Large Margin Learning Vector Quantization

Benjamin Paaßen

CITEC, Bielefeld University

Abstract

Learning vector quantization approaches are advantageous for their online learning capability, their low memory requirements, interpretable model, easy extensibility to multi-class problems, and their fast decision function [6]. However, a key drawback is the highly non-liner and non-convex loss of most LVQ variants, making optimization at times numerically difficult and prone to local optima. We suggest to slightly vary the usual LVQ loss in order to get closer to a convex setup. In particular, we propose to adapt the Large Margin Nearest Neighbor (LMNN) loss of Weinberger and Saul [9] to learning vector quantization models.

We obtain a dissimilarity version of large margin LVQ as a non-convex quadratic program and a kernel version as a convex quadratic program, the latter being analogous to the multi-prototype support vector machine developed by Aiolli and Sperduti [1]. First empiric results suggest that further work is required to make large margin LVQ useful in practice.

Assume we wish to learn K prototypes $w_1, \ldots, w_K \in \mathcal{X}$ with labels $z_1, \ldots, z_K \in \{1, \ldots, L\}$, such that as many data points $x_1, \ldots, x_m \in \mathcal{X}$ with labels $y_1, \ldots, y_m \in \{1, \ldots, L\}$ are correctly classified by a one-nearest neighbor assignment, i.e. data point x_i is assigned the label of the closest prototype. Then, data point x_i is classified *correctly* if and only if the closest prototype has the correct label. Now, let $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be some dissimilarity on $\mathcal{X}$, let $d_{i,l} = d(x_i, w_l)^2$ be the (squared) dissimilarity to to the lth prototype, let $d_i^+ := \min_{k:z_k=y_i} d_{i,k}$ be the (squared) dissimilarity to the closest prototype with the correct label, and let $I_i := \{l | y_i \neq z_l\}$ be the set of prototype indices with different label than the ith data point. Then, the prototype-based LMNN loss for our problem is

$$\ell(w_1, \ldots, w_K) = \overbrace{\sum_{i=1}^m d_i^+}^{\text{pull}} + \frac{1}{2C} \cdot \overbrace{\sum_{i=1}^m \sum_{l \in I_i} \text{ReLU}\big(d_i^+ - d_{i,l} + \gamma\big)^2}^{\text{push}} \tag{1}$$

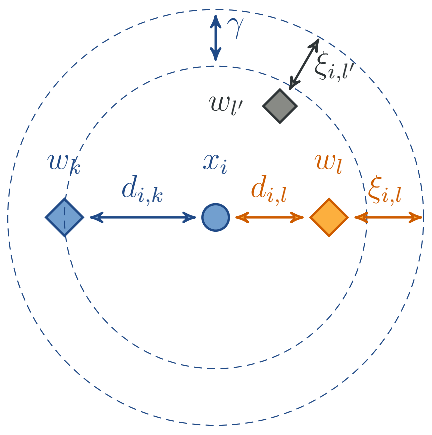

Figure 1: An illustration of the large-margin learning vector quantization loss 1. From the perspective of data point x_i (blue circle), the closest prototype with the same label is w_k (blue diamond). The pull loss tries to move w_k closer to x_i. Further, w_l (orange diamond) is closer to x_i and has a different label, which would mean that x_i would be misclassified, which punish via the push loss. Moreover, we punish all prototypes that are within a margin of safety of γ (outer dashed circle), e.g. $w_{l'}$ (grey diamond). The contribution to the push loss is $\xi_{i,l}$ and $\xi_{i,l'}$ respectively, i.e. the *slack* that is needed to satisfy margin constraints.

where $\text{ReLU} : \mathbb{R} \to \mathbb{R}^+$ is the rectified linear unit aka the hinge loss, defined as $\text{ReLU}(\mu) = \max\{0, \mu\}$. We call this loss the *large margin learning vector quantization* (LM-LVQ) loss.

Intuitively, the first term pulls prototypes closer to data points in their receptive field and the second term pushes prototypes with wrong labels away if they invade a margin of safety around the data point. Also refer to Figure 1 for an illustration of this loss. The hyper-parameter $C \in \mathbb{R}^+$ weighs between the push and pull forces.

Note that the push force can be seen as a variant of generalized learning vector quantization [8] with the nonlinearity $\Phi(\mu) = \text{ReLU}(\mu + \gamma)^2$, albeit without the dissimilarity normalization.

Also note that, in contrast to LMNN, we square the push loss contributions, which has the advantage that the gradient is smooth at point 0 and supports our later optimization steps.

To optimize loss 1, we re-write it first as an optimization problem with slack variables $\xi_{i,l}$, which express how far prototype w_l invades the margin of data point i.

$$\begin{aligned} \min_{\substack{w_1,\ldots,w_K,\\ \Xi\in\mathbb{R}^{m\times K}}} \quad & \frac{1}{2\cdot C}\sum_{i=1}^{m}\sum_{l\in I_i}\xi_{i,l}^2 + \sum_{i=1}^{m} d_i^+ && (2)\\ \text{s.t.} \quad & \xi_{i,l} \geq d_i^+ - d_{i,l} + \gamma & \forall i\in\{1,\ldots,m\} \quad \forall l \in I_i\\ & \xi_{i,l} \geq 0 & \forall i\in\{1,\ldots,m\} \quad \forall l\in\{1,\ldots,K\} \end{aligned}$$

Note that this problem is a quadratically constrained quadratic program [2]. Unfortunately, this program is not convex due to the negative sign in front of $d_{i,l}$. Still, we can make a solution attempt by constructing the Wolfe dual of this problem.

Disregarding the non-negativity constraints for the slack variables for the moment, we obtain the following Lagrangian.

$$\mathcal{L}(\boldsymbol{W},\Xi,\boldsymbol{\Lambda}) = \frac{1}{2\cdot C}\sum_{i=1}^{m}\sum_{l\in I_i}\xi_{i,l}^2 + \sum_{i=1}^{m} d_i^+ - \sum_{i=1}^{m}\sum_{l\in I_i}\lambda_{i,l}\cdot\Big(\xi_{i,l} + d_{i,l} - d_i^+ - \gamma\Big)$$

Let now P_k be the set of all points x_i to which w_k is the closest prototype with the same label, and let $N_k = \{j|y_j \neq z_k\}$ be the set of data points x_j with a different label than w_k. We also call these sets the *positive* and *negative receptive field* of prototype w_k. Then, we can re-write the Lagrangian as follows.

$$\begin{aligned} \mathcal{L}(\boldsymbol{W},\Xi,\boldsymbol{\Lambda}) =& \sum_{k=1}^{K}\sum_{i\in P_k}\big(1+\sum_{l\in I_i}\lambda_{i,l}\big)\cdot d_{i,k} + \sum_{i\in N_k}(-\lambda_{i,k})\cdot d_{i,k} && (3)\\ &+\frac{1}{2\cdot C}\sum_{i=1}^{m}\sum_{k\in I_i}\xi_{i,k}^2 - \sum_{i=1}^{m}\sum_{k\in I_i}\lambda_{i,k}\cdot\xi_{i,k} + \gamma\cdot\sum_{i=1}^{m}\sum_{k\in I_i}\lambda_{i,k} \end{aligned}$$

We further introduce the auxiliary variables $\beta_{k,i}$, defined as $1+\sum_{l\in I_i}\lambda_{i,l}$ if $i\in P_k$, as $-\lambda_{i,k}$ if $i\in N_k$, and as zero otherwise. Accordingly, the first line of the Lagrangian simplifies to $\sum_{k=1}^{K}\sum_{i=1}^{m}\beta_{k,i}\cdot d_{i,k}$.

Our next step is to compute the gradient of our Lagrangian with respect to the optimization variables and set it to zero. To do so, we need to make an assumption about the underlying data space. In particular, we assume that the dissimilarity d is self-equal and symmetric, i.e. for any two data points $x, y\in\mathcal{X}$ we obtain $d(x,x) = 0$ and $d(x,y) = d(y,x)$. Under these assumptions, Pekalska and Duin [7] guarantee a *pseudo-Euclidean embedding*, i.e. there exist two mappings, $\phi^+ : \mathcal{X}\to\mathbb{R}^n$ and $\phi^- : \mathcal{X}\to\mathbb{R}^n$ from the data space into $\mathbb{R}^n$ for some n, such that for any $x, y\in\mathcal{X}$ it holds:

$$d(x,y)^2 = ||\phi^+(x)-\phi^+(y)||^2 - ||\phi^-(x)-\phi^-(y)||^2$$

Accordingly, we obtain the following gradient of the Lagrangian with respect to $\phi^+(w_k)$ and $\phi^-(w_k)$:

$$\nabla_{\phi^+(w_k)}\mathcal{L}(\boldsymbol{W}, \Xi, \boldsymbol{\Lambda}) = \sum_{i=1}^{m} \beta_{k,i} \cdot 2 \cdot \big(\phi^+(w_k) - \phi^+(x_i)\big)$$
$$\nabla_{\phi^-(w_k)}\mathcal{L}(\boldsymbol{W}, \Xi, \boldsymbol{\Lambda}) = \sum_{i=1}^{m} \beta_{k,i} \cdot (-2) \cdot \big(\phi^-(w_k) - \phi^-(x_i)\big)$$

Setting these to zero yields:

$$\phi^+(w_k) = \frac{\sum_{i=1}^m \beta_{k,i} \cdot \phi^+(x_i)}{\sum_{i=1}^m \beta_{k,i}}, \quad \text{and} \quad \phi^-(w_k) = \frac{\sum_{i=1}^m \beta_{k,i} \cdot \phi^-(x_i)}{\sum_{i=1}^m \beta_{k,i}}$$

In other words, the prototype is an affine combination of data points. This, in turn, lets us re-write the squared dissimilarities $d_{i,k}$ as follows [4]:

$$d_{i,k} = \frac{\sum_{j=1}^m \beta_{k,j} \cdot d(x_i, x_j)^2}{\sum_{j=1}^m \beta_{k,j}} - \frac{\vec{\beta}_k^T \cdot \boldsymbol{D}^2 \cdot \vec{\beta}_k}{2 \cdot \big(\sum_{j=1}^m \beta_{k,j}\big)^2}$$

where $\boldsymbol{D}$ is the matrix of all squared pairwise dissimilarities $d(x_i, x_j)^2$.

Finally, consider the derivative of our Lagrangian with respect to $\xi_{i,k}$:

$$\frac{\partial}{\partial \xi_{i,l}}\mathcal{L}(\boldsymbol{W}, \Xi, \boldsymbol{\Lambda}) = \frac{1}{C} \cdot \xi_{i,l} - \lambda_{i,l} \overset{!}{=} 0 \qquad \Longleftrightarrow \qquad \xi_{i,l} = C \cdot \lambda_{i,l}$$

Note that this result ensures that the non-negativity constraints for $\xi_{i,l}$ hold because we already have non-negativity constraints for the Lagrange multipliers $\lambda_{i,l}$.

Plugging these results back into our Lagrangian 3, we obtain:

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\Lambda}) =& \sum_{k=1}^{K} \sum_{i=1}^{m} \beta_{k,i} \cdot \Big(\frac{\sum_{j=1}^m \beta_{k,j} \cdot d(x_i, x_j)^2}{\sum_{j=1}^m \beta_{k,j}} - \frac{\vec{\beta}_k^T \cdot \boldsymbol{D}^2 \cdot \vec{\beta}_k}{2 \cdot \big(\sum_{j=1}^m \beta_{k,j}\big)^2} \Big) \\
&+ \frac{1}{2 \cdot C} \sum_{i=1}^{m} \sum_{k \in I_i} C^2 \cdot \lambda_{i,k}^2 - \sum_{i=1}^{m} \sum_{k \in I_i} \lambda_{i,k} \cdot C \cdot \lambda_{i,k} + \gamma \cdot \sum_{i=1}^{m} \sum_{k \in I_i} \lambda_{i,k} \\
=& \sum_{k=1}^{K} \Big(\frac{\sum_{i=1}^m \sum_{j=1}^m \beta_{k,i} \cdot \beta_{k,j} \cdot d(x_i, x_j)^2}{\sum_{j=1}^m \beta_{k,j}} \Big) - \Big(\frac{\vec{\beta}_k^T \cdot \boldsymbol{D}^2 \cdot \vec{\beta}_k}{2 \cdot \big(\sum_{j=1}^m \beta_{k,j}\big)^2} \cdot \sum_{i=1}^m \beta_{k,i} \Big) \\
&- \frac{C}{2} \sum_{i=1}^{m} \sum_{k \in I_i} \lambda_{i,k}^2 + \gamma \cdot \sum_{i=1}^{m} \sum_{k \in I_i} \lambda_{i,k} \\
=& \frac{1}{2} \sum_{k=1}^{K} \frac{\vec{\beta}_k^T \cdot \boldsymbol{D}^2 \cdot \vec{\beta}_k}{\sum_{j=1}^m \beta_{k,j}} - \frac{C}{2} \sum_{i=1}^{m} \sum_{k \in I_i} \lambda_{i,k}^2 + \gamma \cdot \sum_{i=1}^{m} \sum_{k \in I_i} \lambda_{i,k}
\end{aligned}$$

Accordingly, we obtain the following Wolfe dual form.

$$\begin{aligned}
\min_{\boldsymbol{\Lambda} \in \mathbb{R}^{m \times K}} \quad & -\frac{1}{2} \sum_{k=1}^{K} \frac{\vec{\beta}_k^T \cdot \boldsymbol{D} \cdot \vec{\beta}_k}{\sum_{j=1}^m \beta_{k,j}} + \frac{C}{2} \sum_{i=1}^{m} \sum_{k \in I_i} \lambda_{i,k}^2 - \gamma \cdot \sum_{i=1}^{m} \sum_{k \in I_i} \lambda_{i,k} \qquad (4) \\
\text{s.t.} \quad & \lambda_{i,k} \geq 0, \quad \beta_{k,i} = \begin{cases} 1 + \sum_{l \in I_i} \lambda_{i,l} & \text{if } i \in P_k \\ -\lambda_{i,k} & \text{if } i \in N_k \\ 0 & \text{otherwise} \end{cases} \qquad \begin{array}{l} \forall k \in \{1, \ldots, K\}, \\ \forall i \in \{1, \ldots, m\} \end{array}
\end{aligned}$$

Note that solving the Wolfe dual does *not* guarantee an optimal solution for the primal problem due to non-convexity [2]. Still, we can hope to achieve *good* solutions, given that the dual in itself has a rather intuitive interpretation: If all Lagrange multipliers are zero, prototypes gracefully degenerate to the means of their receptive fields. If this is sufficient to classify the data set correctly, this is the single optimal solution. However, if any prototype l invades the margin of a data point i, the Lagrange multiplier $\lambda_{i,l}$ must be increased, which means that prototype l moves *away* from data point i, whereas this multiplier is *added* to $\beta_{k,i}$ for the closest correct prototype k, such that this prototype moves *closer* to data point i, resembling the typical LVQ behavior.

Another problem is that the dual is numerically problematic due to the sum over all $\beta_{k,i}$ coefficients in the denominator of the quadratic term. Fortunately, this can be addressed by imposing that $\sum_{i=1}^m \beta_{k,i} = |P_k|$ for all k, i.e. that the sum of all β coefficients should sum to the size of the receptive field. This side constraint enforces that the sum of all coefficients stays the same as in its initial state when all Lagrange multipliers are zero. Interestingly, this side constraint is also equivalent to introducing bias terms (we omit the full derivation here for brevity).

We further note that this dual can be transformed into a standard quadratic problem by expressing the vectors $\vec{\beta}_k$ as a (sparse) affine transformation from the Lagrange multipliers. In particular, we can re-write $\vec{\beta}_k = \boldsymbol{A_k} \cdot \vec{\lambda} + \vec{1}_{P_k}$, where $\vec{\lambda}$ is the concatenation of all rows in $\boldsymbol{\Lambda}$, where $a_{k,i,(i-1)\cdot K+k} = -1$ if $i \in N_k$, $a_{k,i,(i-1)\cdot K+l} = +1$ if $i \in P_k$ and $l \in I_i$ and zero otherwise, and where $\vec{1}_{P_k}$ is a m-dimensional vector which is 1 at entries $i \in P_k$ and zero otherwise. Then, problem 4 becomes:

$$\begin{aligned}
\min_{\vec{\lambda} \in \mathbb{R}^{m \cdot K}} \quad & \frac{1}{2} \vec{\lambda}^T \cdot \Big(C \cdot \boldsymbol{I} - \sum_{k=1}^{K} \boldsymbol{A_k}^T \cdot \frac{\boldsymbol{D}}{|P_k|} \cdot \boldsymbol{A_k} \Big) \cdot \vec{\lambda} - \Big(\gamma \cdot \vec{1}^T + \sum_{k=1}^{K} \vec{1}_{P_k}^T \cdot \frac{\boldsymbol{D}}{|P_k|} \cdot \boldsymbol{A_k} \Big) \cdot \vec{\lambda} \\
\text{s.t.} \quad & \vec{\lambda} \geq 0 \qquad (5) \\
& \vec{1}^T \cdot \boldsymbol{A_k} \cdot \vec{\lambda} = 0 \qquad \forall k \in \{1, \ldots, K\}
\end{aligned}$$

Still, this quadratic problem is not necessarily convex because $\boldsymbol{D}$ is indefinite, yielding a convex shape in some search directions and a concave shape in other directions. The regularization term and the equality side constraints may ensure a convex shape, but this appears to not be guaranteed in general. Still, we expect a more smooth loss function compared to relational GLVQ [5] due to the absent dissimilarity normalization.

Finally, we note that our problem formulation relies on the assumption that the positive receptive fields P_k stay fixed, similar to LMNN. However, we can relax this assumption by running a multi-pass scheme where the positive receptive fields are updated after each optimization of problem 5 until the fields do not change anymore. Because any update in positive receptive field is guaranteed to reduce the loss, this scheme is guaranteed to converge, similar to Multi-pass LMNN [3].

If we wish to guarantee proper convexity, we can switch from a dissimilarity formulation to a kernel formulation. In that case, our primal loss changes to:

$$\ell(w_1, \ldots, w_K) = \sum_{i=1}^{m} d_i^+ + \frac{1}{2C} \cdot \sum_{l \in I_i} \mathrm{ReLU}\big(s_{i,l} - s_i^+ + \gamma\big)^2 \tag{6}$$

where $s_{i,l}$ is the kernel value between data point i and prototype l and d_i^+ refers to the dissimilarity between data point i and the closest prototype with the same label, where the dissimilarity measure is given as $d(x,y)^2 = s(x,x) - 2s(x,y) + s(y,y)$. Via a very similar derivation as before, we obtain the following Wolfe dual:

$$\begin{aligned}
\min_{\vec{\lambda} \in \mathbb{R}^{m \cdot K}} \quad & \frac{1}{2}\vec{\lambda}^T \cdot \Big(\sum_{k=1}^{K} \boldsymbol{A_k}^T \cdot \frac{\boldsymbol{S}}{|P_k|} \cdot \boldsymbol{A_k} + C \cdot \boldsymbol{I}\Big) \cdot \vec{\lambda} + \Big(\sum_{k=1}^{K} \vec{1}_{P_k}^T \cdot \frac{\boldsymbol{S}}{|P_k|} \cdot \boldsymbol{A_k} - \gamma \cdot \vec{1}^S\Big) \cdot \vec{\lambda} \\
\text{s.t.} \quad & \vec{\lambda} \geq 0 \\
& \vec{1}^T \cdot \boldsymbol{A_k} \cdot \vec{\lambda} = 0 \qquad \forall k \in \{1, \ldots, K\}
\end{aligned} \tag{7}$$

where $\boldsymbol{S}$ is the matrix of pairwise kernel values $s(x_i, x_j)$. This problem is a standard convex quadratic program with linear side constraints. Indeed, the kernel matrix may even be slightly indefinite because the regularization term $C \cdot \boldsymbol{I}$ applies an implicit shift eigenvalue correction.

This latter form is almost equivalent to the multi-prototype SVM suggested by Aiolli and Sperduti [1], except for the regularization, which here pushes prototypes to the center of the receptive field, whereas the multi-prototype SVM pushes prototypes toward the origin.

Closing Remarks: Merging LMNN and LVQ concepts appears promising from these first attempts. In particular, we obtained a non-convex, but "wellbehaved" dissimilarity formulation for pseudo-Euclidean dissimilarities, and a convex quadratic program formulation for kernels. Unfortunately, first empiric experiments suggest that the approach may not be able to compete with a simple one-versus-one SVM. Likely, further improvements are required - be it in terms of concept or optimization - in order to make this approach useful in practical applications. Still, large margin LVQ offers a conceptual bridge, shedding further light on the strong relations between LVQ, LMNN, and SVMs.

References

[1] Fabio Aiolli and Alessandro Sperduti. "Multiclass classification with multi-prototype support vector machines". In: *Journal of Machine Learning Research* 6 (2005), pp. 817–850. URL: `http://www.jmlr.org/papers/volume6/aiolli05a/aiolli05a.pdf`.

[2] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004. URL: `https://web.stanford.edu/~boyd/cvxbook/`.

[3] Christina Göpfert, Benjamin Paaßen, and Barbara Hammer. "Convergence of Multi-pass Large Margin Nearest Neighbor Metric Learning". In: *Proceedings of the 25th International Conference on Artificial Neural Networks (ICANN 2016)*. (Barcelona, Spain). Ed. by Alessandro E.P. Villa, Paolo Masulli, and Antonio Javier Pons Rivero. Vol. 9886. Springer Nature, Aug. 2016, pp. 510–517. DOI: `10.1007/978-3-319-44778-0_60`. URL: `https://pub.uni-bielefeld.de/record/2905729`.

[4] Barbara Hammer and Alexander Hasenfuss. "Topographic Mapping of Large Dissimilarity Data Sets". In: *Neural Computation* 22.9 (2010), pp. 2229–2284. DOI: `10.1162/NECO_a_00012`.

[5] Barbara Hammer et al. "Learning vector quantization for (dis-)similarities". In: *Neurocomputing* 131 (2014), pp. 43–51. DOI: `10.1016/j.neucom.2013.05.054`.

[6] David Nova and Pablo A. Estévez. "A review of learning vector quantization classifiers". In: *Neural Computing and Applications* 25.3 (2014), pp. 511–524. DOI: `10.1007/s00521-013-1535-3`. URL: `https://arxiv.org/abs/1509.07093`.

[7] Elzbieta Pekalska and Robert Duin. *The Dissimilarity Representation for Pattern Recognition: Foundations And Applications (Machine Perception and Artificial Intelligence)*. River Edge, NJ, USA: World Scientific Publishing Co., Inc., 2005. ISBN: 9812565302.

[8] Atshushi Sato and Keiji Yamada. "Generalized Learning Vector Quantization". In: *Proceedings of the 7th conference on Advances in Neural Information Processing Systems (NIPS 1995)*. Ed. by G. Tesauro, D. Touretzky, and T. Leen. Cambridge, MA, 1995, pp. 423–429. URL: `https://papers.nips.cc/paper/1113-generalized-learning-vector-quantization`.

[9] Kilian Weinberger and Lawrence Saul. "Distance Metric Learning for Large Margin Nearest Neighbor Classification". In: *Journal of Machine Learning Research* 10 (2009), pp. 207–244. URL: `http://www.jmlr.org/papers/v10/weinberger09a.html`.