LOGISTIC SIMILARITY METRIC LEARNING FOR FACE VERIFICATION

Lilei Zheng, Khalid Idrissi, Christophe Garcia, Stefan Duffner and Atilla Baskurt

Université de Lyon, CNRS, INSA-Lyon, LIRIS, UMR5205, F-69621, France

ABSTRACT

This paper presents a new method for similarity metric learning, called Logistic Similarity Metric Learning (LSML), where the cost is formulated as the logistic loss function, which gives a probability estimation of a pair of faces being similar. Especially, we propose to shift the similarity decision boundary gaining significant performance improvement. We test the proposed method on the face verification problem using four single face descriptors: LBP, OCLBP, SIFT and Gabor wavelets. Extensive experimental results on the LFW-a data set demonstrate that the proposed method achieves competitive state-of-the-art performance on the problem of face verification.

Index Terms— Metric learning, face verification, cosine similarity, face recognition, linear transformation

1. INTRODUCTION

The notion of pairwise metric between data points plays an important role in the area of pattern recognition and machine learning [1]. In the context of our work, the task of face verification aims at determining whether two face images are of the same person or not. Naturally, tremendous efforts have been put on studying metric learning methods to provide solutions for face verification [2, 3, 4, 5, 6, 7, 8, 9, 10, 11].

One can divide current metric learning methods into two main families: distance metric learning and similarity metric learning. Typically, most of the work in distance metric learning concerns the Mahalanobis distance [2, 3]: $d_M(x, y) = \sqrt{(x-y)^T M(x-y)}$, where x and y are two vectors, M is the matrix that needs to be learnt. Note that when M is the identity matrix, $d_M(x, y)$ is the Euclidean distance. In contrast, similarity metric learning methods learns similarity of the following form: $s_M(x, y) = x^T My/N(x, y)$, where $N(x, y)$ is a normalization term [12]. Specifically, when $N(x, y) = 1$, $s_M(x, y)$ is the bilinear similarity function [13]; when $N(x, y) = \sqrt{x^T Mx}\sqrt{y^T My}$, $s_M(x, y)$ is the generalized cosine similarity function [5]. Currently, work in metric learning has focused on the above linear models or their variants because they are more convenient to optimize and less prone to over-fitting. For instance, one of the best, the Within Class Covariance Normalization (WCCN) approach has shown its effectiveness on the problem of face verification [8]. Besides, a few approaches have investigated nonlinear metric learning [10, 14, 15]. These nonlinear methods have the potential to outperform linear methods on some problems, but are subject to local optima and more inclined to over-fit the training data [1].

Developing robust face descriptors is another crucial point for improving the verification performance. Popular face descriptors include eigenfaces [16], Gabor wavelets [17], SIFT [18], Local Binary Patterns (LBP) [19], etc. Recently, in contrast to these artificial face descriptors, much attention has been attracted to automatically extracting face representation using convolutional network thanks to the growing interest in deep learning in the last few years [20, 21].

In this paper, our work focuses on linear similarity metric learning methods. Compared with the classical Cosine Similarity Metric Learning (CSML) approach, we introduce the logistic loss function to measure the cost and produce a probability estimation of two faces being similar. More importantly, a parameter is used to shift the cosine similarity decision boundary. Experiments showed that an appropriate shifting parameter leads to significant performance improvement over the classical CSML method. We call our approach LSML, for Logistic Similarity Metric Learning.

To represent face images, besides the popular face descriptors such as Gabor wavelets [17], SIFT [18] and LBP [19], we use Over-Complete LBP [8] as another choice to improve the overall performance on face verification. We conducted experiments on the data set 'Labeled Faces in the Wild' (LFW) [22] comparing our method with the state-of-the-art methods, CSML [5] and WCCN [8]. All the experiments were performed under the LFW restricted configuration with label-free outside data. Results show that our method outperforms the classical CSML and WCCN, and achieves high performance on face verification.

The remaining sections are organized as follows. Section 2 introduces the classical CSML approach. Section 3 presents the proposed LSML. Experiments and analysis are reported in section 4. Finally, conclusions are drawn and perspectives are presented in section 5.

2. COSINE SIMILARITY METRIC LEARNING

In the task of face verification, two face images of the same person are called a similar pair; otherwise, two face images of

Thanks to the China Scholarship Council (CSC) for funding.

different persons are called a dissimilar pair or a different pair. By representing the face images as vectors, the verification of faces becomes a problem of measuring similarity between vectors.

Before introducing the CSML method, we present some important notations: a triplet (x_i, y_i, s_i) represents a pair of instances, where x_i and y_i are two vectors, and $s_i = 1$ (resp. -1) means that the two vectors are similar (resp. dissimilar). A linear metric learning method defines a linear transformation $f(z, A) = Az$ on the raw feature vectors and produces another triplet (a_i, b_i, s_i), where $a_i = f(x_i, A) = Ax_i$ and $b_i = f(y_i, A) = Ay_i$. The objective of CSML is employing this transformation to make similar vectors closer and separate dissimilar vectors: an ideal matrix A makes $cos(a_i, b_i) = 1$ for a pre-defined similar pair (x_i, y_i) while making $cos(a_i, b_i) = -1$ for a dissimilar pair, where the cosine similarity $cos(a_i, b_i)$ is:

$$cos(a_i, b_i) = \frac{a_i^T b_i}{\|a_i\|\|b_i\|}. \quad (1)$$

The cost function of CSML is [5]:

$$J_{CSML} = \frac{1}{n}\sum_{i=1}^{n} -s_i cos(a_i, b_i) + \frac{\lambda}{2}\|A - A_0\|^2, \quad (2)$$

with gradient function:

$$\begin{aligned}\frac{\partial J_{CSML}}{\partial A} &= \frac{1}{n}\sum_{i=1}^{n}\frac{\partial -s_i cos(a_i, b_i)}{\partial A} + \lambda(A - A_0) \\ &= \frac{1}{n}\sum_{i=1}^{n}\frac{s_i}{\|a_i\|\|b_i\|}[(\frac{a_i^T b_i}{\|a_i\|^2}a_i - b_i)x_i^T + (\frac{a_i^T b_i}{\|b_i\|^2}b_i - a_i)y_i^T] \\ &\quad + \lambda(A - A_0),\end{aligned} \quad (3)$$

where n is the number of all similar and dissimilar pairs from the training data, λ is the regularization parameter, and A_0 is any matrix that we want A to be regularized with: we set A to be A_0 before optimizing the cost; hence during the optimization, the larger the parameter λ is, the closer A is to A_0. Usually, we specify A_0 as the identity matrix.

3. LOGISTIC SIMILARITY METRIC LEARNING

Minimizing the CSML cost function (Equation (2)) implies making $cos(a_i, b_i) > 0$ for a similar pair and making $cos(a_i, b_i) < 0$ for a dissimilar pair at the same time. In other words, CSML sets 0 as the decision boundary for this binary decision problem. However, in a limited space which contains a large quantity of classes, it's impossible that all the dissimilar pairs have negative cosine similarity values. For example, when there are more than 4 classes in the 2-dimensional space, we can find at least one pair of classes with the angle less than 90° (i.e., cosine similarity value larger than 0). Thus the assumption of setting $cos(a_i, b_i) < 0$ for all the dissimilar pairs is only feasible if the dimension of the output feature space is large enough. However, for a large number of classes, this high-dimensional output space may lead to many local minima and over-fitting.

Therefore, we introduce a positive constant K to shift the decision boundary. Moreover, following [9, 10] that employed the logistic loss function in distance metric learning to create a decision gap between the similar pairs and dissimilar pairs [23], we incorporate the logistic loss function with the cosine similarity cost function as:

$$J = \frac{1}{n}\sum_{i=1}^{n} ln(1+exp(-\frac{s_i(cos(a_i, b_i) - K)}{T})) + \frac{\lambda}{2}\|A - A_0\|^2, \quad (4)$$

where the constant T is the sharpness parameter which is set to 0.1 in our experiments. The corresponding gradient function is:

$$\frac{\partial J}{\partial A} = \frac{1}{nT}\sum_{i=1}^{n}(1 - \frac{1}{h_i})\frac{\partial -s_i cos(a_i, b_i)}{\partial A} + \lambda(A - A_0), \quad (5)$$

where $h_i = 1+exp(-\frac{s_i(cos(a_i, b_i)-K)}{T})$ and the partial derivative $\frac{\partial -s_i cos(a_i, b_i)}{\partial A}$ is the same as in Equation (3).

Now we relate the LSML method to the task of face verification. At first, we collect labeled similar/dissimilar pairs of vectors which represent pairs of face images, i.e., (x_i, y_i, s_i), as training data. By initializing the linear transformation matrix A with the identity matrix, we can calculate the initial cost and gradient using Equations (4) and (5). After that, we employ the advanced L-BFGS [24] optimization algorithm to automatically update the transformation matrix A until the overall cost gets convergency. Compared with the standard gradient descent algorithm, the L-BFGS algorithm has no need to manually pick a learning rate and is usually much faster. The L-BFGS modules are provided by Mark Schmidt [1]. Finally, we will get an optimal solution $A_\star$ which produces the minimal cost on the current training data, and we use $A_\star$ to transform all the inputs (x_i, y_i) to the outputs (a_i, b_i), remind that $a_i = f(x_i, A_\star) = A_\star x_i$ and $b_i = f(y_i, A_\star) = A_\star y_i$. Formally, we call $A_\star$ the metric that have been learnt.

Naturally, we model the probability p_i that an output pair (a_i, b_i) is similar by the standard logistic function, i.e., the sigmoid function:

$$p_i = \frac{1}{1 + exp(-\frac{cos(a_i, b_i) - K}{T})}. \quad (6)$$

If p_i exceeds a pre-defined threshold γ, we label the pair (a_i, b_i) as similar, otherwise we assign it as dissimilar. The parameter γ is tuned on a validation set, and then the best parameter is used for test evaluation.

[1] http://www.di.ens.fr/ mschmidt/Software/minFunc.html

Table 1. Face verification accuracy (±standard error of the mean) on LFW-a under restricted configuration with label-free outside data. Dimension of the whitened feature vectors is 300. Comparing the performance, LSML>WCCN>CSML>Baseline.

Method	LBP		OCLBP		SIFT		Gabor	
	original	square root	original	square root	original	square root	original	square root
Baseline	77.17±0.49	79.73±0.38	80.43±0.25	81.55±0.44	76.88±0.42	77.52±0.49	75.28±0.45	77.25±0.32
CSML	79.47±0.55	82.92±0.47	82.62±0.55	84.67±0.58	81.88±0.47	82.88±0.37	78.52±0.59	80.38±0.51
WCCN	80.40±0.39	84.23±0.33	83.75±0.51	86.83±0.37	82.72±0.39	84.17±0.25	78.68±0.62	81.52±0.65
LSML	**83.58±0.66**	**85.17±0.50**	**85.48±0.69**	***87.55±0.49***	**84.67±0.46**	**85.77±0.37**	**80.98±0.70**	**83.28±0.43**

4. EXPERIMENT AND ANALYSIS

4.1. Experiment setting

We evaluate our method on the data set 'Labeled Faces in the Wild' (LFW) [22] which contains numerous annotated images collected from Yahoo News. This data set contains most kinds of facial variations in face pose, facial expression, illumination and partial occlusions, etc, and it has been the most popular benchmark for face verification. All of our experiments are performed under *the LFW restricted configuration with label-free outside data*: only the provided 6000 pairs of data are used for training and evaluation.

We only use View 2 subset of LFW for experimental performance evaluation. There are 5749 people in the data set which are divided into mutually exclusive 10 folds: the person in any fold would not appear in the other fold. The total number of images in LFW is 13233, however, the number of images for each person varies from 1 to 530.

We perform a 10-fold cross-validation on the aligned LFW-a data [25]: in each experiment, we select 8 out of the 10 folds as the training set, the other 2 folds are used for validation and testing respectively. For example, the first experiment uses subsets (1,2,3,4,5,6,7,8) for training, subset 9 for validation and subset 10 for testing; the second experiment uses (2,3,4,5,6,7,8,9) for training, subset 10 for validation and subset 1 for testing. After 10 repetitions, we report the mean accuracy (±standard error of the mean).

4.2. Feature vector

We use four face descriptors to represent the face images: Gabor wavelets [17], LBP [19], SIFT [18] and OCLBP [8]. For Gabor and LBP, we used exactly the same setting as in [5], dimension of Gabor and LBP is 4,800 and 7,080, respectively. For SIFT, we directly used the 3,456-d feature data provided by [9]. For OCLBP, the high dimensional variant of LBP, we used the same setting as in [26], dimension of the OCLBP descriptor is 46,846. Compared with LBP using non-overlapping shifting window, OCLBP allows overlapping to adjacent windows, therefore OCLBP is with much higher dimension than LBP and describes more detailed facial texture. Additionally, square roots of all the descriptors are also evaluated. Moreover, following [7], we reduce the dimension of all the raw feature vectors to 300 by whitened PCA.

4.3. Results and Analysis

We perform experiments with CSML [5] and LSML for face verification on LFW-a under *the LFW restricted configuration with label-free outside data*. Especially, we implemented the state-of-the-art method WCCN [8] as a comparison. In the experiments, we have three parameters to tune: the decision threshold γ, the regularization term λ and the shifting parameter K (only for LSML). The tuning range of γ was from 0 to 1 with a step size of 0.001 for all the experiments. The tuning range of λ was from 2×10^{-3} to 10×10^{-3} with a step size of 10^{-3} for CSML. For LSML, the tuning range of λ was from 15×10^{-3} to 20×10^{-3} with a step size of 10^{-3} and the tuning range of K was from 0 to 0.8 with a step size of 0.1.

4.3.1. Comparison with the state-of-the-art

To set a baseline, we first perform evaluation on the 300-d whitened feature vectors, i.e., setting the transformation matrix A as the identity matrix. Results on different features are listed in the first row of Table 1.

Comparing CSML with the baseline, we can see significant performance gain for all the features. For instance, on the square root of OCLBP, CSML obtains a performance gain from 81.55% to 84.67% over the baseline. WCCN [8], further increases the accuracy to 86.83% on the same feature. And the proposed LSML method performs the best on all the features (the fourth row in Table 1). For example, LSML achieves 87.55% on the square root of OCLBP. In summary, comparing the performance of the four methods, LSML>WCCN>CSML>Baseline.

4.3.2. Effectiveness of the shifting parameter K

Figure 1 shows the accuracy-versus-K curve of the proposed LSML method using the square root of OCLBP. We tune the shifting parameter from 0 to 0.8, and record the mean accuracy and its standard error on the 10-fold experiments. The regularization parameter λ is kept as 17×10^{-3}. We can see that the curve rises rapidly when the decision boundary is shifted from 0, and arrives the peak 87.55% at $K=0.5$.

This curve illustrates that shifting the decision boundary towards the positive side can adjust the cost from the similar training pairs and the dissimilar training pairs, which leads to considerable improvement of verification performance.

Table 2. Face verification accuracy (±standard error of the mean) on LFW-a under restricted configuration with label-free outside data. Dimension of the whitened feature vectors is 300. CSML-sim and LSML-sim learns on only the similar pairs from the training set. Comparing the performance, LSML=CSML-sim=LSML-sim.

Method	LBP		OCLBP		SIFT		Gabor	
	original	square root	original	square root	original	square root	original	square root
LSML	83.58±0.66	85.17±0.50	85.48±0.69	**87.55±0.49**	84.67±0.46	85.77±0.37	80.98±0.70	83.28±0.43
CSML-sim	83.27±0.73	85.32±0.56	85.43±0.70	**87.35±0.47**	85.07±0.47	85.98±0.44	81.28±0.59	83.55±0.50
LSML-sim	83.18±0.78	85.47±0.62	85.35±0.68	**87.35±0.48**	85.00±0.46	85.78±0.33	81.22±0.44	83.55±0.45

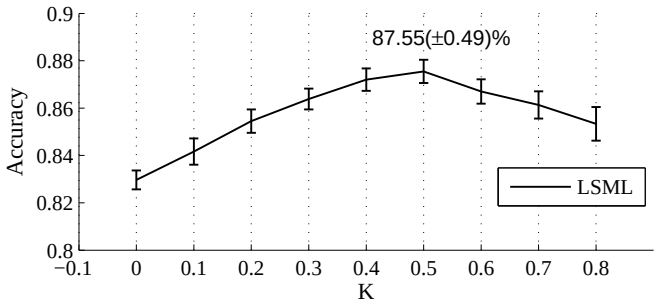

Fig. 1. Accuracy-versus-K curve of the proposed LSML method using the square root of OCLBP. The regularization parameter $\lambda = 17 \times 10^{-3}$. The peak $87.55 \pm 0.49\%$ is at $K = 0.5$.

4.3.3. Learning on similar pairs only

From another perspective on the logistic loss function (Equations (4) and (5)), shifting the decision boundary also means making similar pairs contribute more to the gradient than the dissimilar pairs. To verify this, we sum up the gradient coefficient $(1 - \frac{1}{h_i})$ in Equation (5) for similar pairs and dissimilar pairs, respectively. Generally, the coefficients are all positive numbers in the range $[0, 1]$ and larger coefficients imply more contribution to the gradient. In the example of Figure 1, when $K = 0$, the sum of the gradient coefficients for the similar training pairs is 523.0 and that for the dissimilar pairs is 1200.1; when $K = 0.5$, we get 2186.0 and 19.7 correspondingly. This means that with the decision boundary shifted from 0 to 0.5, the contribution of the similar pairs to the gradient has been increased dramatically.

Thus we propose an argument that *under the linear constraint, learning on similar pairs only can find a proper decision boundary automatically*. Coincidentally, the WCCN computation is only based on pairs from the same class [8]. Concretely, we perform learning only on the similar pairs from the training set for CSML and LSML, namely CSML-sim and LSML-sim: the cost and gradient functions are kept the same but the dissimilar training pairs are abandoned. For LSML-sim, we keep the shifting parameter K to be 0 and the sharpness parameter T to be 1. The results are reported in the last two rows of Table 2. We can see that the two methods achieve almost the same performance with the standard LSML method over all the features. For example, on the square-rooted SIFT descriptor, LSML, CSML-sim and LSML-sim obtain 85.77%, 85.98% and 85.78%, respectively.

Compared with the LSML that shifts the boundary by tuning a parameter K and trains on both similar and dissimilar pairs, fewer parameters and less training data lead to faster training for CSML-sim and LSML-sim. However, it is worth noting that it should be under the linear constraint, otherwise training on similar pairs only is prone to a large over-fitting problem.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a new method called Logistic Similarity Metric Learning for face verification. Specifically, we introduce a parameter K to shift the similarity decision boundary, formulate the cost using the logistic loss function, and produce a probability estimation of a pair of faces being similar.

We performed extensive experiments on the LFW-a data set [22] under restricted configuration with label-free outside data. The proposed method achieved superior performance over the state-of-the-art linear methods. Moreover, we propose a faster way to achieve the same goal: learning on similar pairs only. Learning on similar pairs has one thing in common with shifting the boundary that both of them make the similar training pairs contribute more to the gradient than the dissimilar training pairs. And the latter has fewer parameters to tune and requires less data for training. However, this should be under the linear constraint to prevent the probable large over-fitting problem in training.

In the future, we plan to integrate the similarity metric learning model with some distance metric learning model. We are interested in taking advantages of both the cosine similarity and the Mahalanobis distance [9, 10] to improve the performance of face verification.

6. REFERENCES

[1] A. Bellet, A. Habrard, and M. Sebban, "A survey on metric learning for feature vectors and structured data," *Computing Research Repository*, vol. abs/1306.6709, 2013.

[2] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell, "Distance metric learning with application to clustering with side-information," *Advances in neural information processing systems*, pp. 521–528, 2003.

[3] K. Weinberger, J. Blitzer, and L. Saul, "Distance metric learning for large margin nearest neighbor classification," *Advances in neural information processing systems*, vol. 18, pp. 1473, 2006.

[4] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, "Information-theoretic metric learning," in *Proc. ICML*. ACM, 2007, pp. 209–216.

[5] N. V. Hieu and B. Li, "Cosine similarity metric learning for face verification," in *Proc. ACCV*. 2011, pp. 709–720, Springer.

[6] P. Li, Y. Fu, U. Mohammed, J. H. Elder, and S. JD Prince, "Probabilistic models for inference about identity," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 1, pp. 144–157, 2012.

[7] Q. Cao, Y. Ying, and P. Li, "Similarity metric learning for face recognition," in *Proc. ICCV*, 2013.

[8] O. Barkan, J. Weill, L. Wolf, and H. Aronowitz, "Fast high dimensional vector multiplication face recognition," in *Proc. ICCV*, 2013.

[9] M. Guillaumin, J. Verbeek, and C. Schmid, "Is that you? metric learning approaches for face identification," in *Proc. ICCV*. IEEE, 2009, pp. 498–505.

[10] Junlin Hu, Jiwen Lu, and Yap-Peng Tan, "Discriminative deep metric learning for face verification in the wild," in *Proc. CVPR*, 2014, pp. 1875–1882.

[11] A. J. O'Toole, H. Abdi, F. Jiang, and P. J. Phillips, "Fusing face-verification algorithms and humans," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, no. 5, pp. 1149–1155, 2007.

[12] A. M. Qamar, E. Gaussier, J. P. Chevallet, and J. H. Lim, "Similarity learning for nearest neighbor classification," in *Proc. ICDM*. IEEE, 2008, pp. 983–988.

[13] G. Chechik, V. Sharma, U. Shalit, and S. Bengio, "Large scale online learning of image similarity through ranking," *The Journal of Machine Learning Research*, vol. 11, pp. 1109–1135, 2010.

[14] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proc. CVPR*. IEEE, 2005, vol. 1, pp. 539–546.

[15] D. Kedem, S. Tyree, F. Sha, G. R. Lanckriet, and K. Q. Weinberger, "Non-linear metric learning," in *Advances in Neural Information Processing Systems*, 2012, pp. 2573–2581.

[16] Ma. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proc. CVPR*. IEEE, 1991, pp. 586–591.

[17] J. G. Daugman, "Complete discrete 2-d gabor transforms by neural networks for image analysis and compression," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 36, no. 7, pp. 1169–1179, 1988.

[18] Y. Ke and R. Sukthankar, "Pca-sift: A more distinctive representation for local image descriptors," in *Proc. CVPR*. IEEE, 2004, vol. 2, pp. II–506.

[19] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face recognition with local binary patterns," in *Proc. ECCV*. 2004, pp. 469–481, Springer.

[20] Y. Sun, X. Wang, and X. Tang, "Hybrid deep learning for face verification," in *Proc. ICCV*. IEEE, 2013, pp. 1489–1496.

[21] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proc. CVPR*, 2014, pp. 1701–1708.

[22] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," Tech. Rep., University of Massachusetts, Amherst, 2007.

[23] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang, "A tutorial on energy-based learning," *Predicting structured data*, 2006.

[24] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical programming*, vol. 45, no. 1-3, pp. 503–528, 1989.

[25] Y. Taigman, L. Wolf, and T. Hassner, "Multiple one-shots for utilizing class label information.," in *Proc. BMVC*, 2009, pp. 1–12.

[26] L. Zheng, K. Idrissi, C. Garcia, S. Duffner, and A. Baskurt, "Triangular Similarity Metric Learning for Face Verification," in *Proc. FG*, 2015.