ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

Learning Vector Quantization in Context of Information Bottleneck Theory

M. Mohannazadeh Bakhtiari, D. Staps, and T. Villmann

Mittweida University of Applied Sciences
– Dept. of Comp. Science & Bio-informatics –
Mittweida - Germany

Abstract. This paper is an effort to parameterize Information Bottle-neck Theory to become a supervised classifier. We introduce a parametrization by means of Learning Vector Quantization. With this new approach, one can find suitable components that are necessary for an accurate, yet efficient, classification. A balance between compression and representation is made by means of a specially designed objective function.

1 Introduction

Learning Vector Quantization (LVQ) methods [1, 2] are effective classifiers that naturally compress data. However, LVQ methods are often limited by their architecture that lacks flexibility. There has been efforts [3, 4] to make LVQ methods more flexible by allowing prototypes to represent more than one class. In this paper, such flexible prototypes are called components. The flexibility of components in LVQ may cause instability in learning. In case of Classification-by-components (CbC) [4], modified CbC [5] redefines the reasoning parameters for a more stable learning process. Nonetheless, both CbC and the modified version depend on initialization of the components.

Given the mentioned problems, it is important to define a suitable objective function as well as a suitable adaptation mechanism in order to obtain a model with self-sufficient feature extractor. Also, we need to start the learning process with many components and drop the components that contribute less, as we train the model, to discover better components. One possible solution is using Information Bottleneck Theory (IBT) [6] as a general probabilistic framework for LVQ. IBT is a general method for compressing data, while keeping as much relevant information as possible. IBT optimizes an objective function that makes a careful trade-off between compression and distortion using mutual information. However, optimizing the parameters of IBT has remained a challenging task [7], despite of the fact that IBT algorithm has been proven to converge to a local optimum. Deterministic Information Bottleneck (DIB) [8] is a variant of IBT that uses a deterministic mapping from the input space to the compression space. However, the deterministic assignment of input to the components space limits the flexibility of the model and we would like to have a soft assignment. Therefore, neither IBT nor DIB is a perfect fit for LVQ.

Inspired by IBT and DIB, we propose a special information theoretic framework for LVQ. We introduce an objective function that is more suitable for LVQ.

ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

The optimal assignment from input space to the compression space is a soft assignment. The new method is called Information Theoretic LVQ (ITLVQ).

In the rest of the paper, an introduction to IBT and DIB is provided in section 2. In section 3, we develop ITLVQ, that is LVQ in context of IBT. Section 4 contains the simulations. Finally, in section 5, the paper is concluded.

2 Information Bottleneck Theory for Data Classification

C. Shannon [9] has given a general information-theoretic framework for compressing data efficiently. However, in the context of classification, compressing the data to keep the most relevant information to the final classification requires a slightly different approach. Tishby et al. [6] introduced Information Bottleneck Theory (IBT) that is a general probabilistic method to compress data, while keeping the maximum information. Given an input space X , and a discrete target space $Y = \{y_1, \ldots, y_c\}$, IBT codes X into a compression space K, such that K preserves the maximum information about Y. The probabilistic mapping from the input space X to the compression space K is denoted by the conditional probability $p(\Bbbk|\mathbb{x})$, where $\mathbb{x} \in X$ and $\Bbbk \in K$. The probabilistic mapping from the compression space K to the output space Y is denoted by $p(\mathbb{y}|\Bbbk)$. The objective function of IBT is

$$F_{IBT}(p(\Bbbk|\mathbb{x})) = I(X, K) - \beta \cdot I(Y, K) \tag{1}$$

where $I(.,.)$ is the mutual information and β is the rate-compression trade-off parameter. Finally, in order to find the optimal solution to the above objective function, Tishby et al. [6] include an iterative algorithm

$$\begin{cases} p_d(\Bbbk|\mathbb{x}) = \frac{p_d(\Bbbk)}{Z_d(\mathbb{x},\beta)} \cdot \exp(-\beta \cdot D_{KL}[p(\mathbb{y}|\mathbb{x}), p_d(\mathbb{y}|\Bbbk)]) \\ p_{d+1}(\Bbbk) = \sum_{\mathbb{x}} p(\mathbb{x}) \cdot p_d(\Bbbk|\mathbb{x}) \\ p_{d+1}(\mathbb{y}|\Bbbk) = \frac{1}{p_{d+1}(\Bbbk)} \sum_{\mathbb{x}} p(\mathbb{y}|\mathbb{x}) \cdot p_d(\Bbbk|\mathbb{x}) \cdot p(\mathbb{x}) \end{cases} \tag{2}$$

that estimates all the necessary parameters at time step t based on the optimal mapping $p_d(\Bbbk|\mathbb{x})$ that appears in the first line in (2). Note that $Z_t(\mathbb{x}, \beta)$ is a normalization factor and $D_{KL}[.,.]$ is the Kullback-Leibler divergence between two distributions. Furthermore, $p(\mathbb{y}|\mathbb{x}) = [p(y_1|\mathbb{x}), \ldots, p(y_c|\mathbb{x})]^T$ and $p(\mathbb{y}|\Bbbk) = [p(y_1|\Bbbk), \ldots, p(y_c|\Bbbk)]^T$ are distribution vectors such that the elements of each vector sum up to one. Also, the following probabilities are defined. $p(\mathbb{x})$ is the marginal probability that the input $\mathbb{x}$ occurs. $p(\Bbbk) = \sum_{\mathbb{x}} p(\mathbb{x}) \cdot p(\Bbbk|\mathbb{x})$ is the marginal probability of component $\Bbbk \in K$. $p(\mathbb{y}) = \sum_{\Bbbk} p(\Bbbk) \cdot p(\mathbb{y}|\Bbbk)$ is the probability of the output $\mathbb{y} \in Y$.

Deterministic Information Bottleneck (DIB) [8] modifies the objective of IBT (1) to have $F_{DIB}(p(\Bbbk|\mathbb{x})) = H(K) - \beta \cdot I(Y, K)$. The optimal mapping $p(\Bbbk|\mathbb{x})$ of DIB is the hard assignment $f(\mathbb{x}) = \arg\max_{\Bbbk} \big(\log p(\Bbbk) - \beta \cdot D_{KL}[p(\mathbb{y}|\mathbb{x}), p(\mathbb{y}|\Bbbk)] \big)$.

ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

3 Information Theoretic LVQ

In this section, we introduce Information Theoretic LVQ (ITLVQ) based on the notions of IBT and DIB. ITLVQ will have a recursive learning algorithm based on a limited number of components. We assume that a training set $T = \{(\mathbb{x}_i, y_i)\}_{i=1}^{\tau}$ is given, such that $\mathbb{x}_i \in \mathbb{R}^n$ and $y_i \in C = \{1, 2, \ldots, c\}$ for all i. We would also like to start with a number m of components $W = \{\mathbb{w}_i\}_{i=1}^{m}$, such that $\mathbb{w}_i \in \mathbb{R}^n$ for all i. The following Bayesian probabilistic predictor for the class $j \in C$ of input $\mathbb{x}$ is defined

$$p_W(j|\mathbb{x}) = \sum_t p(\mathbb{w}_t|\mathbb{x}) \cdot p(j|\mathbb{w}_t) \tag{3}$$

where we have assumed that the components W provide sufficient statistics for the input space to predict the class of the input data (the Markov condition), so we can use $p(j|\mathbb{w}_t)$ instead of $p(j|\mathbb{w}_t, \mathbb{x})$ in (3). The relation $\sum_j p_W(j|\mathbb{x}) = 1$ must hold true.

In order to parameterize the notions of IBT and DIB for an LVQ method, we define the representation function $s_t(\mathbb{x}) : W \times \mathbb{R}^n \to [0, 1]$, such that $s_t(\mathbb{x})$ is a parametric measure of the assignment $p(\mathbb{w}_t|\mathbb{x})$. We require $\sum_t s_t(\mathbb{x}) = 1$ for all $\mathbb{x}$. We also define the representation vector $s(\mathbb{x}) = [s_1(\mathbb{x}), \ldots, s_m(\mathbb{x})]^T$.

We continue by defining a suitable objective function for ITLVQ with inspiration from objective functions of IBT and DIB. Basically, we would like an objective function that is the sum of two terms, where one takes care of compression and the another takes care of representation. The trade-off between compression and representation is regulated by a parameter β. The objective function of ITLVQ for a training pair $(\mathbb{x}, y)$ is defined as below.

$$F_{ITLVQ} = Cr\left[p(W|\mathbb{x}), \frac{1}{\gamma}p(y|W)\right] + \beta \cdot D_{KL}\left[p(W|\mathbb{x}), s(\mathbb{x})\right] - \sum_t \lambda \cdot p(\mathbb{w}_t|\mathbb{x}) \tag{4}$$

where $Cr[.,.]$ is the cross entropy between two distributions. We also have $p(W|\mathbb{x}) = [p(\mathbb{w}_1|\mathbb{x}), \ldots, p(\mathbb{w}_m|\mathbb{x})]^T$, $p(y|W) = [p(y|\mathbb{w}_1), \ldots, p(y|\mathbb{w}_m)]^T$, and γ is a normalizing factor, since the elements of $p(y|W)$ may not add up to one. The last term $\sum_t \lambda \cdot p(\mathbb{w}_t|\mathbb{x})$ in (4), with positive Lagrangian multiplier λ, enforces the normalization of parameter $p(\mathbb{w}_t|\mathbb{x})$. We expand (4) for a better explanation below.

$$\begin{aligned} F_{ITLVQ} = \quad & - \sum_t p(\mathbb{w}_t|\mathbb{x}) \cdot \log \frac{1}{\gamma}p(y|\mathbb{w}_t) + \beta \cdot \sum_t p(\mathbb{w}_t|\mathbb{x}) \cdot \log p(\mathbb{w}_t|\mathbb{x}) \\ & - \beta \cdot \sum_t p(\mathbb{w}_t|\mathbb{x}) \cdot \log s_t(\mathbb{x}) - \sum_t \lambda \cdot p(\mathbb{w}_t|\mathbb{x}) \end{aligned} \tag{5}$$

In (5), $-\sum_t p(\mathbb{w}_t|\mathbb{x}) \cdot \log \frac{1}{\gamma}p(y|\mathbb{w}_t)$ makes sure that maximum compression is achieved by mapping a data $\mathbb{x}$ to a component $\mathbb{w}_t$ that has the maximum probability $\frac{1}{\gamma}p(y|\mathbb{w}_t)$, where y is the true class of $\mathbb{x}$. The term $\sum_t p(\mathbb{w}_t|\mathbb{x}) \cdot \log s_t(\mathbb{x})$, on the other hand, tries to maximize representation by mapping a

ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

data point $\mathbb{x}$ to the closest component $\mathbb{w}_t$, based on the representation function $s_t(\mathbb{x})$. We would also like to minimize the uncertainty of the mapping $p(\mathbb{w}_t|\mathbb{x})$ by minimizing the entropy $\sum_t p(\mathbb{w}_t|\mathbb{x}) \cdot \log p(\mathbb{w}_t|\mathbb{x})$. We find the optimal assignment rule in the following theorem.

Theorem 1. *The optimal assignment, based on objective function* (4)*, for a training pair* $(\mathbb{x}, y)$ *is*

$$p^*(\mathbb{w}_t|\mathbb{x}) = N \cdot p^{\frac{1}{\beta}}(y|\mathbb{w}_t) \cdot s_t(\mathbb{x}) \tag{6}$$

where $N = \left(\frac{e^{\lambda-\beta}}{\gamma}\right)^{\frac{1}{\beta}}$ is a normalization factor and e is the Euler constant.

Proof. We start from (5) and take the gradient with respect to $p(\mathbb{w}_k|\mathbb{x})$ and set it equal to zero.

$$\frac{\partial F_{ITLVQ}}{\partial p(\mathbb{w}_k|\mathbb{x})} = -\log\frac{p(y|\mathbb{w}_k)}{\gamma} + \beta \cdot (1 + \log p(\mathbb{w}_k|\mathbb{x})) - \beta \log s_k(\mathbb{x}) - \lambda = 0$$

The above equation is solved for $p(\mathbb{w}_k|\mathbb{x})$.

$$p^*(\mathbb{w}_k|\mathbb{x}) = \left(\frac{e^{\lambda-\beta}}{\gamma}\right)^{\frac{1}{\beta}} \cdot p^{\frac{1}{\beta}}(y|\mathbb{w}_k) \cdot s_k(\mathbb{x}) \tag{7}$$

□

We often have a limited number of components W in LVQ for a fast and computationally efficient learning. Therefore, it would be beneficial to be able to adapt the component for an optimal result. Based on the optimal component assignment (6), we may find the gradient of objective (4) with respect to a component $\mathbb{w}_k$, assuming that only representation functions $s_t(\mathbb{x})$ explicitly depend on $\mathbb{w}_k$.

$$\frac{\partial F_{ITLVQ}}{\partial \mathbb{w}_k} = -\beta \cdot \sum_t \frac{p^*(\mathbb{w}_t|\mathbb{x})}{s_t(\mathbb{x})} \cdot \frac{\partial s_t(\mathbb{x})}{\partial \mathbb{w}_k} \tag{8}$$

Then, we write the above gradient for all $\mathbb{x}$ in the training set to have the following average update for components, where ϵ is a small learning rate.

$$\mathbb{w}_k(new) = \mathbb{w}_k(old) + \epsilon \cdot \beta \cdot \sum_{\mathbb{x}} \sum_t \frac{p^*(\mathbb{w}_t|\mathbb{x})}{s_t(\mathbb{x})} \cdot \frac{\partial s_t(\mathbb{x})}{\partial \mathbb{w}_k} \tag{9}$$

The learning algorithm of ITLVQ, similar to the recursion (2), is as bellow, where the order of learning is the same as the order in which the relations appear.

$$\begin{cases} p^*(\mathbb{w}_k|\mathbb{x}) := N \cdot p^{\frac{1}{\beta}}(y|\mathbb{w}_k) \cdot s_k(\mathbb{x}) \\ p(\mathbb{w}_k) := \sum_{\mathbb{x}} p^*(\mathbb{w}_k|\mathbb{x}) \cdot p(\mathbb{x}) \\ p(j|\mathbb{w}_k) := \frac{1}{p(\mathbb{w}_k)} \cdot \sum_{\mathbb{x}} p(j|\mathbb{x}) \cdot p^*(\mathbb{w}_k|\mathbb{x}) \cdot p(\mathbb{x}) \\ \mathbb{w}_k := \mathbb{w}_k + \epsilon \cdot \beta \cdot \sum_{\mathbb{x}} \sum_t \frac{p^*(\mathbb{w}_t|\mathbb{x})}{s_t(\mathbb{x})} \cdot \frac{\partial s_t(\mathbb{x})}{\partial \mathbb{w}_k} \end{cases} \tag{10}$$

ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

Note that, unlike (2), in (10) we used "equal by definition" symbol := to drop the time step indicator d of the quantities in the iteration,

The parameter β controls the level of compression during learning. When β is small the compression rate is higher, since all data points $\mathbb{x}$ with class y tend to be mapped to more dominant prototypes $\mathbb{w}_k$ that have higher value of $p(y|\mathbb{w}_k)$. Consequently, some components may become obsolete, since they do not receive much assignment from the input space based on the optimal assignment rule (6). The average assignment for components is calculated in the parameter $p(\mathbb{w}_t)$ in (10). Therefore, during learning, a component $\mathbb{w}_t$ may be discarded from the set of components W if the value of $p(\mathbb{w}_t)$ is lower than a specific threshold, that we call the pruning threshold θ. On the other hand, as β goes to infinity, almost all the components are used for prediction. Also, the optimal assignment becomes $\lim_{\beta\to\infty} p^*(\mathbb{w}_k|\mathbb{x}) = s_k(\mathbb{x})$, which is substituted in (3) to have the practical predictor of ITLVQ as $p_W(j|\mathbb{x}) = \sum_t s_t(\mathbb{x}) \cdot p(j|\mathbb{w}_t)$. Therefore, in practice one needs to start from a small β and slowly increase β to a relatively large number during training. Note that the practical predictor is a manifestation of Recognition-by-Components [10]. Basically, $\mathbb{w}_t$ is a basic component and representation level $s_t(\mathbb{x})$ determines how much of the component is observed in $\mathbb{x}$. Then, the correlation $p(j|\mathbb{w}_t)$ is evaluated as a way to measure a correlation between class j and component $\mathbb{w}_t$.

4 Experiments and Simulations

We apply ITLVQ to the Two-Moons data set. We start with 20 randomly initialized components. The trade-off β starts at 0.1 and it is increased slowly every iteration as long as the value of β remains less than 5. We used the following representation function: $s_k(\mathbb{x}) = \frac{\exp(-\sigma \cdot ||\mathbb{x}-\mathbb{w}_k||^2)}{\sum_t \exp(-\sigma \cdot ||\mathbb{x}-\mathbb{w}_t||^2)}$ with $\sigma = 5$. The pruning threshold is set to $\theta = \frac{0.5}{m}$, where m is the number of remaining components at the time. After 50 iterations, we are often left with 4 components. The result is shown in the left image of figure 1, where components (blue dots) 1 and 2 represent the purple class and components 3 and 4 represent the yellow class. The classification accuracy on test data is 100%. The right image in figure 1 is the result of setting the initial $\beta = 1$ and $\theta = \frac{0.3}{m}$. The classification accuracy is 100% and we are often left with 9 to 13 components in the end. Then, we applied ITLVQ to MNIST data set. For $\sigma = 15$, 40 initial components, and initial $\beta = 0.2$, we achieve an average 89.5% classification accuracy with 10 final components. The result is similar to CbC without CNN feature extractor. The pruning threshold is $\theta = \frac{0.5}{m}$. To increase the accuracy, we used Siamese networks with CNN to learn a suitable representation function $s_t(.)$. The CNN consists of two layers of CNN: the first one has 64 filters of size 5×5 and the second has 32 filters of size 5×5 and both layers use tangent hyperbolic activation function. Also, both layers use 2×2 pooling grid. Since we are using deep networks as representation function, we will have to drop the component update during learning (last line in (10)), because the gradient of the learnt $s_t(\mathbb{x})$ with respect to $\mathbb{w}_k$ has to be numerically estimated and it is computationally

ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

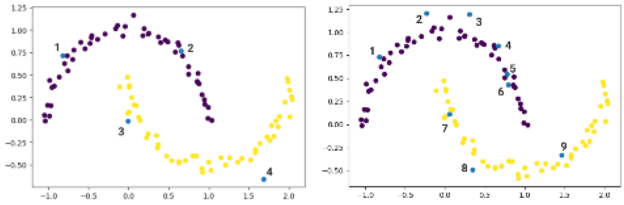

Fig. 1: Two Moons Results

hard. Instead, we start with 10 random samples for each class from training set as components and drop components with $p(\mathbb{w}_t)$ less than threshold $\theta = \frac{0.2}{m}$. We also have $\beta = 0.005$ and slowly increase it. After 10 rounds of training, we often end up with 10 to 15 components and an average accuracy of 99.1%. This result is an improvement when compared to an average of 97.9% accuracy that is achieved by the same setting, with the difference that 1 component is randomly chosen and fixed for each class.

5 Conclusion

In this paper, we defined LVQ in framework of IBT for classification purposes. This approach helps LVQ to extract most effective components for classification. Different levels of compression can be achieved using a trade-off parameter between compression and representation.

References

[1] Teuvo Kohonen and Teuvo Kohonen. Learning vector quantization. *Self-organizing maps*, pages 175–189, 1995.

[2] David Nova and Pablo A Estévez. A review of learning vector quantization classifiers. *Neural Computing and Applications*, 25:511–524, 2014.

[3] Petra Schneider, Tina Geweniger, Frank-Michael Schleif, Michael Biehl, and Thomas Villmann. Multivariate class labeling in robust soft lvq. In *ESANN*, 2011.

[4] Sascha Saralajew, Lars Holdijk, Maike Rees, Ebubekir Asan, and Thomas Villmann. Classification-by-components: Probabilistic modeling of reasoning over a set of components. *Advances in Neural Information Processing Systems*, 32, 2019.

[5] Mehrdad Mohannazadeh Bakhtiari and Thomas Villmann. Modification of the classification-by-component predictor using dempster-shafer-theory. In *Advances in Self-Organizing Maps, Learning Vector Quantization, Clustering and Data Visualization: Dedicated to the Memory of Teuvo Kohonen/Proceedings of the 14th International Workshop, WSOM+ 2022, Prague, Czechia, July 6-7, 2022*, pages 41–52. Springer, 2022.

[6] Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

[7] Artemy Kolchinsky, Brendan D Tracey, and David H Wolpert. Nonlinear information bottleneck. *Entropy*, 21(12):1181, 2019.

[8] DJ Strouse and David J Schwab. The deterministic information bottleneck. *Neural computation*, 29(6):1611–1630, 2017.

[9] Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[10] Irving Biederman. Recognition-by-components: a theory of human image understanding. *Psychological review*, 94(2):115, 1987.