Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

Surpassing Human-Level Face Verification Performance on LFW with GaussianFace

Chaochao Lu **Xiaoou Tang**
Department of Information Engineering
The Chinese University of Hong Kong
{lc013, xtang}@ie.cuhk.edu.hk

Abstract

Face verification remains a challenging problem in very complex conditions with large variations such as pose, illumination, expression, and occlusions. This problem is exacerbated when we rely unrealistically on a single training data source, which is often insufficient to cover the intrinsically complex face variations. This paper proposes a principled multi-task learning approach based on Discriminative Gaussian Process Latent Variable Model (DGPLVM), named GaussianFace, for face verification. In contrast to relying unrealistically on a single training data source, our model exploits additional data from multiple source-domains to improve the generalization performance of face verification in an unknown target-domain. Importantly, our model can adapt automatically to complex data distributions, and therefore can well capture complex face variations inherent in multiple sources. To enhance discriminative power, we introduced a more efficient equivalent form of Kernel Fisher Discriminant Analysis to DGPLVM. To speed up the process of inference and prediction, we exploited the low rank approximation method. Extensive experiments demonstrated the effectiveness of the proposed model in learning from diverse data sources and generalizing to unseen domains. Specifically, the accuracy of our algorithm achieved an impressive accuracy rate of 98.52% on the well-known and challenging Labeled Faces in the Wild (LFW) benchmark. For the first time, the human-level performance in face verification (97.53%) on LFW is surpassed.

Introduction

Face verification, which is the task of determining whether a pair of face images are from the same person, has been an active research topic in computer vision (Kumar et al. 2009; Simonyan et al. 2013; Chen et al. 2013; Cao et al. 2013). It has many important applications, including surveillance, access control, image retrieval, and automatic log-on for personal computers or mobile devices. However, various visual complications deteriorate the performance of face verification, as shown by numerous studies on real-world face images from the wild (Huang et al. 2007). The Labeled Faces in the Wild (LFW) dataset is well known as a challenging benchmark for face verification. The dataset provides a large set of relatively unconstrained face images with complex variations in pose, lighting, expression, race, ethnicity, age, gender, clothing, hairstyles, and other parameters. Not surprisingly, LFW has proven difficult for automatic face verification methods (Huang et al. 2007; Kumar et al. 2009; Lu, Zhao, and Tang 2013; Simonyan et al. 2013; Sun, Wang, and Tang 2013; Taigman et al. 2014). Although there has been significant work (Cao et al. 2013; Chen et al. 2013; Sun, Wang, and Tang 2013; 2014; Taigman et al. 2014; Lu and Tang 2014) on LFW and the accuracy rate has been improved from 60.02% (Turk and Pentland 1991) to 97.35% (Taigman et al. 2014) since LFW is established in 2007, these studies have not closed the gap to human-level performance (Kumar et al. 2009) in face verification.

We motivate this paper by highlighting two weaknesses as follows:

1) Most existing face verification methods assume that the training data and the test data are drawn from the same feature space and follow the same distribution. When the distribution changes, these methods may suffer a large performance drop (Wright and Hua 2009). However, many practical scenarios involve cross-domain data drawn from different facial appearance distributions. Learning a model solely on a single source data often leads to overfitting due to dataset bias (Torralba and Efros 2011). Moreover, it is difficult to collect sufficient and necessary training data to rebuild the model in new scenarios, for highly accurate face verification specific to the target domain. In such cases, it becomes critical to exploit more data from multiple source-domains to improve the generalization of face verification methods in the target-domain.

2) Modern face verification methods are mainly divided into two categories: extracting low-level features (Lowe 2004; Ahonen, Hadid, and Pietikainen 2006; Cao et al. 2010), and building classification models (Sun, Wang, and Tang 2013; Chen et al. 2012; Moghaddam, Jebara, and Pentland 2000; Turk and Pentland 1991; Li et al. 2005). Although these existing methods have made great progress in face verification, most of them are less flexible when dealing with complex data distributions. For the methods in the first category, for example, low-level features such as SIFT (Lowe 2004), LBP (Ahonen, Hadid, and Pietikainen 2006), and Gabor (Liu and Wechsler 2002) are handcrafted. Even

for features learned from data (Cao et al. 2010), the algorithm parameters (such as the depth of random projection tree, or the number of centers in k-means) also need to be specified by users. Similarly, for the methods in the second category, the architectures of deep networks in (Sun, Wang, and Tang 2013; Taigman et al. 2014; Sun, Wang, and Tang 2014) (for example, the number of layers, the number of nodes in each layer, etc.), and the parameters of the models in (Li et al. 2013; Berg and Belhumeur 2012; Kumar et al. 2009; Simonyan et al. 2013) (for example, the number of Gaussians, the number of classifiers, etc.) must also be determined in advance. Since most existing methods require some assumptions to be made about the structures of the data, they cannot work well when the assumptions are not valid. Moreover, due to the existence of the assumptions, it is hard to capture the intrinsic structures of data using these methods.

To this end, we propose the multi-task learning approach based on Discriminative Gaussian Process Latent Variable Model (DGPLVM) (Urtasun and Darrell 2007), named *GaussianFace*, for face verification. In order to take advantage of more data from multiple source-domains to improve the performance in the target-domain, we introduce the multi-task learning constraint to DGPLVM. Here, we investigate the asymmetric multi-task learning because we only focus on the performance improvement of the target task. Moreover, the GaussianFace model is a reformulation based on the Gaussian Processes (GPs) (Rasmussen and Williams 2006), which is a non-parametric Bayesian kernel method. Therefore, our model also can adapt its complexity flexibly to the complex data distributions in the real-world, without any heuristics or manual tuning of parameters.

Reformulating GPs for large-scale multi-task learning is non-trivial. To simplify calculations, we introduce a more efficient equivalent form of Kernel Fisher Discriminant Analysis (KFDA) to DGPLVM. Despite that the GaussianFace model can be optimized effectively using the Scaled Conjugate Gradient (SCG) technique, the inference is slow for large-scale data. We make use of GP approximations (Rasmussen and Williams 2006) and low rank approximation (Liu, He, and Chang 2010) to speed up the process of inference and prediction, so as to scale our model to large-scale data. Our model can be applied to face verification in two different ways: as a binary classifier and as a feature extractor. In the former mode, given a pair of face images, we can directly compute the posterior likelihood for each class to make a prediction. In the latter mode, our model can automatically extract high-dimensional features for each pair of face images, and then feed them to a classifier to make the final decision.

The main contributions of this paper are as follows: (1) We propose a novel GaussianFace model for face verification by virtue of the multi-task learning constraint to DGPLVM. Our model can adapt to complex distributions, avoid over-fitting, exploit discriminative information, and take advantage of multiple source-domains data. (2) We introduce a more efficient and dicriminative equivalent form of KFDA to DGPLVM. This equivalent form reformulates KFDA to the kernel version consistent with the covariance function in GPs, which greatly simplifies calculations. (3) We introduce the low rank approximation to speed up the process of inference and prediction. (4) Based on GaussianFace model, we propose two different approaches for face verification: a binary classifier and a feature extractor. (5) We achieve superior performance on the challenging LFW benchmark (Huang et al. 2007), with an accuracy rate of 98.52%, beyond human-level performance reported in (Kumar et al. 2009) for the first time.

Preliminary

In this section, we briefly review Gaussian Processes (GPs) for classification and clustering (Kim and Lee 2007), and Gaussian Process Latent Variable Model (GPLVM) (Lawrence 2003). We recommend Rasmussen and Williams's excellent monograph for further reading (Rasmussen and Williams 2006).

Gaussian Processes for Binary Classification

Formally, for two-class classification, suppose that we have a training set $\mathcal{D}$ of N observations, $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, where the i-th input point $\mathbf{x}_i \in \mathbb{R}^D$ and its corresponding output y_i is binary, with $y = 1_i$ for one class and $y_i = -1$ for the other. Let $\mathbf{X}$ be the $N \times D$ matrix, where the row vectors represent all n input points, and $\mathbf{y}$ be the column vector of all n outputs. We define a latent variable f_i for each input point $\mathbf{x}_i$, and let $\mathbf{f} = [f_1, \ldots, f_N]^\top$. A sigmoid function $\pi(\cdot)$ is imposed to squash the output of the latent function into $[0, 1]$, $\pi(f_i) = p(y_i = 1|f_i)$. Assuming the data set is i.i.d, then the joint likelihood factorizes to

$$p(\mathbf{y}|\mathbf{f}) = \prod_{i=1}^{N} p(y_i|f_i) = \prod_{i=1}^{N} \pi(y_i f_i). \tag{1}$$

Moreover, the posterior distribution over latent functions is

$$p(\mathbf{f}|\mathbf{X}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{X})}{p(\mathbf{y}|\mathbf{X})}, \tag{2}$$

where $p(\mathbf{f}|\mathbf{X})$ is a zero-mean Gaussian Process prior over f with covariance $\mathbf{K}$ and $\mathbf{K}_{i,j} = k(x_i, x_j)$. The hyper-parameters of $\mathbf{K}$ are denoted by $\boldsymbol{\theta}$. Since neither $p(\mathbf{f}|\mathbf{X}, \mathbf{y})$ nor $p(\mathbf{y}|\mathbf{f})$ can be computed analytically, the Laplace method is utilized to approximate the posterior. The optimal value of $\boldsymbol{\theta}$ can be acquired by using the gradient method. Given any unseen test point x_*, the probability of its latent function f_* is

$$f_*|\mathbf{X}, \mathbf{y}, x_* \sim \mathcal{N}(\mathbf{K}_* \mathbf{K}^{-1} \hat{\mathbf{f}}, \mathbf{K}_{**} - \mathbf{K}_* \tilde{\mathbf{K}}^{-1} \mathbf{K}_*^\top), \tag{3}$$

where $\tilde{\mathbf{K}} = \mathbf{K} + \mathbf{W}^{-1}$, $\mathbf{W} = -\nabla\nabla \log p(\mathbf{f}|\mathbf{X}, \mathbf{y})|_{\mathbf{f}=\hat{\mathbf{f}}}$, $\mathbf{K}_* = [k(x_*, x_1), \ldots, k(x_*, x_N)]$, $\mathbf{K}_{**} = k(x_*, x_*)$ and $\hat{\mathbf{f}} = \arg\max_{\mathbf{f}} p(\mathbf{f}|\mathbf{X}, \mathbf{y})$. Finally, we squash f_* to find the probability of class membership as follows

$$\bar{\pi}(f_*) = \int \pi(f_*) p(f_*|\mathbf{X}, \mathbf{y}, x_*) \mathrm{d}f_*. \tag{4}$$

Gaussian Processes for Clustering

The principle of GP clustering is based on the key observation that the variances of predictive values are smaller in dense areas and larger in sparse areas. The variances can be

employed as a good estimate of the support of a probability density function, where each separate support domain can be considered as a cluster. This observation can be explained from the variance function of any predictive data point x_*

$$\sigma^2(x_*) = \mathbf{K}_{**} - \mathbf{K}_* \tilde{\mathbf{K}}^{-1} \mathbf{K}_*^\top. \quad (5)$$

If x_* is in a sparse region, then $\mathbf{K}_* \tilde{\mathbf{K}}^{-1} \mathbf{K}_*^\top$ becomes small, which leads to large variance $\sigma^2(x_*)$, and vice versa. Another good property of Equation (5) is that it does not depend on the labels, which means it can be applied to the unlabeled data.

To perform clustering, the following dynamic system associated with Equation (5) can be written as

$$F(x) = -\nabla \sigma^2(x). \quad (6)$$

The theorem in (Kim and Lee 2007) guarantees that almost all the trajectories approach one of the stable equilibrium points detected from Equation (6). After each data point finds its corresponding stable equilibrium point, we can employ a complete graph (Ben-Hur et al. 2002; Kim and Lee 2007) to assign cluster labels to data points with the stable equilibrium points. Obviously, the variance function in Equation (5) completely determines the performance of clustering.

Gaussian Process Latent Variable Model

GPLVM has been extensively studied (Damianou and Lawrence 2013; Lawrence 2003). Formally, let $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_N]^\top$ denote the matrix whose rows represent corresponding positions of $\mathbf{X}$ in latent space, where $\mathbf{z}_i \in \mathbb{R}^d$ ($d \ll D$). The GPLVM can be interpreted as a Gaussian process mapping from a low dimensional latent space to a high dimensional data set, where the locale of the points in latent space is determined by maximizing the Gaussian process likelihood with respect to $\mathbf{Z}$. Given a covariance function for the Gaussian process, denoted by $k(\cdot, \cdot)$, the likelihood of the data given the latent positions is as follows,

$$p(\mathbf{X}|\mathbf{Z}, \boldsymbol{\theta}) = \frac{1}{\sqrt{(2\pi)^{ND}|\mathbf{K}|^D}} \exp\Big(-\frac{1}{2}\mathrm{tr}(\mathbf{K}^{-1}\mathbf{X}\mathbf{X}^\top)\Big), \quad (7)$$

where $\mathbf{K}_{i,j} = k(\mathbf{z}_i, \mathbf{z}_j)$. Therefore, the posterior can be written as

$$p(\mathbf{Z}, \boldsymbol{\theta}|\mathbf{X}) = \frac{1}{\mathcal{Z}_a} p(\mathbf{X}|\mathbf{Z}, \boldsymbol{\theta}) p(\mathbf{Z}) p(\boldsymbol{\theta}), \quad (8)$$

where $\mathcal{Z}_a$ is a normalization constant, the uninformative priors over $\boldsymbol{\theta}$, and the simple spherical Gaussian priors over $\mathbf{Z}$ are introduced (Urtasun and Darrell 2007). To obtain the optimal $\boldsymbol{\theta}$ and $\mathbf{Z}$, we need to optimize the above likelihood (8) with respect to $\boldsymbol{\theta}$ and $\mathbf{Z}$, respectively.

GaussianFace

In order to automatically learn discriminative features or covariance function, and to take advantage of source-domain data to improve the performance in face verification, we develop a principled GaussianFace model by including the multi-task learning constraint into Discriminative Gaussian Process Latent Variable Model (DGPLVM) (Urtasun and Darrell 2007).

DGPLVM Reformulation

The DGPLVM is an extension of GPLVM, where the discriminative prior is placed over the latent positions, rather than a simple spherical Gaussian prior. The DGPLVM uses the discriminative prior to encourage latent positions of the same class to be close and those of different classes to be far. Since face verification is a binary classification problem and the GPs mainly depend on the kernel function, it is natural to use Kernel Fisher Discriminant Analysis (KFDA) (Kim, Magnani, and Boyd 2006) to model class structures in kernel spaces. For simplicity of inference in the followings, we introduce another equivalent formulation of KFDA to replace the one in (Urtasun and Darrell 2007).

KFDA is a kernelized version of linear discriminant analysis method. It finds the direction defined by a kernel in a feature space, onto which the projections of positive and negative classes are well separated by maximizing the ratio of the between-class variance to the within-class variance. Formally, let $\{\mathbf{z}_1, \ldots, \mathbf{z}_{N_+}\}$ denote the positive class and $\{\mathbf{z}_{N_+ + 1}, \ldots, \mathbf{z}_N\}$ the negative class, where the numbers of positive and negative classes are N_+ and $N_- = N - N_+$, respectively. Let $\mathbf{K}$ be the kernel matrix. Therefore, in the feature space, the two sets $\{\phi_{\mathbf{K}}(\mathbf{z}_1), \ldots, \phi_{\mathbf{K}}(\mathbf{z}_{N_+})\}$ and $\{\phi_{\mathbf{K}}(\mathbf{z}_{N_+ + 1}), \ldots, \phi_{\mathbf{K}}(\mathbf{z}_N)\}$ represent the positive class and the negative class, respectively. The optimization criterion of KFDA is to maximize the ratio of the between-class variance to the within-class variance

$$J(\omega, \mathbf{K}) = \frac{(\mathbf{w}^\top(\mu_{\mathbf{K}}^+ - \mu_{\mathbf{K}}^-))^2}{\mathbf{w}^\top(\boldsymbol{\Sigma}_{\mathbf{K}}^+ + \boldsymbol{\Sigma}_{\mathbf{K}}^- + \lambda \mathbf{I}_N)\mathbf{w}}, \quad (9)$$

where λ is a positive regularization parameter, $\mu_{\mathbf{K}}^+ = \frac{1}{N_+}\sum_{i=1}^{N_+} \phi_{\mathbf{K}}(\mathbf{z}_i)$, $\mu_{\mathbf{K}}^- = \frac{1}{N_-}\sum_{i=N_+ + 1}^{N} \phi_{\mathbf{K}}(\mathbf{z}_i)$, $\boldsymbol{\Sigma}_{\mathbf{K}}^+ = \frac{1}{N_+}\sum_{i=1}^{N_+}(\phi_{\mathbf{K}}(\mathbf{z}_i) - \mu_{\mathbf{K}}^+)(\phi_{\mathbf{K}}(\mathbf{z}_i) - \mu_{\mathbf{K}}^+)^\top$, and $\boldsymbol{\Sigma}_{\mathbf{K}}^- = \frac{1}{N_-}\sum_{i=N_+ + 1}^{N}(\phi_{\mathbf{K}}(\mathbf{z}_i) - \mu_{\mathbf{K}}^-)(\phi_{\mathbf{K}}(\mathbf{z}_i) - \mu_{\mathbf{K}}^-)^\top$.

In this paper, however, we focus on the covariance function rather than the latent positions. To simplify calculations, we represent Equation (9) with the kernel function, and let the kernel function have the same form as the covariance function. Therefore, it is natural to introduce a more efficient equivalent form of KFDA with certain assumptions as Kim *et al.* points out (Kim, Magnani, and Boyd 2006), i.e., maximizing Equation (9) is equivalent to maximizing the following equation

$$J^* = \frac{1}{\lambda}(\mathbf{a}^\top \mathbf{K}\mathbf{a} - \mathbf{a}^\top \mathbf{K}\mathbf{A}(\lambda \mathbf{I}_n + \mathbf{A}\mathbf{K}\mathbf{A})^{-1}\mathbf{A}\mathbf{K}\mathbf{a}), \quad (10)$$

where $\mathbf{A} = \mathrm{diag}\Big(\frac{\big(\mathbf{I}_{N_+} - \frac{1}{N_+}\mathbf{1}_{N_+}\mathbf{1}_{N_+}^\top\big)}{\sqrt{N_+}}, \frac{\big(\mathbf{I}_{N_-} - \frac{1}{N_-}\mathbf{1}_{N_-}\mathbf{1}_{N_-}^\top\big)}{\sqrt{N_-}}\Big)$, $\mathbf{a} = [\frac{\mathbf{1}_{N_+}^\top}{N_+}, -\frac{\mathbf{1}_{N_-}^\top}{N_-}]$, and λ is a positive regularization parameter. Here, $\mathbf{I}_N$ denotes the $N \times N$ identity matrix and $\mathbf{1}_N$ denotes the length-N vector of all ones in $\mathbb{R}^N$. Therefore, the discriminative prior over the latent positions in DGPLVM can be written as

$$p(\mathbf{Z}) = \frac{1}{\mathcal{Z}_b} \exp\Big(-\frac{1}{\sigma^2} J^*\Big), \quad (11)$$

where $\mathcal{Z}_b$ is a normalization constant, and σ^2 represents a global scaling of the prior.

The covariance matrix obtained by DGPLVM is discriminative and more flexible than the one used in conventional GPs for classification (GPC), since they are learned based on a discriminative criterion, and more degrees of freedom are estimated than conventional kernel hyper-parameters.

Multi-task Learning Constraint

From an asymmetric multi-task learning perspective, the tasks should be allowed to share common hyper-parameters of the covariance function. Moreover, from an information theory perspective, the information cost between target task and multiple source tasks should be minimized. A natural way to quantify the information cost is to use the mutual entropy, because it is the measure of the mutual dependence of two distributions. For multi-task learning, we extend the mutual entropy to multiple distributions as follows

$$\mathcal{M} = H(p_t) - \frac{1}{S}\sum_{i=1}^{S} H(p_t|p_i), \tag{12}$$

where $H(\cdot)$ is the marginal entropy, $H(\cdot|\cdot)$ is the conditional entropy, S is the number of source tasks, $\{p_i\}_{i=1}^{S}$, and p_t are the probability distributions of source tasks and target task, respectively.

GaussianFace Model

In this section, we describe our GaussianFace model in detail. Suppose we have S source-domain datasets $\{\mathbf{X}_1, \ldots, \mathbf{X}_S\}$ and a target-domain data $\mathbf{X}_T$. For each source-domain data or target-domain data $\mathbf{X}_i$, according to Equation (7), we write its marginal likelihood

$$p(\mathbf{X}_i|\mathbf{Z}_i, \boldsymbol{\theta}) = \frac{1}{\sqrt{(2\pi)^{ND}|\mathbf{K}|^D}} \exp\Big(-\frac{1}{2}\mathrm{tr}(\mathbf{K}^{-1}\mathbf{X}_i\mathbf{X}_i^\top)\Big). \tag{13}$$

where $\mathbf{Z}_i$ represents the domain-relevant latent space. For each source-domain data and target-domain data, their covariance functions $\mathbf{K}$ have the same form because they share the same hyper-parameters $\boldsymbol{\theta}$. In this paper, we use a widely used kernel

$$\begin{aligned}\mathbf{K}_{i,j} = k_{\boldsymbol{\theta}}(\mathbf{z}_i, \mathbf{z}_j) =& \theta_0 \exp\Big(-\frac{1}{2}\sum_{m=1}^{d}\theta_m(\mathbf{z}_i^m - \mathbf{z}_j^m)^2\Big) \\ &+ \theta_{d+1} + \frac{\delta_{\mathbf{z}_i,\mathbf{z}_j}}{\theta_{d+2}},\end{aligned} \tag{14}$$

where $\boldsymbol{\theta} = \{\theta_i\}_{i=0}^{d+2}$ and d is the dimension of the data point. Then, from Equations (8), learning the DGPLVM is equivalent to optimizing

$$p(\mathbf{Z}_i, \boldsymbol{\theta}|\mathbf{X}_i) = \frac{1}{\mathcal{Z}_a} p(\mathbf{X}_i|\mathbf{Z}_i, \boldsymbol{\theta}) p(\mathbf{Z}_i) p(\boldsymbol{\theta}), \tag{15}$$

where $p(\mathbf{X}_i|\mathbf{Z}_i, \boldsymbol{\theta})$ and $p(\mathbf{Z}_i)$ are respectively represented in (13) and (11). According to the multi-task learning constraint in Equation (12), we can attain

$$\mathcal{M} = H(p(\mathbf{Z}_T, \boldsymbol{\theta}|\mathbf{X}_T)) - \frac{1}{S}\sum_{i=1}^{S} H(p(\mathbf{Z}_T, \boldsymbol{\theta}|\mathbf{X}_T)|p(\mathbf{Z}_i, \boldsymbol{\theta}|\mathbf{X}_i)). \tag{16}$$

From Equations (13), (15), and (16), we know that learning the GaussianFace model amounts to minimizing the following marginal likelihood

$$\mathcal{L}_{Model} = -\log p(\mathbf{Z}_T, \boldsymbol{\theta}|\mathbf{X}_T) - \beta\mathcal{M}, \tag{17}$$

where the parameter β balances the relative importance between the target-domain data and the multi-task learning constraint. We can now optimize Equation (17) with respect to the hyper-parameters $\boldsymbol{\theta}$ and the latent positions $\mathbf{Z}_i$ by the Scaled Conjugate Gradient (SCG) technique. More detailed derivations can be found in the supplementary material.

Speedup

In the GaussianFace model, we need to invert the large matrix when doing inference and prediction. For large problems, both storing the matrix and solving the associated linear systems are computationally prohibitive. In this paper, we use the anchor graphs method (Liu, He, and Chang 2010) to speed up this process. To put it simply, we first select q ($q \ll n$) anchors to cover a cloud of n data points, and form an $n \times q$ matrix $\mathbf{Q}$, where $\mathbf{Q}_{i,j} = k_{\boldsymbol{\theta}}(\mathbf{z}_i, \mathbf{z}_j)$. $\mathbf{z}_i$ and $\mathbf{z}_j$ are from n latent data points and q anchors, respectively. Then the original kernel matrix $\mathbf{K}$ can be approximated as $\mathbf{K} \approx \mathbf{Q}\mathbf{Q}^\top$. Using the Woodbury identity (Higham 1996), computing the $n \times n$ matrix $\mathbf{Q}\mathbf{Q}^\top$ can be transformed into computing the $q \times q$ matrix $\mathbf{Q}^\top\mathbf{Q}$, which is more efficient.

Speedup on Inference When optimizing Equation (17), we need to invert the matrix $(\lambda\mathbf{I}_n + \mathbf{A}\mathbf{K}\mathbf{A})$. During inference, we take q k-means clustering centers as anchors to form $\mathbf{Q}$. Substituting $\mathbf{K} \approx \mathbf{Q}\mathbf{Q}^\top$ into $(\lambda\mathbf{I}_n + \mathbf{A}\mathbf{K}\mathbf{A})$, and then using the Woodbury identity, we get

$$\begin{aligned}(\lambda\mathbf{I}_n + \mathbf{A}\mathbf{K}\mathbf{A})^{-1} &\approx (\lambda\mathbf{I}_n + \mathbf{A}\mathbf{Q}\mathbf{Q}^\top\mathbf{A})^{-1} \\ &= \lambda^{-1}\mathbf{I}_n - \lambda^{-1}\mathbf{A}\mathbf{Q}(\lambda\mathbf{I}_q + \mathbf{Q}^\top\mathbf{A}\mathbf{A}\mathbf{Q})^{-1}\mathbf{Q}^\top\mathbf{A}.\end{aligned}$$

Similarly, let $\mathbf{K}^{-1} \approx (\mathbf{K} + \tau\mathbf{I})^{-1}$ where τ a constant term, then we can get

$$\mathbf{K}^{-1} \approx (\mathbf{K} + \tau\mathbf{I})^{-1} \approx \tau^{-1}\mathbf{I}_n - \tau^{-1}\mathbf{Q}(\tau\mathbf{I}_q + \mathbf{Q}^\top\mathbf{Q})^{-1}\mathbf{Q}^\top.$$

Speedup on Prediction When we compute the predictive variance $\sigma(\mathbf{z}_*)$, we need to invert the matrix $(\mathbf{K} + \mathbf{W}^{-1})$. At this time, we can use the method in Section to calculate the accurate clustering centers that can be regarded as the anchors. Using the Woodbury identity again, we obtain

$$(\mathbf{K} + \mathbf{W}^{-1})^{-1} \approx \mathbf{W} - \mathbf{W}\mathbf{Q}(\mathbf{I}_q + \mathbf{Q}^\top\mathbf{W}\mathbf{Q})^{-1}\mathbf{Q}^\top\mathbf{W},$$

where $(\mathbf{I}_q + \mathbf{Q}^\top\mathbf{W}\mathbf{Q})$ is only a $q \times q$ matrix, and its inverse matrix can be computed more efficiently.

GaussianFace Model for Face Verification

In this section, we describe two applications of the GaussianFace model to face verification: as a binary classifier and as a feature extractor. Each face image is first normalized to 150×120 size by an affine transformation based on five landmarks (two eyes, nose, and two mouth corners). The image is then divided into overlapped patches of 25×25 pixels with

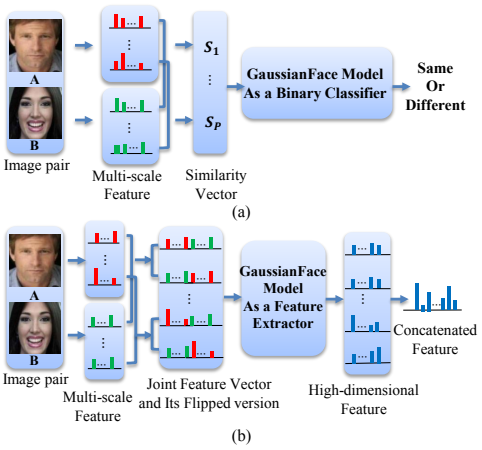

Figure 1: Two approaches based on GaussianFace model for face verification. (a) GaussianFace model as a binary classifier. (b) GaussianFace model as a feature extractor.

a stride of 2 pixels. Each patch within the image is mapped to a vector by a certain descriptor, and the vector is regarded as the feature of the patch, denoted by $\{\mathbf{x}_p^A\}_{p=1}^P$ where P is the number of patches within the face image A. In this paper, the multi-scale LBP feature of each patch is extracted (Chen et al. 2013). The difference is that the multi-scale LBP descriptors are extracted at the center of each patch instead of accurate landmarks.

GaussianFace Model as a Binary Classifier

For classification, our model can be regarded as an approach to learn a covariance function for GPC, as shown in Figure 1 (a). Here, for a pair of face images A and B from the same (or different) person, let the similarity vector $\mathbf{x}_i = [s_1, \ldots, s_p, \ldots, s_P]^\top$ be the input data point of the GaussianFace model, where s_p is the similarity of $\mathbf{x}_p^A$ and $\mathbf{x}_p^B$, and its corresponding output is $y_i = 1$ (or -1). With the learned hyper-parameters of covariance function from the training data, given any un-seen pair of face images, we first compute its similarity vector $\mathbf{x}_*$ using the above method, then estimate its latent representation $\mathbf{z}_*$ using the same method in (Urtasun and Darrell 2007) [1], and finally predict whether the pair is from the same person through Equation (4). In this paper, we prescribe the sigmoid function $\pi(\cdot)$ to be the cumulative Gaussian distribution $\Phi(\cdot)$, which can be solved analytically as $\bar{\pi}_* = \Phi\big(\bar{f}_*(\mathbf{z}_*)/\sqrt{1+\sigma^2(\mathbf{z}_*)}\big)$, where $\sigma^2(\mathbf{z}_*) = \mathbf{K}_{**} - \mathbf{K}_*\tilde{\mathbf{K}}^{-1}\mathbf{K}_*^\top$ and $\bar{f}_*(\mathbf{z}_*) = \mathbf{K}_*\mathbf{K}^{-1}\hat{\mathbf{f}}$ from Equation (3) (Rasmussen and Williams 2006). We call the method *GaussianFace-BC*.

GaussianFace Model as a Feature Extractor

As a feature extractor, our model can be regarded as an approach to automatically extract facial features, shown in Figure 1 (b). Here, for a pair of face images A and B from the same (or different) person, we regard the joint feature vector $\mathbf{x}_i = [(\mathbf{x}_i^A)^\top, (\mathbf{x}_i^B)^\top]^\top$ as the input data point of the GaussianFace model, and its corresponding output is $y_i = 1$ (or -1). To enhance the robustness of our approach, the flipped form of $\mathbf{x}_i$ is also included; for example, $\mathbf{x}_i = [(\mathbf{x}_i^B)^\top, (\mathbf{x}_i^A)^\top]^\top$. After the hyper-parameters of covariance function are learnt from the training data, we first estimate the latent representations of the training data using the same method in (Urtasun and Darrell 2007), then can use the method in Section to group the latent data points into different clusters automatically. Suppose that we finally obtain C clusters. The centers of these clusters are denoted by $\{\mathbf{c}_i\}_{i=1}^C$, the variances of these clusters by $\{\Sigma_i^2\}_{i=1}^C$, and their weights by $\{w_i\}_{i=1}^C$ where w_i is the ratio of the number of latent data points from the i-th cluster to the number of all latent data points. Then we refer to each $\mathbf{c}_i$ as the input of Equation (3), and we can obtain its corresponding probability p_i and variance σ_i^2. In fact, $\{\mathbf{c}_i\}_{i=1}^C$ can be regarded as a codebook generated by our model.

For any un-seen pair of face images, we also first compute its joint feature vector $\mathbf{x}_*$ for each pair of patches, and estimate its latent representation $\mathbf{z}_*$. Then we compute its first-order and second-order statistics to the centers. Similarly, we regard $\mathbf{z}_*$ as the input of Equation (3), and can also obtain its corresponding probability p_* and variance σ_*^2. The statistics and variance of $\mathbf{z}_*$ are represented as its high-dimensional facial features, denoted by $\hat{\mathbf{z}}_* = [\Delta_1^1, \Delta_1^2, \Delta_1^3, \Delta_1^4, \ldots, \Delta_C^1, \Delta_C^2, \Delta_C^3, \Delta_C^4]^\top$, where $\Delta_i^1 = w_i\big(\frac{\mathbf{z}_* - \mathbf{c}_i}{\Sigma_i}\big)$, $\Delta_i^2 = w_i\big(\frac{\mathbf{z}_* - \mathbf{c}_i}{\Sigma_i}\big)^2$, $\Delta_i^3 = \log\frac{p_*(1-p_i)}{p_i(1-p_*)}$, and $\Delta_i^4 = \frac{\sigma_*^2}{\sigma_i^2}$. We concatenate all of the new high-dimensional features from each pair of patches to form the final new high-dimensional feature for the pair of face images, and then compute the cosine similarity. The new high-dimensional facial features not only describe how the distribution of features of an un-seen face image differs from the distribution fitted to the features of all training images, but also encode the predictive information including the probabilities of label and uncertainty. We call this approach *GaussianFace-FE*.

Experimental Settings

In this section, we conduct experiments on face verification. We start by introducing the source-domain datasets and the target-domain dataset in all of our experiments (see Figure 2 for examples). The source-domain datasets include four different types of datasets as follows:

Multi-PIE (Gross et al. 2010). This dataset contains face images from 337 subjects under 15 view points and 19 illumination conditions in four recording sessions. These images are collected under controlled conditions.

MORPH (Ricanek and Tesafaye 2006). The MORPH database contains 55,000 images of more than 13,000 people within the age ranges of 16 to 77. There are an average of 4 images per individual.

Web Images[2]. This dataset contains around 40,000 facial

[1] For convenience, readers can also refer to the supplementary material for the inference.

[2] These two datasets are collected by our own from the Web. It is guaranteed that these two datasets are mutually exclusive with the LFW dataset.

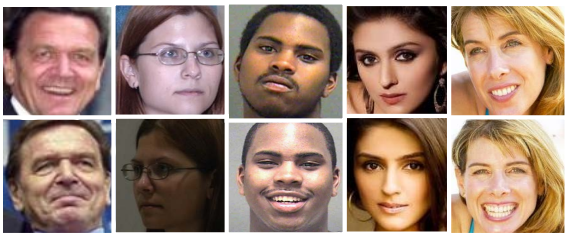

Figure 2: Samples of the datasets in our experiments. From left to right: LFW, Multi-PIE, MORPH, Web Images, and Life Photos.

images from 3261 subjects; that is, approximately 10 images for each person. The images were collected from the Web with significant variations in pose, expression, and illumination conditions.

Life Photos[2]. This dataset contains approximately 5000 images of 400 subjects collected online. Each subject has roughly 10 images.

If not otherwise specified, the target-domain dataset is the benchmark of face verification as follows:

LFW (Huang et al. 2007). This dataset contains 13,233 uncontrolled face images of 5749 public figures with variety of pose, lighting, expression, race, ethnicity, age, gender, clothing, hairstyles, and other parameters. All of these images are collected from the Web.

We use the LFW dataset as the target-domain dataset because it is well known as a challenging benchmark. Using it also allows us to compare directly with other existing face verification methods (Cao et al. 2013; Berg and Belhumeur 2012; Chen et al. 2013; Simonyan et al. 2013; Chen et al. 2012). Besides, this dataset provides a large set of relatively unconstrained face images with complex variations as described above, and has proven difficult for automatic face verification methods (Huang et al. 2007; Kumar et al. 2009). In all the experiments conducted on LFW, we strictly follow the standard unrestricted protocol of LFW (Huang et al. 2007). More precisely, during the training procedure, the four source-domain datasets are: Web Images, Multi-PIE, MORPH, and Life Photos, the target-domain dataset is the training set in View 1 of LFW, and the validation set is the test set in View 1 of LFW. At the test time, we follow the standard 10-fold cross-validation protocol to test our model in View 2 of LFW.

For each one of the four source-domain datasets, we randomly sample 20,000 pairs of matched images and 20,000 pairs of mismatched images. The training partition and the testing partition in all of our experiments are mutually exclusive. In other words, there is no identity overlap among the two partitions.

For the experiments below, "The Number of SD" means "the Number of Source-Domain datasets that are fed into the GaussianFace model for training". By parity of reasoning, if "The Number of SD" is i, that means the first i source-domain datasets are used for model training. Therefore, if "The Number of SD" is 0, models are trained with the training data from target-domain data only.

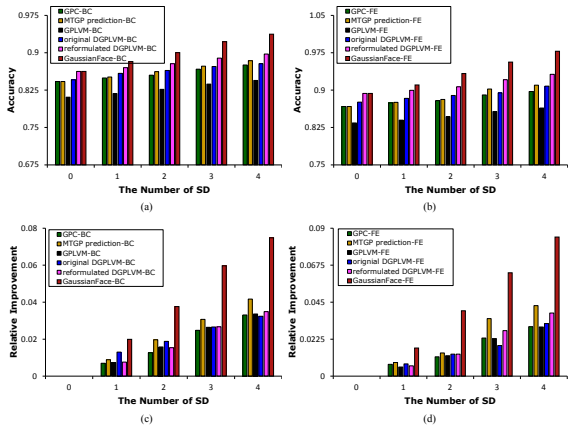

Figure 3: (a) The accuracy rate (%) of the GaussianFace-BC model and other competing MTGP/GP methods as a binary classifier. (b) The accuracy rate (%) of the GaussianFace-FE model and other competing MTGP/GP methods as a feature extractor. (c) The relative improvement of each method as a binary classifier with the increasing number of SD, compared to their performance when the number of SD is 0. (d) The relative improvement of each method as a feature extractor with the increasing number of SD, compared to their performance when the number of SD is 0.

Implementation details. Our model involves four important parameters: λ in (10), σ in (11), β in (17), and the number of anchors q [3]. Following the same setting in (Kim, Magnani, and Boyd 2006), the regularization parameter λ in (10) is fixed to 10^{-8}. σ reflects the tradeoff between our method's ability to discriminate (small σ) and its ability to generalize (large σ), and β balances the relative importance between the target-domain data and the multi-task learning constraint. Therefore, the validation set (the test set in View 1 of LFW) is used for selecting σ and β. Each time we use different number of source-domain datasets for training, the corresponding optimal σ and β should be selected on the validation set.

Since our model is based on the kernel method with a large number of image pairs for training (20,000 matched pairs and 20,000 mismatched pairs from each source-domain dataset), thus an important consideration is how to efficiently approximate the kernel matrix using a low-rank method in the limited space and time. We adopt the low rank method for kernel approximation. In our experiments, we take two steps to determine the number of anchor points. In the first step, the optimal σ and β are selected on the validation set in each experiment. In the second step, we fix σ and β, and then tune the number of anchor points. We vary the number of anchor points to train our model on the training set, and test it on the validation set. We report the average accuracy for our model over 10 trials. After we consider the trade-off between memory and running time in practice, the number of anchor points with the best average accuracy is determined in each experiments.

[3]The other parameters, such as the hyper-parameters in the kernel function, can be automatically learned from the data.

Experimental Results

In this section, we conduct five experiments to demonstrate the validity of the GaussianFace model.

Comparisons with Other MTGP/GP Methods

Since our model is based on GPs, it is natural to compare our model with four popular GP models: GPC (Rasmussen and Williams 2006), MTGP prediction (Bonilla, Chai, and Williams 2008), GPLVM (Lawrence 2003), and original DGPLVM (Urtasun and Darrell 2007). For fair comparisons, all these models are trained on multiple source-domain datasets using the same two methods as our GaussianFace model described in Section . After the hyper-parameters of covariance function are learnt for each model, we can regard each model as a binary classifier and a feature extractor like ours, respectively. Figure 3 shows that our model significantly outperforms the other four GPs models, and the superiority of our model becomes more obvious as the number of source-domain datasets increases. Here, GPC, GPLVM, and original DGPLVM cannot make the best of multiple source-domains. MTGP prediction has only considered the symmetric multi-task learning. In addition, it also did not consider the latent space of data.

In fact, Figure 3 (c) and (d) have demonstrated the validity of the multi-task learning constraint. To validate the effect of the reformulated KFDA, we also present the results of reformulated DGPLVM in Figure 3. Obviously, this reformulation is also necessary, which has around 2% improvement.

Comparisons with Other Binary Classifiers

Since our model can be regarded as a binary classifier, we have also compared our method with other classical binary classifiers. For this paper, we chose three popular representatives: SVM (Chang and Lin 2011), logistic regression (LR) (Fan et al. 2008), and Adaboost (Freund, Schapire, and Abe 1999). Table 1 demonstrates that the performance of our method GaussianFace-BC is much better than those of the other classifiers. Furthermore, these experimental results demonstrates the effectiveness of the multi-task learning constraint. For example, our GaussianFace-BC has about 7.5% improvement when all four source-domain datasets are used for training, while the best one of the other three binary classifiers has only around 4% improvement.

Comparisons with Other Feature Extractors

Our model can also be regarded as a feature extractor, which is implemented by clustering to generate a codebook. Therefore, we evaluate our method by comparing it with three popular clustering methods: K-means (Hussain et al. 2012), Random Projection (RP) tree (Dasgupta and Freund 2009), and Gaussian Mixture Model (GMM) (Simonyan et al. 2013). Since our method can determine the number of clusters automatically, for fair comparison, all the other methods generate the same number of clusters as ours. As shown in Table 2, our method GaussianFace-FE significantly outperforms all of the compared approaches, which verifies the effectiveness of our method as a feature extractor. The results have also proved that the multi-task learning constraint is effective. Each time one different type of source-domain dataset is added for training, the performance can be improved significantly. Our GaussianFace-FE model achieves over 8% improvement when the number of SD varies from 0 to 4, which is much higher than the $\sim$3% improvement of the other methods.

The Number of SD	0	1	2	3	4
SVM	83.21	84.32	85.06	86.43	87.31
LR	81.14	81.92	82.65	83.84	84.75
Adaboost	82.91	83.62	84.80	86.30	87.21
GaussianFace-BC	**86.25**	**88.24**	**90.01**	**92.22**	**93.73**

Table 1: The accuracy rate (%) of our method as a binary classifier and other competing methods on LFW using the increasing number of source-domain datasets.

The Number of SD	0	1	2	3	4
K-means	84.71	85.20	85.74	86.81	87.68
RP Tree	85.11	85.70	86.45	87.52	88.34
GMM	86.63	87.02	87.58	88.60	89.21
GaussianFace-FE	**89.33**	**91.04**	**93.31**	**95.62**	**97.79**

Table 2: The accuracy rate (%) of our method as a feature extractor and other competing methods on LFW using the increasing number of source-domain datasets.

Comparison with the state-of-art Methods

Motivated by the appealing performance of both GaussianFace-BC and GaussianFace-FE, we further explore to combine them for face verification. Specifically, after facial features are extracted using GaussianFace-FE, GaussianFace-BC [4] is used to make the final decision. Figure 4 shows the results of this combination compared with state-of-the-art methods (Cao et al. 2013; Chen et al. 2013; Simonyan et al. 2013; Sun, Wang, and Tang 2013; Taigman et al. 2014). The best published result on the LFW benchmark is 97.35%, which is achieved by (Taigman et al. 2014). Our GaussianFace model can improve the accuracy to 98.52%, which for the first time beats the human-level performance (97.53%, cropped) (Kumar et al. 2009). Figure 5 presents some example pairs that were always incorrectly classified by our model. Obviously, even for humans, it is also difficult to verify some of them. Here, we emphasize that only the alignment based on five landmarks and 200 thousand image pairs, instead of the accurate 3D alignment and four million images in (Taigman et al. 2014), are utilized to train our model. This makes our method simpler and easier to use. Readers can refer to the supplementary material for more experimental results.

General Discussion

There is an implicit belief among many psychologists and computer scientists that human face verification abilities are currently beyond existing computer-based face verification algorithms (O'Toole et al. 2006). This belief, however, is supported more by anecdotal impression than by scientific evidence. By contrast, there have already been a number of papers comparing human and computer-based face verification performance (Tang and Wang 2004; O'Toole et al. 2007;

[4] Here, the GaussianFace BC is trained with the extracted high-dimensional features using GaussianFace-FE.

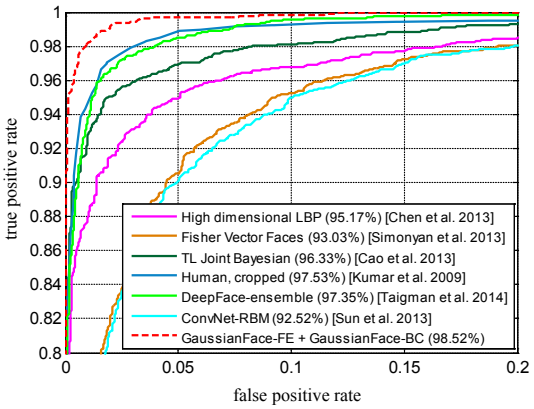

Figure 4: The ROC curve on LFW. Our method achieves the best performance, beating human-level performance.

Phillips and O'Toole 2014). It has been shown that the best current face verification algorithms perform better than humans in the good and moderate conditions. So, it is really not that difficult to beat human performance in some specific scenarios.

As pointed out by (O'Toole et al. 2012; Sinha et al. 2005), humans and computer-based algorithms have different strategies in face verification. Indeed, by contrast to performance with unfamiliar faces, human face verification abilities for familiar faces are relatively robust to changes in viewing parameters such as illumination and pose. For example, Bruce (Bruce 1982) found human recognition memory for unfamiliar faces dropped substantially when there were changes in viewing parameters. Besides, humans can take advantages of non-face configurable information from the combination of the face and body (e.g., neck, shoulders). It has also been examined in (Kumar et al. 2009), where the human performance drops from 99.20% (tested using the original LFW images) to 97.53% (tested using the cropped LFW images). Hence, the experiments comparing human and computer performance may not show human face verification skill at their best, because humans were asked to match the cropped faces of people previously unfamiliar to them. To the contrary, those experiments can fully show the performance of computer-based face verification algorithms. First, the algorithms can exploit information from enough training images with variations in all viewing parameters to improve face verification performance, which is similar to information humans acquire in developing face verification skills and in becoming familiar with individuals. Second, the algorithms might exploit useful, but subtle, image-based detailed information that give them a slight, but consistent, advantage over humans.

Therefore, surpassing the human-level performance may only be symbolically significant. In reality, a lot of challenges still lay ahead. To compete successfully with humans, more factors such as the robustness to familiar faces and the usage of non-face information, need to be considered in developing future face verification algorithms.

Figure 5: The two rows present examples of matched and mismatched pairs respectively from LFW that were incorrectly classified by the GaussianFace model.

Conclusion and Future Work

This paper presents a principled Multi-Task Learning approach based on Discriminative Gaussian Process Latent Variable Model, named *GaussianFace*, for face verification by including a computationally more efficient equivalent form of KFDA and the multi-task learning constraint to the DGPLVM model. We use Gaussian Processes approximation and anchor graphs to speed up the inference and prediction of our model. Based on the GaussianFace model, we propose two different approaches for face verification. Extensive experiments on challenging datasets validate the efficacy of our model. The GaussianFace model finally surpassed human-level face verification accuracy, thanks to exploiting additional data from multiple source-domains to improve the generalization performance of face verification in the target-domain and adapting automatically to complex face variations.

Although several techniques such as the Laplace approximation and anchor graph are introduced to speed up the process of inference and prediction in our GaussianFace model, it still takes a long time to train our model for the high performance. In addition, large memory is also necessary. Therefore, for specific application, one needs to balance the three dimensions: memory, running time, and performance. Generally speaking, higher performance requires more memory and more running time. In the future, the issue of running time can be further addressed by the distributed parallel algorithm or the GPU implementation of large matrix inversion. To address the issue of memory, some online algorithms for training need to be developed. Another more intuitive method is to seek a more efficient sparse representation for the large covariance matrix.

References

Ahonen, T.; Hadid, A.; and Pietikainen, M. 2006. Face description with local binary patterns: Application to face recognition. *TPAMI*.

Ben-Hur, A.; Horn, D.; Siegelmann, H. T.; and Vapnik, V. 2002. Support vector clustering. *JMLR*.

Berg, T., and Belhumeur, P. N. 2012. Tom-vs-pete classifiers and identity-preserving alignment for face verification. In *BMVC*.

Bonilla, E.; Chai, K. M.; and Williams, C. 2008. Multi-task gaussian process prediction. In *NIPS*.

Bruce, V. 1982. Changing faces: Visual and non-visual coding processes in face recognition. *British Journal of Psychology*.

Cao, Z.; Yin, Q.; Tang, X.; and Sun, J. 2010. Face recognition with learning-based descriptor. In *CVPR*.

Cao, X.; Wipf, D.; Wen, F.; and Duan, G. 2013. A practical transfer learning algorithm for face verification. In *ICCV*.

Chang, C.-C., and Lin, C.-J. 2011. Libsvm: a library for support vector machines. *ACM TIST*.

Chen, D.; Cao, X.; Wang, L.; Wen, F.; and Sun, J. 2012. Bayesian face revisited: A joint formulation. In *ECCV*.

Chen, D.; Cao, X.; Wen, F.; and Sun, J. 2013. Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification. In *CVPR*.

Damianou, A., and Lawrence, N. 2013. Deep gaussian processes. In *AISTATS*.

Dasgupta, S., and Freund, Y. 2009. Random projection trees for vector quantization. *IEEE Transactions on Information Theory*.

Fan, R.-E.; Chang, K.-W.; Hsieh, C.-J.; Wang, X.-R.; and Lin, C.-J. 2008. Liblinear: A library for large linear classification. *JMLR*.

Freund, Y.; Schapire, R.; and Abe, N. 1999. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*.

Gross, R.; Matthews, I.; Cohn, J.; Kanade, T.; and Baker, S. 2010. Multi-pie. *Image and Vision Computing*.

Higham, N. J. 1996. *Accuracy and Stability of Numberical Algorithms*. Siam.

Huang, G. B.; Ramesh, M.; Berg, T.; and Learned-Miller, E. 2007. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, University of Massachusetts, Amherst.

Hussain, S. U.; Napoléon, T.; Jurie, F.; et al. 2012. Face recognition using local quantized patterns. In *BMVC*.

Kim, H.-C., and Lee, J. 2007. Clustering based on gaussian processes. *Neural computation*.

Kim, S.-J.; Magnani, A.; and Boyd, S. 2006. Optimal kernel selection in kernel fisher discriminant analysis. In *ICML*.

Kumar, N.; Berg, A. C.; Belhumeur, P. N.; and Nayar, S. K. 2009. Attribute and simile classifiers for face verification. In *ICCV*.

Lawrence, N. D. 2003. Gaussian process latent variable models for visualisation of high dimensional data. In *NIPS*.

Li, Z.; Liu, W.; Lin, D.; and Tang, X. 2005. Nonparametric subspace analysis for face recognition. In *CVPR*.

Li, H.; Hua, G.; Lin, Z.; Brandt, J.; and Yang, J. 2013. Probabilistic elastic matching for pose variant face verification. In *CVPR*.

Liu, C., and Wechsler, H. 2002. Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition. *TIP*.

Liu, W.; He, J.; and Chang, S.-F. 2010. Large graph construction for scalable semi-supervised learning. In *ICML*.

Lowe, D. G. 2004. Distinctive image features from scale-invariant keypoints. *IJCV*.

Lu, C., and Tang, X. 2014. Learning the face prior for bayesian face recognition. In *ECCV*.

Lu, C.; Zhao, D.; and Tang, X. 2013. Face recognition using face patch networks. In *ICCV*.

Moghaddam, B.; Jebara, T.; and Pentland, A. 2000. Bayesian face recognition. *Pattern Recognition*.

O'Toole, A. J.; Jiang, F.; Roark, D.; and Abdi, H. 2006. Predicting human performance for face recognition. *Face Processing: Advanced Methods and Models. Elsevier, Amsterdam*.

O'Toole, A. J.; Phillips, P. J.; Jiang, F.; Ayyad, J.; Pénard, N.; and Abdi, H. 2007. Face recognition algorithms surpass humans matching faces over changes in illumination. *TPAMI*.

O'Toole, A. J.; An, X.; Dunlop, J.; Natu, V.; and Phillips, P. J. 2012. Comparing face recognition algorithms to humans on challenging tasks. *ACM Transactions on Applied Perception*.

Phillips, P. J., and O'Toole, A. J. 2014. Comparison of human and computer performance across face recognition experiments. *Image and Vision Computing*.

Rasmussen, C. E., and Williams, C. K. I. 2006. Gaussian processes for machine learning.

Ricanek, K., and Tesafaye, T. 2006. Morph: A longitudinal image database of normal adult age-progression. In *Automatic Face and Gesture Recognition*.

Simonyan, K.; Parkhi, O. M.; Vedaldi, A.; and Zisserman, A. 2013. Fisher vector faces in the wild. In *BMVC*.

Sinha, P.; Balas, B.; Ostrovsky, Y.; and Russell, R. 2005. Face recognition by humans: 20 results all computer vision researchers should know about. *Department of Brain and Cognitive Sciences, MIT, Cambridge, MA*.

Sun, Y.; Wang, X.; and Tang, X. 2013. Hybrid deep learning for face verification. In *ICCV*.

Sun, Y.; Wang, X.; and Tang, X. 2014. Deep learning face representation from predicting 10,000 classes. In *CVPR*.

Taigman, Y.; Yang, M.; Ranzato, M.; and Wolf, L. 2014. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. *CVPR*.

Tang, X., and Wang, X. 2004. Face sketch recognition. *IEEE Transactions on Circuits and Systems for Video Technology*.

Torralba, A., and Efros, A. A. 2011. Unbiased look at dataset bias. In *CVPR*.

Turk, M. A., and Pentland, A. P. 1991. Face recognition using eigenfaces. In *CVPR*.

Urtasun, R., and Darrell, T. 2007. Discriminative gaussian process latent variable model for classification. In *ICML*.

Wright, J., and Hua, G. 2009. Implicit elastic matching with random projections for pose-variant face recognition. In *CVPR*.