Notes on Asymmetric Metric Learning for k-NN Classification

Shubhendu Trivedi
Toyota Technological Institute
Chicago, IL 60637
shubhendu@{cs.uchicago or ttic}.edu

Abstract

Running notes on Asymmetric Similarity Learning

1 Introduction

Suppose we are given N training examples $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ and their class labels $\mathbf{y} = [y_1, \ldots, y_n]^\top$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in [R]$. Let $\Delta(r, r') \geq 0$ be the classification loss where $\Delta(r, r) = 0$ for any $r \in [R]$. In the problem of Metric Learning as usually stated, we are interested in the Mahalanobis Metrics:

$$D_{\mathbf{W}}(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x} - \mathbf{x}_i)^T \mathbf{W} (\mathbf{x} - \mathbf{x}_i) \tag{1}$$

The Mahalanobis metrics are characterized by $d \times d$ positive semidefinite matrices $\mathbf{W}$. Since $\mathbf{W}$ is psd, it can be factorized as $\mathbf{W} = \mathbf{L}^T \mathbf{L}$, and thus:

$$D_{\mathbf{W}}(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x} - \mathbf{x}_i)^T \mathbf{L}^T \mathbf{L} (\mathbf{x} - \mathbf{x}_i) = (\mathbf{L}\mathbf{x} - \mathbf{L}\mathbf{x}_i)^T (\mathbf{L}\mathbf{x} - \mathbf{L}\mathbf{x}_i) = \|\mathbf{L}(\mathbf{x} - \mathbf{x}_i)\|_2^2 \tag{2}$$

That is, the Mahalanobis Metrics are equivalent to learning a linear embedding $\mathbf{x} \rightarrow \mathbf{L}\mathbf{x}$ of the data. These are convenient as they allow access to a large set of methods that can be used for fast metric search.

More generally, one might also consider non-linear embeddings $\mathbf{x} \rightarrow \phi(\mathbf{x}; w)$, and consider the distance in the embedded metric space:

$$D(\mathbf{x}, \mathbf{x}_i) = \|\phi(\mathbf{x}) - \phi(\mathbf{x}_i)\|_2^2 \tag{3}$$

The mapping may be learned discriminatively to optimize for the k-NN performance in the embedded space. This is usually done so as to satisfy pairwise, triplet, or more general ranking constraints so as to improve kNN classification performance.

Another approach to this problem might be taken, which is arguably different in the sense that all the points are not mapped to the same space. That is the query point and the database points are (linearly)projected into different spaces, thus making the distance computation asymmetric (as contrasted to 2).

$$D_{\mathbf{W}}(\mathbf{x}, \mathbf{x}_i) = \|\mathbf{U}\mathbf{x} - \mathbf{V}\mathbf{x}_i\|_2^2 = (\mathbf{U}\mathbf{x} - \mathbf{V}\mathbf{x}_i)^T (\mathbf{U}\mathbf{x} - \mathbf{V}\mathbf{x}_i) \tag{4}$$

The above may be rewritten as follows:

$$D_{\mathbf{W}}(\mathbf{x}, \mathbf{x}_i) = \begin{bmatrix} \mathbf{x} & \mathbf{x}_i \end{bmatrix} \begin{bmatrix} \mathbf{U}^T\mathbf{U} & -\mathbf{U}^T\mathbf{V} \\ -\mathbf{V}^T\mathbf{U} & \mathbf{V}^T\mathbf{V} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{x}_i \end{bmatrix} \tag{5}$$

Thus, the problem of learning $\mathbf{U}$ and $\mathbf{V}$ as specified in the distance formulation of 4 is equivalent to learning a matrix $\mathbf{W} \in \mathbb{R}^{2d \times 2d}$ such that $\mathbf{W} \succeq 0$

Given $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{d \times d}$, for any $h \in \mathbf{X}$, we can define the similarity between $\mathbf{x}$ and h as:

$$S_{\mathbf{U},\mathbf{V}}(\mathbf{x}, h) = -\sum_{\mathbf{x}_i \in h} (\mathbf{U}\mathbf{x} - \mathbf{V}\mathbf{x}_i)^T (\mathbf{U}\mathbf{x} - \mathbf{V}\mathbf{x}_i) \tag{6}$$

Using the above measure of similarity we can define the following surrogate loss similar to [1]

$$L(\mathbf{x}, y, \{\mathbf{U}, \mathbf{V}\}) = \max_h \left[S_{\mathbf{U},\mathbf{V}}(x, h) + \Delta(y, h)\right] - \max_{h:\Delta(y,h)=0} S_{\mathbf{U},\mathbf{V}}(\mathbf{x}, h) \tag{7}$$

For capacity control we penalize for the Frobenius norm of the full matrix $\mathbf{W}$. The objective becomes:

$$\min_{\mathbf{U},\mathbf{V}} \|\mathbf{W}\|_F + C \sum_i L(\mathbf{x}_i, \mathbf{y}_i, \{\mathbf{U}, \mathbf{V}\}) \tag{8}$$

Next we write down the derivatives of the loss and the regularizer with respect to $\mathbf{U}$ (query gradient) and $\mathbf{V}$ (database gradient):

$$\frac{\partial S_{\mathbf{U},\mathbf{V}}(\mathbf{x}, h)}{\partial \mathbf{U}} = -2 \left(\sum_{\mathbf{x}_i \in h} \mathbf{U}(\mathbf{x}\mathbf{x}^T) - (\mathbf{V}\mathbf{x}_i)\mathbf{x}^T \right) \tag{9}$$

$$\frac{\partial S_{\mathbf{U},\mathbf{V}}(\mathbf{x}, h)}{\partial \mathbf{V}} = -2 \left(\sum_{\mathbf{x}_i \in h} \mathbf{V}(\mathbf{x}_i\mathbf{x}_i^T) - (\mathbf{U}\mathbf{x})\mathbf{x}_i^T \right) \tag{10}$$

$$\frac{\partial \|W\|_F}{\partial \mathbf{U}} = 2(\mathbf{U}\mathbf{U}^T)\mathbf{U} + 4(\mathbf{V}\mathbf{V}^T)\mathbf{U} \tag{11}$$

$$\frac{\partial \|W\|_F}{\partial \mathbf{V}} = 2(\mathbf{V}\mathbf{V}^T)\mathbf{V} + 4(\mathbf{U}\mathbf{U}^T)\mathbf{V} \tag{12}$$

Other regularizers that might also be well motivated are $\|\mathbf{U}^T\mathbf{U}\|_F$ or $\|\mathbf{U}\|_F$ in the update equation for $\mathbf{U}$ and $\|\mathbf{V}^T\mathbf{V}\|_F$ or $\|\mathbf{V}\|_F$ in the update equation for $\mathbf{V}$,

2 Optimization

Each iteration t of the algorithm consists of three steps:

1. Targeted inference of h_i^* for each sample $\mathbf{x}_i$:
$$h_i^* = \operatorname*{argmax}_{h:\Delta(y_i,h)=0} S_{\mathbf{U}^{(t)},\mathbf{V}^{(t)}}(\mathbf{x}_i, h) \tag{13}$$
This can be done by algorithm 2 in [1].
2. Loss augmented inference of $\hat{h}_i$ for each sample $\mathbf{x}_i$:
$$\hat{h}_i = \operatorname*{argmax}_h \left[S_{\mathbf{U}^{(t)},\mathbf{V}^{(t)}}(\mathbf{x}_i, h) + \Delta(y_i, h)\right] \tag{14}$$
3. Gradient updates for $\mathbf{U}$ and $\mathbf{V}$.
$$\mathbf{U}^{(t+1)} = \mathbf{U}^{(t)} - \eta^{(t)} \left[\frac{\partial S_{\mathbf{U},\mathbf{V}}(\mathbf{x}, \hat{h}_i)}{\partial \mathbf{U}} - \frac{\partial S_{\mathbf{U},\mathbf{V}}(\mathbf{x}, h_i^*)}{\partial \mathbf{U}} + 2(\mathbf{U}\mathbf{U}^T)\mathbf{U} + 4(\mathbf{V}\mathbf{V}^T)\mathbf{U} \right]$$
$$\mathbf{V}^{(t+1)} = \mathbf{V}^{(t)} - \eta^{(t)} \left[\frac{\partial S_{\mathbf{U},\mathbf{V}}(\mathbf{x}, \hat{h}_i)}{\partial \mathbf{V}} - \frac{\partial S_{\mathbf{U},\mathbf{V}}(\mathbf{x}, h_i^*)}{\partial \mathbf{V}} + 2(\mathbf{V}\mathbf{V}^T)\mathbf{V} + 4(\mathbf{U}\mathbf{U}^T)\mathbf{V} \right]$$

The above can also be done similarly for the case of different non-linear maps for database and query points.

References

[1] Shubhendu Trivedi, David Mcallester, and Gregory Shakhnarovich. Discriminative metric learning by neighborhood gerrymandering. In *Advances in Neural Information Processing Systems*, pages 3392–3400, 2014.