Parallel Gaussian-Bernoulli Restricted Boltzmann Machines with Self-Organizing Maps for Land Cover Classification of Hyperspectral Imagery

Jesus Torres
Student at Departamento de Electrónica, Sistemas e Informática
Instituto Tecnológico y de Estudios Superiores de Occidente
Guadalajara, México
j.jesus@outlook.com

***Abstract*—This paper introduces the use of Gaussian-Bernoulli Restricted Boltzmann Machines (GBRBM) in parallel and Self-Organizing Maps (SOM) for hyperspectral image clustering of land cover. The proposed approach takes advantage of the GBRBMs as feature extractors and uses a final layer of SOM to group these features into a number of clusters representing regions of similar land cover. The model is trained over two public hyperspectral datasets: Salinas and Pavia University datasets. Experimental results show that the proposed method offers good performance for completely unsupervised clustering of hyperspectral images.**

Keywords—restricted Boltzmann machines, clustering, unsupervised learning

I. Introduction

Many machine learning techniques have been applied, such as K-nearest neighbor [1], Bayes classifier [2], and support vector machines (SVM) [3], to achieve classification. However, the increase of spectral dimensions leads to low efficiency of classification algorithms and large computational cost [4]. Therefore it is important to reduce dimensionality before classification. Several techniques for dimensionality reduction has been developed and documented. There are linear techniques, among others, Independent Component Analysis, Linear Discriminant Analysis or Supervised linear manifold learning. Non linear models are also applied for this task, these methods include: linear embedding, Isomap and Laplacian eigenmaps or kernel PCA. Deep learning has also brought new methods that has improved several aspects of previous models. Between these, some techniques are: deep belief networks (DBNs), deep Boltzmann machines, deep auto-encoder neural networks and deep convolutional neural networks. Deep neural networks, like the ones mentioned previously, have shown an improved performance when compared against traditional machine learning as statistical methods. However it has been noted that for hyperspectral classification, classification accuracy may be degraded with the increase of hidden layers.

A recent architecture, proposes parallel Gaussian Bernoulli Restricted Boltzmann Machines, or GBRBMs, to improve performance of feature extraction without the downsides of increasing the number of layers [5]. The parallel GBRBMs are tuned in their hyperparameters, more specifically, in the number of hidden neurons. This allows each GBRBMs to extract a different set of high level features that are fed into a classifier.

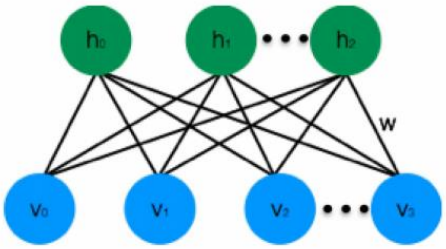

Figure 1: Restricted Boltzmann Machine with visible units and hidden units

The rest of the paper is organized as follows. Section II explains the theory behind Restricted Boltzmann Machines, or RBMs, and GBRBMs. Section III explains the datasets and model built for this project, Section IV shows the results obtained and Section V the conclusions.

II. Background Data

A. Restricted Boltzmann Machines

Restricted Boltzmann Machines are unsupervised nonlinear feature learners based on a probabilistic model. The features extracted by the RBM or a hierarchy of RBMs often provide good results when fed into a classifier. The RBM are probabilistic graphical models that can be interpreted as stochastic neural networks which are a particular form of the log-linear Markov Random Field (MRF). The conventional model of RBM composes of a layer of visible unit's v and hidden layer h, connected by weighted connection, as shown in Fig. 1.

The pixels correspond to “visible” units of the RBM because their states are observed; the feature detectors correspond to “hidden” units [6]. A joint configuration, (v, h) of the visible and hidden units has an energy [7] given by:

$$E(\mathbf{v},\mathbf{h}) = -\sum_{i\in\text{visible}} a_i v_i - \sum_{j\in\text{hidden}} b_j h_j - \sum_{i,j} v_i h_j w_{ij} \quad (1)$$

where v_i , h_j are the binary states of visible unit i and hidden unit j, a_i , b_j are their biases and w_{ij} is the weight between them. The

network assigns a probability to every possible pair of a visible and a hidden vector via this energy function:

$$p(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{v},\mathbf{h})} \tag{2}$$

where the "partition function", *Z*, is given by summing over all possible pairs of visible and hidden vectors:

$$Z = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} \tag{3}$$

The probability that the network assigns to a visible vector, *v*, is given by summing over all possible hidden vectors:

$$p(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})} \tag{4}$$

The probability that the network assigns to a training image can be raised by adjusting the weights and biases to lower the energy of that image and to raise the energy of other images, especially those that have low energies and therefore make a big contribution to the partition function. The derivative of the log probability of a training vector with respect to a weight is surprisingly simple.

$$\frac{\partial \log p(\mathbf{v})}{\partial w_{ij}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model} \tag{5}$$

where the angle brackets are used to denote expectations under the distribution specified by the subscript that follows. This leads to a very simple learning rule for performing stochastic steepest ascent in the log probability of the training data:

$$\Delta w_{ij} = \epsilon(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model}) \tag{6}$$

where ε is a learning rate.

Because there are no direct connections between hidden units in an RBM, it is very easy to get an unbiased sample of $\langle v_i h_j \rangle_{data}$. Given a randomly selected training image, *v*, the binary state, h_j, of each hidden unit, *j*, is set to 1 with probability

$$p(h_j = 1 \mid \mathbf{v}) = \sigma(b_j + \sum_i v_i w_{ij}) \tag{7}$$

where $\sigma(x)$ is the logistic sigmoid function $1/(1 + \exp(-x))$. $v_i h_j$ is then an unbiased sample. Because there are no direct connections between visible units in an RBM, it is also very easy to get an unbiased sample of the state of a visible unit, *given a hidden vector*

$$p(v_i = 1 \mid \mathbf{h}) = \sigma(a_i + \sum_j h_j w_{ij}) \tag{8}$$

Getting an unbiased sample of $\langle v_i h_j \rangle_{model}$, however, is much more difficult. It can be done by starting at any random state of the visible units and performing alternating Gibbs sampling for a very long time. One iteration of alternating Gibbs sampling consists of updating all of the hidden units in parallel using equation 7 followed by updating all of the visible units in parallel using equation 8.

A much faster learning procedure was proposed in [7]. This starts by setting the states of the visible units to a training vector. Then the binary states of the hidden units are all computed in parallel using equation 7. Once binary states have been chosen for the hidden units, a "reconstruction" is produced by setting each v_i to 1 with a probability given by equation 8. The change in a weight is then given by

$$\Delta w_{ij} = \epsilon(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{recon}) \tag{9}$$

A simplified version of the same learning rule that uses the states of individual units instead of pairwise products is used for the biases.

The learning works well even though it is only crudely approximating the gradient of the log probability of the training data [7]. The learning rule is much more closely approximating the gradient of another objective function called the Contrastive Divergence [7] which is the difference between two Kullback-Liebler divergences, but it ignores one tricky term in this objective function so it is not even following that gradient. Indeed, Sutskever and Tieleman have shown that it is not following the gradient of any function [9]. Nevertheless, it works well enough to achieve success in many significant applications.

RBMs typically learn better models if more steps of alternating Gibbs sampling are used before collecting the statistics for the second term in the learning rule, which will be called the negative statistics. CD_n will be used to denote learning using *n* full steps of alternating Gibbs sampling.

B. Gaussian-Bernoulli Restricted Boltzmann Machines

For data such as patches of natural images or the Mel-Cepstrum coefficients used to represent speech, logistic units are a very poor representation. One solution is to replace the binary visible units by linear units with independent Gaussian noise. The energy function then becomes:

$$E(\mathbf{v}, \mathbf{h}) = \sum_{i \in \text{vis}} \frac{(v_i - a_i)^2}{2\sigma_i^2} - \sum_{j \in \text{hid}} b_j h_j - \sum_{i,j} \frac{v_i}{\sigma_i} h_j w_{ij} \tag{10}$$

where σ_i is the standard deviation of the Gaussian noise for visible unit *i*.

It is possible to learn the variance of the noise for each visible unit but this is difficult using CD_1. In many applications, it is much easier to first normalise each component of the data to have zero mean and unit variance and then to use noise free reconstructions, with the variance in equation 10 set to 1. The reconstructed value of a Gaussian visible unit is then equal to its top-down input from the binary hidden units plus its bias.

The learning rate needs to be about one or two orders of magnitude smaller than when using binary visible units and some of the failures reported in the literature are probably due to using a learning rate that is much too big. A smaller learning rate is required because there is no upper bound to the size of a component in the reconstruction and if one component becomes very large, the weights emanating from it will get a very big learning signal. With binary hidden and visible units, the learning signal for each training case must lie between −1 and 1, so binary-binary nets are much more stable.

III. Proposed Method

A. *Dataset Description*

In this project, the Salinas fields dataset is used to verify the effectiveness of the proposed model. This scene was collected by the 224-band AVIRIS sensor over Salinas Valley, California, and is characterized by high spatial resolution (3.7-meter pixels). The area covered comprises 512 lines by 217 samples. It includes vegetables, bare soils, and vineyard fields. Salinas groundtruth contains 16 classes as shown in Table I.

B. *Feature Extraction and Clustering Using Multiple GBRBM in Parallel*

The basic idea is to use multiple GBRBMs in parallel to overcome the limitations posed by GBRBMs sequentially when used for feature extraction. It has been argued that, when used sequentially, GBRBMs performance is degraded. A multiple parallel configuration is not affected in this regard. After the training is done in each GBRBM, the extracted features are fed

TABLE I
GROUNDTRUTH CLASSES FOR THE SALINAS SCENE AND THEIR RESPECTIVE SAMPLES NUMBER

#	Class	Samples
1	Brocoli_green_weeds_1	2,009
2	Brocoli_green_weeds_2	3,726
3	Fallow	1,976
4	Fallow_rough_plow	1,394
5	Fallow_smooth	2,678
6	Stubble	3,959
7	Celery	3,579
8	Grapes_untrained	11,271
9	Soil_vinyard_develop	6,203
10	Corn_senesced_green_weeds	3,278
11	Lettuce_romaine_4wk	1,068
12	Lettuce_romaine_5wk	1,927
13	Lettuce_romaine_6wk	916
14	Lettuce_romaine_7wk	1,070
15	Vinyard_untrained	7,268
16	Vinyard_vertical_trellis	1,807

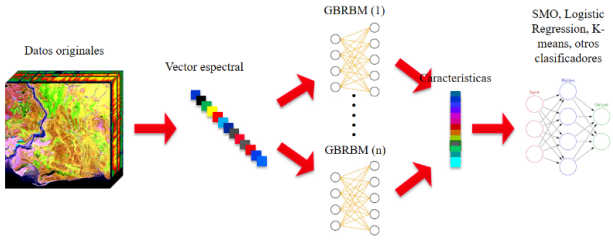

Figure 2: Structure of proposed model using multiple GBRBMs in parallel to extract high-level features, which feeds the classifiers.

to a classifier. In the experiments done until now, the classifier used is K-means. For final experiments, a self organizing map will be implemented. The structure of the model proposed is described in Fig. 2.

IV. Experiments and Results

In this section, I show the results of the clustering model compared visually against the groundtruth of the dataset. In Fig. 3, the image at the left is the false color RGB image, at the center is the groundtruth and, finally, at the right, the clustered image generated by the algorithm.

V. Conclusions

GBRBMs offer feature extraction capabilities for real applications in the space of hyperspectral image clustering. However, there are parameters to be adjusted and a careful review of the model structure must be made. Also, one must analyze if the model should be sequential or if it would make more sense to apply a parallel paradigm for the neural network.

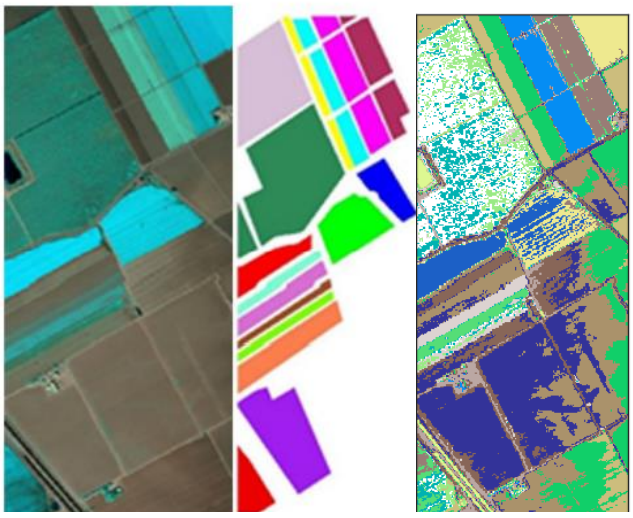

Figure 3: Visual comparison between false color RGB image of the dataset, its groundtruth and the generated clustered image.

VI. References

[1] M. M. Crawford, M. Li, X. Yang and Y. Guo, "Local-manifold-learningbased graph construction for semisupervised hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 5, pp. 2832-2844, 2015.

[2] A. Bhattacharya and D. Dunson, "Nonparametric Bayes classification and hypothesis testing on manifolds," *J. Multivariate Anal.*, vol. 111, no. 5, pp. 1-19, 2012.

[3] K. Tan, J. Zhang, D. Qian and X. Wang, "GPU parallel implementation of support vector machines for hyperspectral image classification," *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.*, vol. 8, no. 10, pp. 4647-4656, 2015.

[4] C. I. Chang, Hyperspectral Data Exploitation: Theory and Applications, New York, NY, USA: Wiley, 2007.

[5] K. Tan, F. Wu, Q. Du, P. Du and Y. Chen, "A Parallel Gaussian–Bernoulli Restricted Boltzmann Machine for Mining Area Classification With Hyperspectral Imagery," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 12, no. 2, pp. 627 - 636, 2019.

[6] G. Hinton, "A Practical Guide to Training Restricted Boltzmann Machines," University of Toronto, Toronto, Canada, 2010.

[7] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," in *Proceedings of the National Academy of Sciences*, 1982.

[8] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1711-1800, 2002.

[9] I. Sutskever and Tieleman, "On the convergence properties of contrastive divergence," in *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, Sardinia, Italy, 2010.