum is at least 95% of the total sum.

This method takes advantage of the co-adaptation of hidden units, which is a well known phenomenon in RBM training (see, for example, (Hinton et al., 2012)). The term *co-adaptation* describes a situation where several hidden units tend to behave very similarly; this implies that the rank of the weight matrix might be small, although the number of hidden units may be larger. The idea to use SVD to set the number of hidden units in neural networks has been proposed before, see (Teoh et al., 2006).

6. Experimental Results

In this section we compare the performance of the proposed DNN approach to several other approaches, and report experimental results obtained on four simulated data sets as well as real world data sets from two different domains. Our datasets, as well as the scripts used to obtain the reported results are publicly available at `https://github.com/ushaham/RBMpaper` [1].

Specifically, we compare between the following unsupervised ensemble methods:

- **Vote.** Majority voting, which is the maximum likelihood prediction, assuming that all classifiers are conditionally independent and have the same accuracy.
- **DS.** Approximate maximum likelihood predictions under the Dawid and Skene model. Specifically, we use Spectral Meta Learner (Parisi et al., 2014), and Restricted Likelihood (Jaffe et al., 2014).
- **CUBAM** The method of Welinder et al. (2010), which assumes conditional independence, but allows the accuracy of each classifier to vary across different regions of the input domain.
- **L-SML** Latent SML (Jaffe et al., 2015). This method relaxes the conditional independence assumption to a depth 2 tree model.
- **DNN** The approach presented in this manuscript, with the depth and number of hidden units in each layer determined by the SVD approach, described in Section 5.3.

Following Jaffe et al. (2015), the performance measure we chose is the balanced accuracy, given by

$$\frac{\sum \mathbb{I}\{\text{true label is 0 and predicted label is 0}\}}{2\sum \mathbb{I}\{\text{true label is 0}\}} + \frac{\sum \mathbb{I}\{\text{true label is 1 and predicted label is 1}\}}{2\sum \mathbb{I}\{\text{true label is 1}\}},$$

where $\mathbb{I}\{\cdot\}$ is the indicator function.

6.1. Simulated Datasets

In this experiment we carefully generated four synthetic datasets, in order to demonstrate the performance of the DNN approach in several specific scenarios. In all four datasets the observed data is a $n \times d$ binary matrix, with input dimension $d = 15$ and sample size $n = 10,000$. A detailed description of the datasets generation process is given in Appendix E.1.

- **CondInd** A dataset where the conditional independence holds, and 10 of the 15 classifiers are in fact random guess.
- **Tree15-3-1** A dataset generated from a depth-2 tree with layer sizes 1,3,15. Every node in the intermediate layer is connected to five nodes in the bottom layer. This dataset is generated from the model considered by L-SML, and does not satisfy the conditional independence assumption, as is shown in Figure 6.
- **LayeredGraph15-5-5-1** A dataset generated from a depth-3 layered graph, with layer sizes 1,5,5,15. In this case, the conditional independence assumption does not hold, although in practice the amount of dependence in the data is not high (see Figure 11).
- **TruncatedGaussian.** Here $X = (1 + \text{sign}(Z))/2$, where the random variable Z follows a a mixture of two d-dimensional Gaussians with different means and same covariance matrix. The label Y indicates the specific Gaussian from which X is sampled. In this case, the data is highly dependent, as can be seen in Figure 11.

The results are summarized in Table 1. Along with the five unsupervised methods, the table also shows the accuracy of a supervised learner and the estimated accuracy of the

[1] Our scripts are based on the publicly available code in Geoffrey Hinton's website `http://www.cs.toronto.edu/~hinton/MatlabForSciencePaper.html`.

Table 1. Balanced accuracy of various unsupervised ensemble methods on the four synthetic datasets, along with a supervised learner (SUP), and the Bayes optimal classifier (Bayes-Opt). The results are presented as mean ± standard deviation, based on 5 repetitions, where in each repetition a new dataset was sampled from the model. The numbers in brackets denote the architecture of the DNN, found by the SVD approach.

method	condInd	Tree15-3-1	LG15-5-5-1	TG
Vote	75.93 ± 0.5	93.45 ± 0.19	76.61 ± 0.09	80.14 ± 0.4
DS	**94.78 ± 0.13**	92.68 ± 0.14	86.36 ± 0.2	82.03 ± 0.27
CUBAM	91.96 ± 0.18	90.74 ± 0.3	77.12 ± 0.26	83.43 ± 0.31
L-SML	55.94 ± 21.88	**95.83 ± 0.15**	85.87 ± 0.21	79.5 ± 1.35
DNN	**94.78 ± 0.13** (15-1)	95.13 ± 0.71 (15-3-1)	**86.83 ± 0.2** (15-4-1)	**88.09 ± 0.52** (15-3-1)
SUP	94.45 ± 0.11	95.54 ± 0.27	87.01 ± 0.18	90.8 ± 0.4
Bayes-Opt	95.32	96.12	87.05	91.39

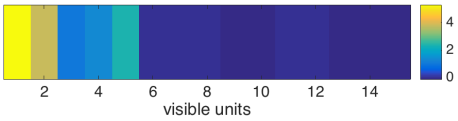

Figure 4. The RBM weight vector on the condInd dataset. The hidden unit is strongly connected only to the first five visible units, reflecting the fact that in an unsupervised manner, the RBM detected that the remaining units are random guess classifiers.

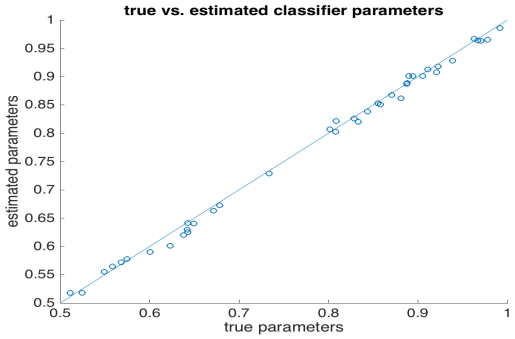

Figure 5. Recovery of the conditional Independence model parameters $\{\psi_i, \eta_i\}$ from a RBM with a single hidden node, on a dataset sampled from a conditional independence model. The parameters were uniformly sampled from $[0.5, 1]$. Each circle corresponds to a single parameter (e.g., ψ_i for some i). For convenience, the identity line was added to the plot.

Bayes-optimal classifier. The supervised learner is a Multi Layer Perceptron (MLP) with two hidden layers of sizes 4 and 2, that was trained on a dataset with $n = 10,000$ samples (independent of the test dataset). The Bayes-optimal approximated accuracy was computed on a sample of size $10,000$, with the true posterior probabilities of all 2^d possible binary vectors estimated using a sample of size 10^6 from the corresponding model.

On all of the above datasets, the DNN always outperformed the majority vote rule and CUBAM. On the CondInd dataset, the DNN performs similarly to DS, and significantly better than the other methods. Despite being unsupervised, on this dataset both methods perform slightly better than the specific supervised learner we considered, and around the Bayes-optimal accuracy. The architecture determined by the SVD approach in this case is indeed a single RBM (with a single hidden node). The weight matrix of the RBM is shown in Figure 4, and corresponds to the fact that only the first five classifiers actually contain information about the true label in this dataset.

Figure 5 shows the recovery of the true conditional independence model parameters $\{\psi_i, \eta_i\}$ of a similar conditional independent dataset (however with no random guess classifiers) from a RBM with a single hidden node, using the map in Lemma 4.1.

On the Tree15-3-1 dataset, L-SML, which is tailored for data generated by a tree, outperforms the DNN. This result is expected, since it can be shown that the distribution of the bottom two layers of a tree cannot be parametrized as a RBM (see Appendix D). Still, the DNN performs significantly better than DS, CUBAM and majority vote, and not far from the supervised learner and the optimal Bayes classifier. Figure 6 shows the correlation matrix at the input and hidden layers, as well as the first layer weight matrix, demonstrating that the DNN captured the true data generation model. Consequently, the 3 hidden units are nearly conditionally uncorrelated given the label y.

Figure 7 shows the cumulative proportion of the singular values on the condInd and Tree15-3-1 datasets, which explains the architecture determined by the SVD approach for both datasets.

On the LayeredGraph15-5-5-1 dataset, while outperforming the other methods, the DNN achieved accuracy close to the supervised learner and the Bayes optimal accuracy; however, the chosen DNN architecture is different from the one of the true data generation model.

The conditional independence assumption is strongly vio-

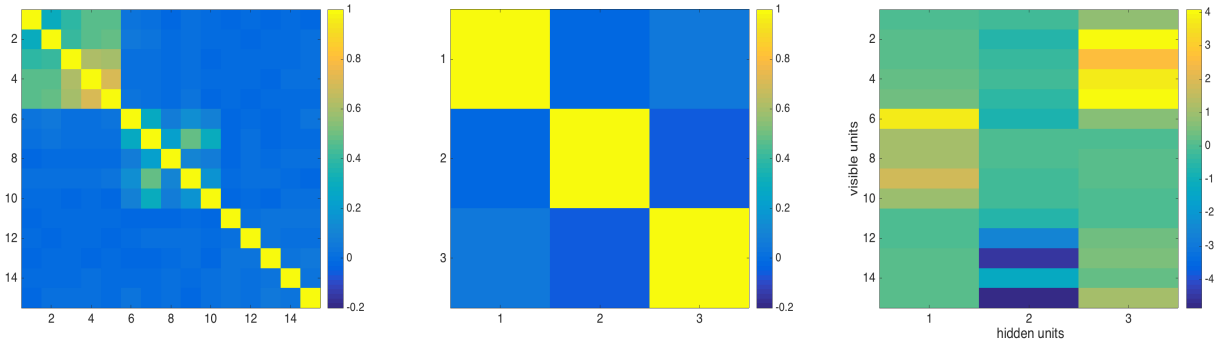

Figure 6. The Tree15-3-1 experiment. Left: correlation matrix of the input data for the $y = 0$ class. The first and middle five $X_i's$ are not conditionally independent of each other. Center: correlation matrix of the hidden layer of the DNN for the $y = 0$ class. The hidden units are approximately uncorrelated. Right: weight matrix of the bottom RBM of the DNN, showing that each hidden unit is strongly connected to 5 visible units, as in the original data generation model.

Table 2. Balanced accuracy of various methods on the DREAM datasets S1, S2 and S3. DNN results are averaged over 5 repetitions, and are presented as mean ± standard deviation. The numbers in brackets denotes the architecture of the DNN, found by the SVD approach. * results reported in (Jaffe et al., 2015)

Dataset	Vote	DS	CUBAM	L-SML	DNN
S1	97.2 *	98.3 *	92.31	**98.4** *	**98.42 ± 0.0** (124-1)
S2	96 *	97.2 *	69.19	**97.7** *	97.55 ± 0.01 (114-1)
S3	95.7 *	97.7 *	87.65	98.2 *	**98.51 ± 0.01** (99-25-1)

lated in the case of the TruncatedGaussian dataset. Here the DNN performs better than all other methods by a large margin.

6.2. Real-World Datasets

In this section we experiment with two groups of datasets, from two different domains, as follows:

DREAM Three datasets from the DREAM mutation calling challenge (Ewing et al., 2015); this challenge is an international effort to improve standard methods for identifying cancer-associated mutations and rearrangements in whole-genome sequencing data. The accuracy of current variant calling algorithms is not optimal due to sequencing errors, other experimental factors, parametric choices in each algorithm and preprocessing and filtering decisions. Unsupervised ensemble learning of multiple variant callers is expected to provide more robust predictions. One of the goals of this challenge is to develop a state-of-the-art meta pipeline for somatic mutation detection, to output accurate as possible mutation calls associated with cancer. Specifically, we used three datasets, (S1, S2, S3) containing the predictions of classifiers that determine the presence or absence of of mutations in genome sequencing data. The data is available at (Ellrot, 2013). In S1, $d = 124$, $n = 92,362$. In S2, d = 114, n = 70,561. In S3, $d = 99$, $n = 78,643$.

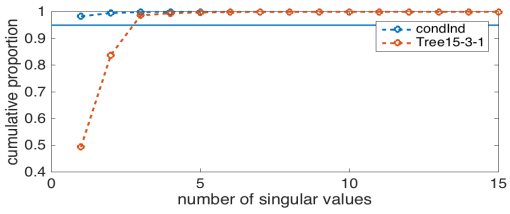

Figure 7. Cumulative proportion of singular values on the condInd and Tree15-3-1 datasets. While in the condInd case the first singular value is more than 95% of the total sum of singular values, the first three singular values are needed on the Tree15-3-1 dataset. The horizontal line at 0.95 is added to the plot for convenience.

Magic Forty datasets, which are constructed from the Magic dataset in the UCI repository[2]. This dataset contains $n = 19,020$ instances with 11 attributes, which consists of physical measurements of gamma particles; the learning task is to classify each instance as background or high energy gamma rays. Each of the Forty datasets we constructed contains binary predictions of $d = 16$ classifiers, obtained in the Weka machine learning software. The 16 classifiers belong to four groups: four random forest classifiers, three logistic trees classifiers, four SVM classifiers, and five naive Bayes classifiers. This setting is adopted from (Jaffe et al., 2015). The group of SVM classifiers is highly correlated, as well as the group of Naive Bayes classifiers, as can be seen in Appendix E.2. Each of the forty datasets was obtained by predictions of the same classifiers, however trained on a different subset of the original Magic dataset (a random subset of size 500 each time).

Table 2 shows the performance of the various methods on

[2] https://archive.ics.uci.edu/ml/datasets/MAGIC+Gamma+Telescope

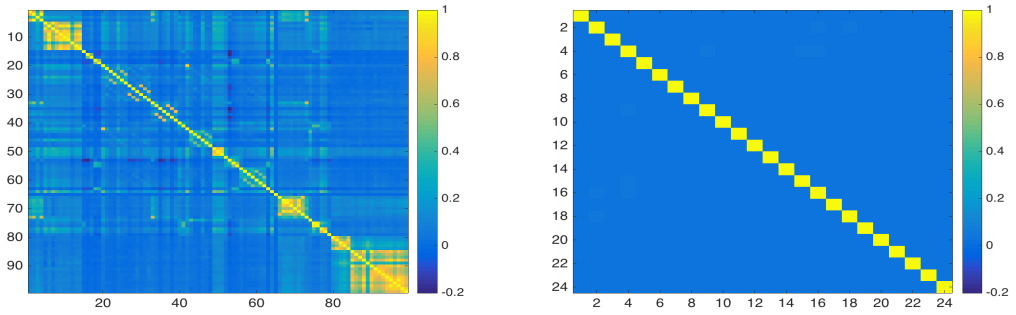

Figure 8. correlation matrices of the input (left) and hidden (right) layers of the DNN on the S3 dataset, for the $y = 0$ class. Remarkably, the hidden units are almost perfectly uncorrelated, conditioned on the class.

the DREAM datasets. As can be seen, the DNN and L-SML perform similarly on S1, while the former performs better on S3 and the latter on S2. The two methods outperform the majority vote rule, DS and CUBAM on all three datasets. Remarkably, the hidden representation on the S3 dataset is such that the units are perfectly uncorrelated, conditioned on the hidden label. This is shown in Figure 8.

The results on the Magic datasets are shown in Figure 9. On most of these datasets, the DNN outperforms all other methods, with a relatively large margin. On all forty datasets, the SVD approach yielded a 15-3-1 architecture. We also used paired t-test to compare the DNN result to each of the other methods. The null hypothesis of this test is that the means of the performance of each method are equal. In all four tests the null hypothesis was rejected with p-value $\leq 10^{-13}$.

To summarize our experiments, we observed that RBM-based DNN performs at least as well and often better than various other methods, on both simulated and real datasets, and that the SVD approach can serve as an effective tool for determination of the DNN architecture.

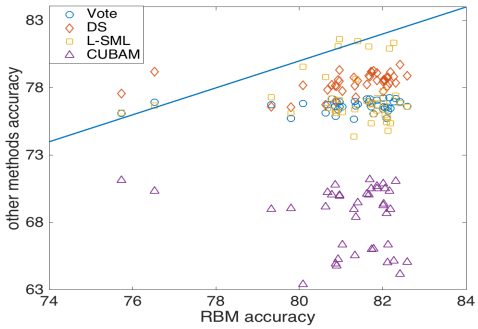

Figure 9. Performance of the various methods on the Magic datasets. For convenience, the identity line is added to the plot. Most of the points are below the identity line, which indicates that the DNN tend to outperform all other methods on these datasets.

We remark that in our experiments, we observed that RBMs tend to be highly sensitive to hyper-parameter tuning (such as learning rate, momentum, regularization type and penalty), and these hyper-parameters need to be carefully tuned. To obtain a reasonable hyper-parameter setting we found it useful to apply the random configuration sampling procedure, proposed in (Bergstra & Bengio, 2012), and evaluate different models by average log-likelihood approximation, (see, for example, (Salakhutdinov & Murray, 2008) and the corresponding MATLAB scripts in (Salakhutdinov, 2010)).

7. Summary and Discussion

We demonstrated how deep learning techniques can be used for unsupervised ensemble learning, and showed that the DNN approach proposed in this manuscript often performs at least as well and often better than state-of the art methods, especially when the conditional independence assumption made by Dawid & Skene (1979) does not hold.

Possible directions for future research include extending the approach to multiclass problems, possible using Discrete RBMs (Montúfar & Morton, 2013), theoretical analysis of the SVD approach, and information theoretic analysis of the de-correlation, while preserving label information, that occurs while propagating data through a RBM-based DNN.

Acknowledgements

The authors thank George Linderman, Alex Cloninger, Tingting Jiang, Raphy Coifman, Sahand Negahban, Andrew Barron, Alex Kovner, Shahar Kovalsky, Maria Angelica Cueto, Jason Morton, and Bernd Sturmfels for their help. This research was funded by the Intel Collaborative Research Institute for Computational Intelligence (B.N.) and by NIH grant 1R01HG008383-01A1 (Y.K. and B.N.).

References

Arora, Sanjeev, Liang, Yingyu, and Ma, Tengyu. Why are deep nets reversible: A simple theory, with implications for training. *arXiv preprint arXiv:1511.05653*, 2015.

Bengio, Yoshua. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

Bengio, Yoshua, Courville, Aaron, and Vincent, Pierre. Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1798–1828, 2013.

Bergstra, James and Bengio, Yoshua. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, 13(1):281–305, 2012.

Bishop, Christopher M. *Pattern recognition and machine learning*. springer, 2006.

Blei, David M, Kucukelbir, Alp, and McAuliffe, Jon D. Variational inference: A review for statisticians. *arXiv preprint arXiv:1601.00670*, 2016.

Casella, George and Berger, Roger L. *Statistical inference*, volume 2. Duxbury Pacific Grove, CA, 2002.

Chang, Joseph T. Full reconstruction of markov models on evolutionary trees: identifiability and consistency. *Mathematical biosciences*, 137(1):51–73, 1996.

Cueto, María Angélica, Morton, Jason, and Sturmfels, Bernd. Geometry of the restricted boltzmann machine. *Algebraic Methods in Statistics and Probability,(eds. M. Viana and H. Wynn), AMS, Contemporary Mathematics*, 516:135–153, 2010.

Dawid, Alexander Philip and Skene, Allan M. Maximum likelihood estimation of observer error-rates using the em algorithm. *Applied statistics*, pp. 20–28, 1979.

Donmez, Pinar, Lebanon, Guy, and Balasubramanian, Krishnakumar. Unsupervised supervised learning i: Estimating classification and regression errors without labels. *The Journal of Machine Learning Research*, 11: 1323–1351, 2010.

Eldan, Ronen and Shamir, Ohad. The power of depth for feedforward neural networks. *arXiv preprint arXiv:1512.03965*, 2015.

Ellrot, Kyle. Icgc-tcga dream mutation calling challenge. https://www.synapse.org/#!Synapse:syn312572/wiki/58893, 2013. Online; accessed 12-November-2015.

Ewing, Adam D, Houlahan, Kathleen E, Hu, Yin, Ellrott, Kyle, Caloian, Cristian, Yamaguchi, Takafumi N, Bare, J Christopher, P'ng, Christine, Waggott, Daryl, Sabelnykova, Veronica Y, et al. Combining tumor genome simulation with crowdsourcing to benchmark somatic single-nucleotide-variant detection. *Nature methods*, 2015.

Fox, Charles W and Roberts, Stephen J. A tutorial on variational bayesian inference. *Artificial intelligence review*, 38(2):85–95, 2012.

Hinton, Geoffrey E, Osindero, Simon, and Teh, Yee-Whye. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

Hinton, Geoffrey E, Srivastava, Nitish, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdinov, Ruslan R. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

Jaffe, Ariel, Nadler, Boaz, and Kluger, Yuval. Estimating the accuracies of multiple classifiers without labeled data. *arXiv preprint arXiv:1407.7644*, 2014.

Jaffe, Ariel, Fetaya, Ethan, Nadler, Boaz, Jiang, Tingting, and Kluger, Yuval. Unsupervised ensemble learning with dependent classifiers. *arXiv preprint arXiv:1510.05830*, 2015.

Jain, Prateek and Oh, Sewoong. Learning mixtures of discrete product distributions using spectral decompositions. *arXiv preprint arXiv:1311.2972*, 2013.

LeCun, Yann, Bengio, Yoshua, and Hinton, Geoffrey. Deep learning. *Nature*, 521(7553):436–444, 2015.

Mehta, Pankaj and Schwab, David J. An exact mapping between the variational renormalization group and deep learning. *arXiv preprint arXiv:1410.3831*, 2014.

Montúfar, Guido and Morton, Jason. Discrete restricted boltzmann machines. *arXiv preprint arXiv:1301.3529*, 2013.

Montufar, Guido F, Pascanu, Razvan, Cho, Kyunghyun, and Bengio, Yoshua. On the number of linear regions of deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 2924–2932, 2014.

Parisi, Fabio, Strino, Francesco, Nadler, Boaz, and Kluger, Yuval. Ranking and combining multiple predictors without labeled data. *Proceedings of the National Academy of Sciences*, 111(4):1253–1258, 2014.

Platanios, A, Blum, Avrim, and Mitchell, Tom M. Estimating accuracy from unlabeled data. In *In Proceedings of UAI*, 2014.

Raykar, Vikas C, Yu, Shipeng, Zhao, Linda H, Valadez, Gerardo Hermosillo, Florin, Charles, Bogoni, Luca, and Moy, Linda. Learning from crowds. *The Journal of Machine Learning Research*, 11:1297–1322, 2010.

Salakhutdinov, Ruslan. Ruslan salakhutdinov's web page, 2010. URL http://www.cs.toronto.edu/~rsalakhu/code.html.

Salakhutdinov, Ruslan and Murray, Iain. On the quantitative analysis of deep belief networks. In *Proceedings of the 25th international conference on Machine learning*, pp. 872–879. ACM, 2008.

Teoh, Eu Jin, Tan, Kay Chen, and Xiang, Cheng. Estimating the number of hidden neurons in a feedforward network using the singular value decomposition. *Neural Networks, IEEE Transactions on*, 17(6):1623–1629, 2006.

Tishby, Naftali and Zaslavsky, Noga. Deep learning and the information bottleneck principle. *arXiv preprint arXiv:1503.02406*, 2015.

Welinder, Peter, Branson, Steve, Perona, Pietro, and Belongie, Serge J. The multidimensional wisdom of crowds. In *Advances in neural information processing systems*, pp. 2424–2432, 2010.

Whitehill, Jacob, Wu, Ting-fan, Bergsma, Jacob, Movellan, Javier R, and Ruvolo, Paul L. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in neural information processing systems*, pp. 2035–2043, 2009.

Zhang, Yuchen, Chen, Xi, Zhou, Dengyong, and Jordan, Michael I. Spectral methods meet em: A provably optimal algorithm for crowdsourcing. In *Advances in neural information processing systems*, pp. 1260–1268, 2014.