THE BRAIN STRATEGY FOR ONLINE LEARNING

Stefan Vlaski, Bicheng Ying, and Ali H. Sayed

Department of Electrical Engineering, University of California, Los Angeles

ABSTRACT

Complexity is a double-edged sword for learning algorithms when the number of available samples for training in relation to the dimension of the feature space is small. This is because simple models do not sufficiently capture the nuances of the data set, while complex models overfit. While remedies such as regularization and dimensionality reduction exist, they themselves can suffer from overfitting or introduce bias. To address the issue of overfitting, the incorporation of prior structural knowledge is generally of paramount importance. In this work, we propose a BRAIN strategy for learning, which enhances the performance of traditional algorithms, such as logistic regression and SVM learners, by incorporating a graphical layer that tracks and learns in real-time the underlying correlation structure among feature subspaces. In this way, the algorithm is able to identify salient subspaces and their correlations, while simultaneously dampening the effect of irrelevant features. This effect is particularly useful for high-dimensional feature spaces.

Index Terms— Online learning, high-dimensional feature space, correlation, stochastic gradient learning.

1. INTRODUCTION

Given feature vectors $\boldsymbol{h} \in \mathbb{R}^M$ and binary class labels $\boldsymbol{\gamma} \in \{\pm 1\}$, the broad objective of learning solutions is to seek classifiers $c(\boldsymbol{h})$ from the set $\mathcal{C}$ that solve [1–4]:

$$c^\star(\boldsymbol{h}) = \arg\min_{c(\cdot)\in\mathcal{C}} \mathbb{P}\{c(\boldsymbol{h}) \neq \boldsymbol{\gamma}\} \tag{1}$$

The exact solution of (1) is generally intractable, mainly because it requires knowledge of the joint probability distribution of the feature and class variables. It is customary to replace the cost function by some regularized convex risk function and to seek instead the classifier, $c^o(\boldsymbol{h})$, that minimizes:

$$c^o(\boldsymbol{h}) = \arg\min_{c(\cdot)\in\mathcal{C}} \mathbb{E}\, Q\big(c(\cdot)\,;\,\boldsymbol{h},\boldsymbol{\gamma}\big) + R\big(c(\cdot)\big) \tag{2}$$

In this formulation, the term $R(c)$ denotes a regularizer intended to endow $c^o(\boldsymbol{h})$ with useful properties (such as sparsity), and $Q(\cdot)$ is a loss function. Under the assumption that the stochastic process generating realizations $\{h_n, \gamma(n)\}_{n=0}^{N-1}$ is ergodic, the mean in (2) can be approximated by its sample average, resulting in an empirical risk minimization problem directly in terms of the training data:

$$c^o(\boldsymbol{h}) \triangleq \arg\min_{c(\cdot)\in\mathcal{C}} \frac{1}{N}\sum_{n=0}^{N-1} Q\big(c(\cdot); h_n, \gamma(n)\big) + R\big(c(\cdot)\big) \tag{3}$$

This work was supported in part by NSF grants CCF–1524250 and ECCS–1407712, and DARPA project N66001–14–2–4029. Emails:{svlaski, ybc, sayed}@ucla.edu.

Various machine learning algorithms are derived from this perspective. Examples include logistic regression [2], support-vector machines [5, 6], as well as neural networks [7, 8] and deep neural networks [9, 10]. When the number of available samples N is much larger than the dimension of the feature space M and the VC dimension of the classifier set $\mathcal{C}$, it follows from the Vapnik-Chervonenkis theory [5] that the solution of (3) will result in a classifier with good generalization ability. This property refers to the fact that, although the classifier has been trained on a finite number of training data, it will still perform reasonably well on unseen data.

On the other hand, it is well known that when the number of samples N available for training is limited, appropriate prevention of overfitting becomes necessary. This scenario is common in cases where data collection is expensive, for example in biomedical applications, or when there is lack of information about the nature of the features, resulting in the collection of high dimensional feature data. Among the most commonly used remedies for overfitting are dimensionality reduction, feature selection, and regularization, all of which effectively reduce the complexity of the classifier set. However, several useful methods for dimensionality reduction, such as principal component analysis [1, 11] or Fisher discriminant analysis [12, 13], can still suffer from overfitting for small sample sizes.

In this work, we propose a framework for classification that involves an adaptive "soft" feature selection mechanism involving a graph topology that is also learned and tuned online during the same training process. The proposed framework is motivated by the observation that many feature spaces in practice include an implicit structure that may be learned and exploited for enhanced classification performance. The graph topology is used for this purpose; its role is to learn and track correlations among feature subspaces over time, and this information is fed into the learning algorithm in real-time. By doing so, the resulting learning machanism reduces the complexity of the classifier and combats overfitting. Once trained, one prominent feature of the proposed solution is that it provides an "x-ray" view into the correlation structure of the feature space, offering an opportunity for iterative refinement of the features.

Figure 1 provides a high level overview of the proposed architecture; it includes elements that are meant to mimic processing in the brain. The block with dictionary learning agents plays the role of a local memory that learns and stores foundational atoms (or basis) for the representation of feature subspaces. The block with the graph topology plays the role of interconnections that are also learned from correlations among the feature subspaces. Thus, while traditional learning algorithms focus on learning a mapping from the feature space to the class label, the proposed learning strategy slices the feature space into subspaces and incorporates learned memory and correlation graphs. We refer to the architecture in the figure as the BRAIN strategy, where the acroynm stands for **B**lock-**R**educed **A**daptation and **I**nference from **N**etworked subspaces. Due to space limitations, in this article, however, we do not study the BRAIN structure in its generality. As a proof of concept, we shall ignore

the dictionary blocks (i.e., we let the basis be the feature vectors themselves) and illustrate the enhancement that already results from exploiting the graphical correlation information alone.

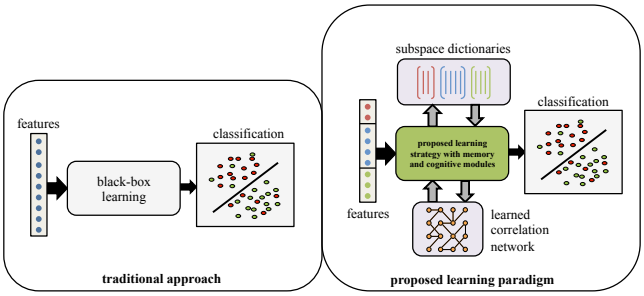

Fig. 1: (*left*) Traditional learning paradigm. (*right*) The BRAIN strategy with dictionary and correlation networks.

1.1. Relation to other works

Graphs have been used before as a useful tool to encode dependency among random variables, as happens, for example, in Bayesian and Markov networks [14, 15]. These structures are appropriate when there is a fundamental understanding of how the variables relate to each other. For the case when this information is not available, algorithms with latent variables, such as expectation-maximization algorithms [1, 16], restricted Boltzman machines [17], or deep belief networks [18] are generally employed. One drawback of such architectures is that, while powerful when trained with sufficient amounts of training data, the populated hidden layers are not always interpretable. In contrast, given only the structure of the feature space and no information about correlation, our solution attaches a single correlation layer to the shallow learner. Unlike deep strategies, this layer does not play a role in the actual classification decision, but rather learns and tracks low-variablility representations of the feature space. Moreover, this layer does not operate directly on the feature data but rather on scalar score variables defined in (7). These steps enable the algorithm to more accurately learn the subset of informative features and after convergence provides an insight into the correlation structure that resides in the feature space with respect to the classification decision.

2. ALGORITHM FORMULATION

Consider a large feature vector $\boldsymbol{h} \in \mathbb{R}^M$, which can be divided into a collection of subvectors $\boldsymbol{h}_k \in \mathbb{R}^{M_k}, k = 1, \ldots, K$, so that $\boldsymbol{h} = \text{col}\{\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_K\}$. These collections are application specific and can, for example, correspond to different bands in a densely sampled spectrogram, different performance metrics for a sector of the economy, or different regions of the human genome. In this work, we consider linear classifiers of the form:

$$c(\boldsymbol{h}) \triangleq \text{sign}(w^{\mathsf{T}} \boldsymbol{h}) \tag{4}$$

which can be decomposed under the assumed structure for the feature space into

$$c(\boldsymbol{h}) = \text{sign}\left(\sum_{k=1}^{K} w_k^{\mathsf{T}} \boldsymbol{h}_k\right) \tag{5}$$

where $w_k \in \mathbb{R}^{M_k}$ is the subvector of the linear classifier associated with $\boldsymbol{h}_k$, i.e.,

$$w \triangleq \text{col}\{w_1, w_2, \ldots, w_K\}, \quad \boldsymbol{h} \triangleq \text{col}\{\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_K\} \tag{6}$$

Traditional methods for dimensionality reduction operate directly on $\boldsymbol{h} \in \mathbb{R}^M$. They include projections techniques, such as PCA [1, 11] or FDA [12, 13] and selection techniques based on various measures of information — see for example [19, 20]. These methods rely on the computation of statistics of the feature vectors; accurate estimation of these statistics is challenging in high-dimensional spaces. Furthermore, projection based transformations are agnostic to the underlying structure of $\boldsymbol{h} = \text{col}\{\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_K\}$. The resulting classifier, based on a scrambled feature vector, can become difficult to interpret. In contrast, we propose to operate on the classifier soft sub-scores, defined as:

$$\boldsymbol{s}_w = \text{col}\{w_1^{\mathsf{T}} \boldsymbol{h}_1, w_2^{\mathsf{T}} \boldsymbol{h}_2, \ldots, w_K^{\mathsf{T}} \boldsymbol{h}_K\} \in \mathbb{R}^K, \tag{7}$$

This vector is of dimension $K \ll M$. Working with this reduced dimension has several advantages. First, overfitting is less likely to occur, as the dimension under consideration is significantly smaller than the underlying dimension of the feature space for appropriately chosen subvectors $\boldsymbol{h}_k$. Second, the structure of the feature space is preserved, allowing for more interpretable results. Third, we exploit the information gathered from statistics of $\boldsymbol{s}_w$ in real-time by feeding it back into the computation of w. This additional information results in more accurate estimate of w, which in turn stabilizes the statistics of $\boldsymbol{s}_w$ by reducing the weight of noisy features. This closed loop results in more accurate identification of relevant features.

To motivate the mechanism proposed in the sequel, recall that the general objective of feature selection in the context of classification is the identification of subsets of the feature vector $\boldsymbol{h}$, that are highly correlated with the class label $\boldsymbol{\gamma}$. Here we propose to obtain a measure of this correlation by analyzing the statistics of $\{w_k^{\mathsf{T}} \boldsymbol{h}_k\}$ directly. To this end, we recall the definition of the Pearson correlation coefficient of two scalar random variables $\boldsymbol{x}, \boldsymbol{y}$ with means $\mu_{\boldsymbol{x}}, \mu_{\boldsymbol{y}}$ and standard deviations $\sigma_{\boldsymbol{x}}, \sigma_{\boldsymbol{y}}$:

$$\rho_{\boldsymbol{x},\boldsymbol{y}} = \frac{\mathbb{E}\left[(\boldsymbol{x} - \mu_{\boldsymbol{x}})(\boldsymbol{y} - \mu_{\boldsymbol{y}})\right]}{\sigma_{\boldsymbol{x}} \sigma_{\boldsymbol{y}}} \tag{8}$$

We collect the absolute values of the pairwise correlation coefficients for the individual predictions, $\rho_{w_k^{\mathsf{T}} \boldsymbol{h}_k, w_\ell^{\mathsf{T}} \boldsymbol{h}_\ell}$, into a symmetric matrix $A \in [0, 1]^{K \times K}$, so that the element in the ℓ-th row and k-th column is defined as:

$$A^{(\ell k)} \triangleq a_{\ell k} \triangleq |\rho_{w_\ell^{\mathsf{T}} \boldsymbol{h}_\ell, w_k^{\mathsf{T}} h_k}| \tag{9}$$

This matrix is a measure of the linear correlations among predictions based on subsets of feature vector $\boldsymbol{h}$. A value $a_{\ell k}$ close to zero indicates that it is difficult to linearly predict the classification score based on the k-th sub-vector from the ℓ-th sub-vector and vice-versa. This implies that at least one of the sub-vectors contributes little information to the classification decision. Motivated by this observation, we proceed to interpret the A matrix as an adjacency matrix to a K-node graph, where each node k represents a sub-vector $\boldsymbol{h}_k$ of the feature vector $\boldsymbol{h}$. This is illustrated in Fig. 2, which corresponds to one particular realization of the BRAIN structure; in future works we will examine more elaborate structures, involving, in addition, local memory and dictionaries evolving over time.

The strength of the link between nodes k and ℓ is given by the absolute value of the Pearson corellation coefficient $\rho_{w_k^{\mathsf{T}} \boldsymbol{h}_k, w_\ell^{\mathsf{T}} \boldsymbol{h}_\ell}$. Nodes with strong connections have a tendency to agree in their predictions of the class variable. These predictions are in turn based on the vectors $\boldsymbol{h}_k$ and $\boldsymbol{h}_\ell$, respectively. In the sequel, we will show how to incorporate the learned correlation information into the online update of the learning algorithm. The objective is to devise an algorithm, where opinions of nodes in a strongly connected cluster

are reinforced. This behavior mimicks the fact that specific neural connections in the brain are reinforced as a result of learning [21].

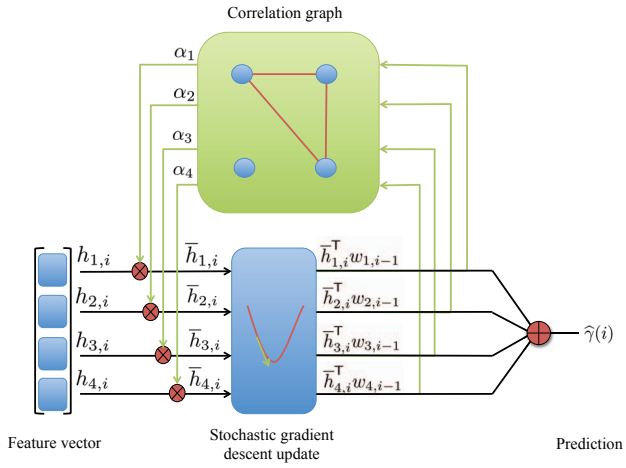

Fig. 2: Illustration of a correlation layer placed on top of an online learning algorithm.

3. CORRELATION-AWARE ONLINE UPDATE

We associate with each node k a scalar weight α_k, which is obtained from the adjacency matrix A of the graph according to

$$\alpha_k = \sum_{\ell=1}^{K} a_{\ell k} = \sum_{\ell=1}^{K} a_{k\ell} \tag{10}$$

These weights can be interpreted as a measure of trust placed in node k by its neighbors. This trust, loosely speaking, is the result of agreeing on classification decisions during past realizations of the feature vectors. We use this measure to scale incoming sub-vectors of the feature vector. The full algorithm, applied to a dataset of observations, $\{h_n, \gamma(n)\}_{n=0}^{N-1}$, generated form random variables $\{\boldsymbol{h}, \boldsymbol{\gamma}\}$, is summarized below where the notation $\partial Q(\cdot)$ refers to the gradient vector of $Q(\cdot)$ when it is differentiable or to a sub-gradient vector when it is non-differentiable. Likewise, for $\partial R(\cdot)$.

Algorithm 1 Online BRAIN strategy

Parameters: ν, N
Initialize: w_0, m_0, Σ_0
Run:
  for $i < N$ **do**
    Statistics:
    $s_{i-1} = \left(w_{1,i-1}^{\mathsf{T}} h_{1,i}, w_{2,i-1}^{\mathsf{T}} h_{2,i}, \ldots, w_{K,i-1}^{\mathsf{T}} h_{K,i}\right)^{\mathsf{T}}$
    $m_i = (1-\nu) m_{i-1} + \nu s_{i-1}$
    $\Sigma_i = (1-\nu)\Sigma_{i-1} + \nu\,(s_{i-1} - m_i)\,(s_{i-1} - m_i)^{\mathsf{T}}$
    Weights:
    $a_{\ell k}(i) = \frac{\Sigma_i^{(\ell k)}}{\sqrt{\Sigma_i^{(\ell\ell)} \Sigma_i^{(kk)}}},\ \forall\ \ell, k$
    $\alpha_k(i) = \sum_{\ell=1}^{K} a_{\ell k}(i),\ \forall\ k$
    Learning:
    $\overline{h}_i = \mathrm{col}\{\alpha_1(i) h_{1,i}, \alpha_2(i) h_{2,i}, \ldots, \alpha_K(i) h_{K,i}\}$
    $w_i = w_{i-1} - \mu \cdot \partial Q(w_{i-1}; \overline{h}_i, \gamma(i)) - \mu \cdot \partial R(w_{i-1})$
  end for
Return: w_N, Σ_N, A_N

Observe that the above algorithm is fully online, which is particularly useful when the feature vector is high-dimensional. The statistics information of $w_{k,i}^{\mathsf{T}} h_{k,i}$ is estimated adaptively, where the parameter ν controls the trade-off between accuracy of estimation and speed of convergence. The update of the weight vector w_i for classification is performed through a stochastic gradient step. For example, for online logistic regression, where

$$Q(w) = \ln\left(1 + e^{-\boldsymbol{\gamma}\,\boldsymbol{h}^{\mathsf{T}} w}\right), \quad R(w) = \delta \|w\|^2, \tag{11}$$

the algorithm would take the form

$$w_i = (1 - \mu\delta) w_{i-1} - \mu\gamma(i)\overline{h}_i \left(1 + e^{-\gamma(i)\overline{h}_i}\right)^{-1}. \tag{12}$$

For support vector machines with ℓ_2 regularization, where

$$Q(w) = \max\left(1 - \boldsymbol{\gamma}\,\boldsymbol{h}^{\mathsf{T}} w, 0\right), \quad R(w) = \delta \|w\|^2, \tag{13}$$

the algorithm becomes

$$w_i = (1 - \mu\delta) w_{i-1} - \mu\gamma(i)\overline{h}_i \cdot \mathbb{I}[\gamma(i)\overline{h}_i^{\mathsf{T}} w_{i-1} < 1] \tag{14}$$

where μ is the step size. The notation $\mathbb{I}[\cdot]$ represents the 0-1 indicator function, which is equal to 1 when the statement is true and 0 when it is false.

4. SIMULATION RESULTS

4.1. Artificial Data

We begin by illustrating performance on synthetic data. The dataset is generated using the `make_classification` function[1] from the `sklearn.datasets` Python module [22], which is adapted from one of the datasets in the 2003 NIPS feature selection challenge [23]. The method allows for the specification of the number of informative and non-informative features. We generate $N = 3000$ feature vectors of dimension $M = 400$, where only the first 40 indices contain class information. The remaining 360 indices contain noise.

To begin with, we confirm that the statistics of classifier scores indeed allow the classifier to learn the subset of informative features. In Fig. 3 we show the evolution of the correlation network, which controls the weighting of the incoming feature blocks. We represent the weight α_k of node k through the size of its dot, and the correlation between a pair of subvectors of h through the thickness of the connecting link. The correlation matrix Σ is initialized as the identity matrix, resulting in a set of K unconnected nodes and $\alpha_k = \frac{1}{K}$, depicted in the leftmost plot. The second plot depicts the state of the correlation network after convergence, resulting in a fully connected network, albeit with two dominant nodes, namely nodes 1 and 2, which correspond to the first 40 elements of the feature vector, which is the informative subset. This dominance becomes more clear in the rightmost plot, where weak links were removed by simple thresholding. It is evident that there is a strong link between nodes 1 and 2. This means that decisions formed from the first 20 elements and those formed from the second set of 20 features have a strong tendency to agree. These are in fact the informative subset, as constructed. In contrast, none of the decisions formed based on the remaining 18 subsets show any meaningful correlation.

[1] `http://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_classification.html`

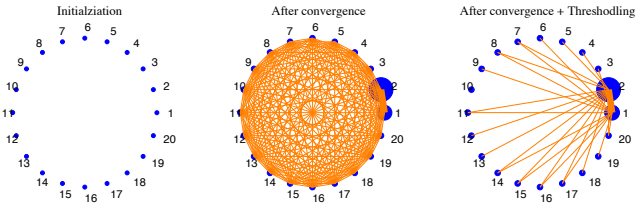

Fig. 3: Evolution of correlation network of classifier sub-scores.

Figure 4 shows the evolution of the classification accuracy for ordinary online logistic regression compared to BRAIN online logistic regression. For this particular example, we observe fastest convergence and highest performance for a damping factor $\nu = 0.01$ and $K = 20$ equally spaced divisions of the feature vector. We show two additional accuracy evolutions to assess the sensitivity of the algorithm performance for varying design choices.

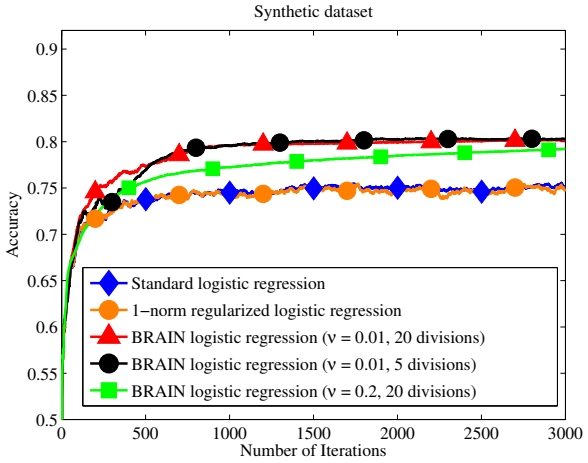

Fig. 4: Learning curves for logistic regression with and without the correlation layer on synthetic data.

4.2. p53 Mutants Dataset

Here, we test the performance of the algorithm on real data, compiled in the University of California, Irvine (UCI) Machine Learning Repository[2], discussed in [24]. The dataset contains biophysical features pertaining to the p53 protein, which is also known as a tumor suppressor protein. When active, p53 guards the genome against cancer. The objective is to predict the state of p53 (active or inactive) from $M = 5408$ features. One challenge in this dataset is that the classes are highly unbalanced, with a majority of p53 realizations being active (healthy). Of the 16772 available instances, only 286 are inactive. To remedy this imbalance, we randomly select 286 active instances. After leaving 72 samples for testing, we are left with a training set of size $N = 500$.

Here we endow a support vector machine with the correlation layer and compare performance against ordinary SVM in Fig. 5. Since we have no prior information on the structure of the feature space, we divide the feature vector into $K = 50$ divisions of equal size. To allow the algorithms to converge, we run multiple passes over the small training set.

We show the learned correlation network in Fig. 6. We observe that the nodes in the bottom left form a cluster (nodes 25–43) and contribute most strongly to the classification decision.

[2] http://archive.ics.uci.edu/ml/datasets/p53+Mutants

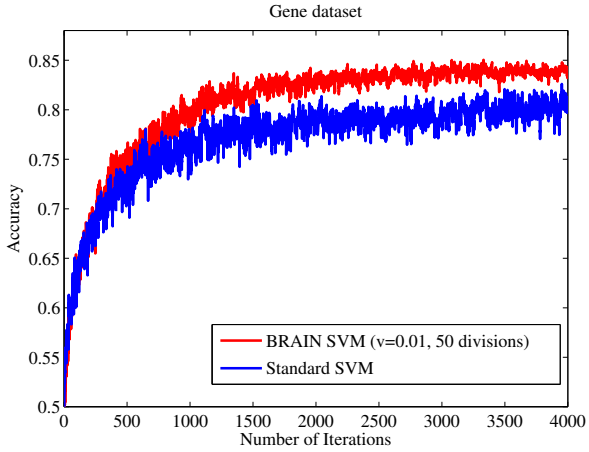

Fig. 5: Learning curves for Support-Vector-Machine with and without correlation layer on gene data, $\mu = 0.01$, $\nu = 0.01$, and $\rho = 0.01$.

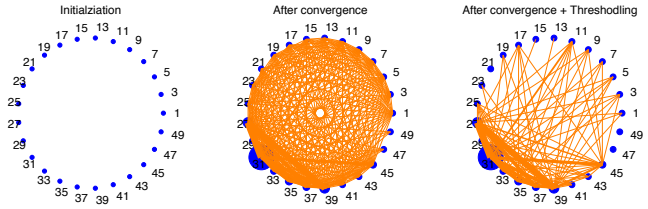

Fig. 6: Correlation network evolution on p53 mutants.

5. CONCLUSION AND FUTURE WORK

We proposed a BRAIN strategy to enhance the performance of online classifiers for high-dimensional feature spaces. We illustrated results and performance on both artificially generated and real data examples and observed experimentally that **(a)** the correlation layer is able to identify the subset of informative features; **(b)** this information seeps into the final weight vector, and **(c)** this results in improved performance when compared to regular versions of the respective online learners.

This work opens avenues for further research. Recall that one of the key features of the correlation layer is that it weights features based on classifier subscores. These can be interpreted as single-dimensional projections of the sub-feature vectors with reduced variance. More elaborate and possibly higher-dimensional, albeit still variance-reduced, representations can be considered by means of dictionaries, which are updated in an online manner, similar to [25]. A second opportunity for improvement is the automatic and iterative refinement of feature vector divisions in the absence of exact prior knowledge. In this work, we were able to show performace improvement with evenly spaced divisions, but do not make a claim of optimality. On the other hand, correlation networks after convergence contain information on the amount of information contained in a given feature subset. This information can be used to inform a restructuring of the feature vector subsets, before running the algorithm again with the previous weight vector as a starting point. In this manner, the information provided by the correlation graph can be more fully exploited. Finally, distributed implementations can be pursued, along the lines of [3, 26]. This is particularly useful when feature samples are available at dispersed locations.

6. REFERENCES

[1] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[2] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, 2009.

[3] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Foundations and Trends in Machine Learning*, vol. 7, no. 4-5, pp. 311–801, July 2014.

[4] S. Theodoridis, *Machine Learning: A Bayesian and Optimization Perspective*, Academic Press, 2015.

[5] V. Vapnik, *Statistical Learning Theory*, Wiley NY, 1998.

[6] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[7] C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.

[8] S. Haykin, *Neural Networks and Learning Machines (3rd Edition)*, Prentice Hall, 2009.

[9] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[10] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, pp. 1097–1105. Lake Tahoe, USA, 2012.

[11] I. Jolliffe, *Principal Component Analysis*, Wiley NY, 2002.

[12] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, 2013.

[13] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K. R. Mullers, "Fisher discriminant analysis with kernels," in *Neural Networks for Signal Processing IX*, pp. 41–48. Madison, USA, Aug 1999.

[14] C. M. Bishop, "Model-based machine learning," *Phil. Trans. Royal Society of London A*, vol. 371, no. 1984, pp. 1–17, 2012.

[15] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*, MIT Press, 2009.

[16] M. J. Beal, *Variational Algorithms for Approximate Bayesian Inference*, Ph.D. thesis, University College London, London, United Kingdom, May 2003.

[17] G. E. Hinton, "A practical guide to training restricted boltzmann machines," in *Neural Networks: Tricks of the Trade: Second Edition*, pp. 599–619. Springer, 2012.

[18] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, July 2006.

[19] M. A. Hall, *Correlation-based Feature Selection for Machine Learning*, Ph.D. thesis, University of Waikato, Hamilton, New Zealand, April 1999.

[20] G. Brown, A. Pocock, M.-J. Zhao, and M. Luján, "Conditional likelihood maximisation: A unifying framework for information theoretic feature selection," *J. Mach. Learn. Res.*, vol. 13, pp. 27–66, Jan. 2012.

[21] Y. Zhang, R. H. Cudmore, D.-T. Lin, D. J. Linden, and R. L. Huganir, "Visualization of NMDA receptor-dependent AMPA receptor synaptic plasticity in vivo," *Nat Neurosci*, vol. 18, no. 3, pp. 402–407, 03 2015.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[23] I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror, "Result Analysis of the NIPS 2003 Feature Selection Challenge," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, pp. 545–552. Montreal, Canada, 2005.

[24] S. A. Danziger, S. J. Swamidass, Jue Zeng, L. R. Dearth, Qiang Lu, J. H. Chen, J. Cheng, V. P. Hoang, H. Saigo, R. Luo, P. Baldi, R. K. Brachmann, and R. H. Lathrop, "Functional census of mutation sequence spaces: the example of p53 cancer rescue mutants," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 3, no. 2, pp. 114–125, April 2006.

[25] J. Chen, Z. J. Towfic, and A. H. Sayed, "Dictionary learning over distributed models," *IEEE Transactions on Signal Processing*, vol. 63, no. 4, pp. 1001–1016, Feb 2015.

[26] A. H. Sayed, "Adaptive networks," *Proceedings of the IEEE*, vol. 102, no. 4, pp. 460–497, April 2014.