A Gabor Filterbank Approach for Face Recognition and Classification Using Hybrid Metric Learning

Alican Nalci
Electrical and Computer Engineering
University of California, San Diego
La Jolla, California 92092
analci@eng.ucsd.edu

Shouvik Ganguly
Electrical and Computer Engineering
University of California, San Diego
La Jolla, California 92093
shgangul@eng.ucsd.edu

Abstract—In this paper, we consider the notion of distance/similarity metric learning method for classification and similarity search for face recognition purposes. We use the theory of filter banks, and Gabor wavelets for extraction of face features in three datasets: the ORL dataset, Yale face dataset, and ATT face dataset, and we compare the recognition performance with Gabor features to discrete Fourier transform baseline method for feature extraction. For the learning application part, there are various different approaches proposed under the distance metric learning framework. We base our work on two different approaches: i) Distance metric learning for LMNN [1], ii) Information Theoretical Metric Learning (ITML) [2], and. Our main goal is, employing optimization-based techniques, to combine two distance metrics into a single hybrid metric explaining the similarity/dissimilarity relation better than each of its base metrics. Since the selection of features impacts the classification and recognition results, we propose that Gabor features obtained using a parallel bank of Gabor filter at different scales and orientations provides the best recognition rate. Furthermore, we attempt to perform the objective of collapsing two distance/similarity metrics into a single one by optimizing an appropriate weighting which enables the hybrid metric to inherit the useful characteristics of its component metrics in terms of classification accuracy with similarity/distance information. We conduct experiments with hybrid metric approach on various datasets to compare its performance to that of each of its components individually. To this end, we first provide a general background overview on Gabor filters for feature extraction, distance metrics and some special family of convex optimization problems along with the metric learning approaches introduced in [1], [2], and [3]. Finally, we introduce our hybrid metric learning approach and present our experimental results obtained from the face data-sets.

I. INTRODUCTION

This paper involves two studies that heavily depend on selection of good features for face recognition and classification purposes. First study involves the extraction of good features from face images using a bank of parallel Gabor Filters and presents quantitative comparisons with a baseline method of Discrete Fourier Transform for feature extraction. The second study investigates how the two feature extraction methods actually perform with three different metric learning algorithms on face datasets. We show that, the features obtained by applying a parallel bank of Gabor Filters at different orientations and scales provide the best results for our metric learning algorithms as the face recognition rate we obtain is contemporary to recent face recognition papers, and is significantly higher than the baseline method. The method we describe in this paper is useful in many fields such as pattern recognition, face recognition and activity analysis. Performance of such applications depend critically on the choice of features that are used to train the classifier [1]. Therefore, to achieve good classification results, obtaining good features is an important challenge, therefore we base our work on three different such approaches: i) Distance metric learning for LMNN [4], Information Theoretical Metric Learning (ITML) [5], and (iii) our Hybrid Metric Learning approach. We conduct experiments with our Gabor filterbank based hybrid metric approach on various image datasets to compare its performance to that of each of its ancestor components individually, and with a baseline methods Discrete Fourier transform for feature extraction. The idea presented in this paper can be extended to similarity search applications, since the trained weight matrix transforms the input
data such that the most similar images are closer to each other as points in the k-Euclidean space. Meaning that, given a specific image the application returns the most similar images.

In several areas of machine learning, the notion of metric naturally arises as a machinery to measure the distance or similarity between a pair of objects in question. In objectives like clustering, classification, ranking, and k-nearest neighbors (a.k.a kNN), learning an appropriate distance metric from labeled examples, and performing the objective based on the learned metric has been a significant and promising method. For instance, performing kNN classification with metric learning approach relies substantially on the power of the metric in explaining the distances/similarities between different examples. Most trivial way of measuring this similarity would be considering the similarity as inverse proportional to Euclidean distance between vector representations of two data points in \( R^d \). However, in general, this approach was proven to be prone to perform poorly since some features may be more important than others in terms of similarity information and hence need to be scaled accordingly. High-level idea in the problem of distance metric learning is to learn a linear transformation of the data points in \( R^d \), application of which enforces the similar points to get close to each other while pushing the dissimilar points away from each other based on the extracted similarity information from supervised data. In other words, the main idea is to reverse engineer the objective of specific application through hypothetical distance/similarity measure that we want to learn.

In this paper, we focus on optimization-based distance metric learning approach for classification and similarity search applications. To this end, we first provide the necessary mathematical and algorithmic background for distance metric learning along with the formulation and details of convex optimization with quadratic, and semi-definite programming. With the formulation and details of convex optimization background for distance metric learning along first provide the necessary mathematical and algorithmic background for distance metric learning along with the formulation and details of convex optimization. After providing the required background, we present general overviews for these three approaches along with their formulations as optimization problems and the respective algorithms solving them. In Section 3, we provide the details and formulations of our attempts to combine two learned metrics into a hybrid distance metric to achieve a better practical performance for classification and similarity search purposes. In Section 4, we describe datasets and experiments, and in section 5, we discuss the experimental results and compare these three famous approaches with our hybrid model.

II. BACKGROUND

In this section, we provide the technical background material.

A. Gabor Filter Bank

The Gabor filters are a class of filters that have been used with considerable success \([13], [17], [14]\) for image processing. The main reason for their success is that they show robustness against variations in illumination, rotation, scale and translation. They can also withstand image noise better than most other filters in literature \([15]\). Further, the frequency and orientation representations of the Gabor filters are similar to the human visual system. All these factors make the Gabor filters an attractive tool for applications like text and face recognition.

In the spatial domain, a 2D Gabor filter can be represented as

\[
G_{f, \theta, \gamma, \eta}(x, y) = \frac{f^2}{\pi \gamma \eta} \exp \left[ -\left( \alpha^2 x^2 + \beta^2 y^2 \right) + j2\pi f x' \right]
\]

where,

\[
\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]

where \( f \) is the center frequency of a sinusoidal plane wave modulating the Gaussian filter, \( \theta \) is the anticlockwise rotation of the Gaussian and the plane wave, \( \alpha \) is the sharpness of the Gaussian profile along the direction parallel to the wave, and \( \beta \) is the sharpness along the direction perpendicular to the wave. \( \gamma = \frac{f}{\alpha} \) and \( \eta = \frac{f}{\beta} \) are defined to keep the ratio of frequency to sharpness constant. We observe from the above equation that all Gabor filters can be generated from a single filter by rotation and dilation, i.e. they are all self-similar. To extract useful features from an image, usually a set of Gabor filters with different frequencies and orientations are required:

\[
H_{u, v} \equiv G_{f_u, \theta_v, \gamma, \eta}.
\]

We have taken \( f_u = \frac{f_{\text{max}}}{2^{u/2}}, 0 \leq u \leq 4 \), and \( \theta_v = \frac{\pi}{8} v, 0 \leq v \leq 7 \), i.e. in total, 40 Gabor filters at
5 different scales and 8 different orientations. The block diagram for the analysis using the 40 filters is as follows.

The parameters $f_{\text{max}}$, $\gamma$ and $\eta$ are still left to be determined. While, in principle, we could take $f_{\text{max}}$ to be as large as 0.5, useful information in mainly contained in the low frequency band. Thus $f_{\text{max}}$ is taken to be 0.25, as suggested in [16]. We also take $\gamma = \eta = \sqrt{2}$, as used in most Gabor filter applications.

### B. DFT-based Filter Bank

Another set of filters used for comparison are 2-dimensional DFT-based filterbanks. The one dimensional equivalent of the filterbank can be explained by the following block diagram.

For the 2D analogue, we have to replace the polyphase components by the two-dimensional polyphase components $E_{00}$, $E_{01}$, etc. and the DFT matrix by the 2D DFT matrix. The fast computational method known as the FFT produces the same results much faster, so during the simulations we use that.

### C. Further Processing and PCA

If $I(x,y)$ be the image, the output of the $(u,v)^{th}$ filter (before down sampling) can be writen as

$$I_{u,v}(x,y) = (I * H_{u,v})(x,y).$$

These outputs are then downsampled by $M = 2$ along each dimension for the Gabor filters, normalized to zero mean and unit variance, and turned into a vector $z_{u,v}$ by concatenating the rows. A feature vector $z$ is formed by concatenating all these vectors $z_{u,v}$ for all $u$ and $v$.

We compute the feature vectors $z$ for all the images in a particular dataset and stack them to form a matrix $X$, whose each row is the feature vector for one particular image. Principal Components Analysis (PCA) uses an orthogonal transformation to reduce the dimension of a set of observations of possibly correlated variables. We use PCA to convert $X$ into a matrix of smaller size, as explained in [12]. We adjust the dimensionality so as to include 99% of the variance of the data.

### D. Metric Learning with Mahalanobis Distance

A metric is an essential tool that allows us to relate any two points to each other in a vector space. In other words, it is a map from pairs of elements in a vector space to real numbers. Formally, on a set $X$, a map $d : X \times X \rightarrow R$ is called metric if

- $d(x,y) \geq 0$
- $d(x,y) = 0 \iff x = y$
- $d(x,y) = d(y,x)$
- $d(x,y) + d(y,z) \geq d(x,z)$

for all $x,y,z$ in $X$. Here, first and second conditions stand for positive semi-definiteness, third one enforces symmetry, and the last condition is known as the triangle inequality. One can note here that it is possible to define various different metrics on the same vector space. For instance, if $X$ is a normed vector space, then $d(x,y) = \|x - y\|$ defines a valid metric. This is how we will interpret the metric in our work. Let $S^d$ denote the set of symmetric matrices in $R^{d \times d}$, and let $S^d_+ \subset S^d$ denote the set of $d \times d$ positive semi-definite matrices. We call the following metric as Mahalanobis distance:

$$d_M(x,y) = \sqrt{(x-y)^T M (x-y)}$$

with respect to $M \in S^d_+$. Since $M$ is PSD matrix, one can consider its Cholesky decomposition $L^T L = M$.

Rewriting Mahalanobis distance

$$d_M(x,y) = \sqrt{(x-y)^T L^T L (x-y)}$$

$$= \sqrt{(L(x-y))^T L (x-y)}$$

$$= \sqrt{||L(x-y)||^2_2}$$

$$= ||L(x-y)||_2,$$
It is important to note here that Mahalanobis distance between two points \( x, y \in X \) can also be interpreted as the Euclidean distance between the points \( Lx, Ly \) in the transformed space by linear transformation \( L \).

We now briefly discuss the way metric learning is performed using the supervised data that contains some information regarding the similarity. Let \( \{x_1, x_2, \ldots, x_n\} \) be a set of training examples with some kind of similarity notion. Similarity information can be given in the form of similar and dissimilar pairs of examples, or as an explicit labels \( \{y_1, y_2, \ldots, y_n\} \) corresponding to each example. In the first case, we are given two sets \( S \) and \( D \) corresponding to similar and dissimilar examples, respectively. Formally, each pair \((i, j) \in S \) implies that examples \( x_i \) and \( x_j \) are similar, and \((i, j) \in D \) implies that examples \( x_i \) and \( x_j \) are dissimilar. If instead explicit labels are provided, sets \( S \) and \( D \) may be formed various different ways depending on the objective and application. One trivial way would be as follows: \((i, j) \in S \) if \( y_i = y_j \) and \((i, j) \in D \) otherwise.

\[
(i, j) \in S, \text{ if } y_i = y_j \\
(i, j) \in D, \text{ otherwise.}
\]

As we hinted before, a natural idea in learning the metric is to enforce similar examples to get close to each other while pushing the dissimilar ones away from each other in terms of Euclidean distance in the transformed space. There are several approaches to formulate this high-level goal in terms of metric learning problem depending on the application and respective more specific objective. As an example to the reader, we provide the following formulation as in [7]:

\[
\begin{align*}
\text{maximize} &\quad \sum_{(i,j) \in D} d_M(x_i, x_j) \\
\text{subject to} &\quad \sum_{(i,j) \in S} d_M(x_i, x_j)^2 \leq 1 \\
&\quad M \geq 0.
\end{align*}
\]

It is important to note here that this optimization problem can be solved efficiently since it is a convex problem. In the upcoming sections, there will appear optimization formulations that are non-convex, and hence need some special methods or extra effort to be solved.

III. DISTANCE/SIMILARITY METRIC LEARNING APPROACHES

A. Large Margin Nearest Neighbor Classification

In [1], Weinberger and Saul introduce an optimization-based formulation for learning a Mahalanobis distance metric for classification from labeled examples. Their formulation is based on the idea of embedding the \( kNN \) decision rule in metric learning phase. More precisely, the algorithm is designed in a way that \( k \) nearest neighbors of an example with respect to the resulting learned metric are imposed to share the same label. This high-level objective is achieved through expressing it in terms of objective and constraint functions, and translating the whole picture into an appropriate optimization framework.

We now overview the process of translation proposed by the authors of [1]. They define the notions of target and impostor. Target neighbors of example \( x_i \) are defined to be the examples \( x_j \) that are desired to be closest to \( x_i \). In other words, the attempt is to learn a linear transformation \((L)\) of the input space, so that the resulting closest neighbors of \( x_i \); based on the Euclidean distance in the transformed space are its target neighbors. This target neighbor relation is denoted by the following notation: \( j \sim i \) if \( x_j \) is a target neighbor of \( x_i \). It is important to note that establishing this relation requires a side information in addition to the labels of examples. Authors suggest using Euclidean distance in the original input space to establish target neighbor relations in the absence of auxiliary information. In order to succeed the high-level objective described above, no training example with different label should be closer than any target neighbor of \( x_i \) in the transformed space. In this regard, authors define impostors of an example \( x_i \) as the set of all examples that do not share the same label with \( x_i \). These are the examples that are desired to be pushed away from the closest neighborhood of \( x_i \). More precisely, the goal of learning becomes maintaining a large margin between impostors and perimeters established by target neighbors.

In the formulation, the loss function is consisting of two parts, first of which imposes penalty on large distances between target neighbors, and the second penalizes the small distance between examples and their impostors. In other words, the first component of loss function attempts to pull target neighbors together while the second term pushes impostors
apart from the target neighborhood. Formally, 
\[
\epsilon_{\text{pull}}(M) = \sum_{i,j\neq i} d_M(x_i, x_j)^2 \\
= \sum_{i,j\neq i} ||x_i - x_j||_M^2 \\
= \sum_{i,j\neq i} D_M(x_i, x_j)
\]
where \( D_M(x_i, x_j) \) is the square Mahalanobis distance with respect to \( M \). The second term, corresponding to push, is defined as follows:
\[
\epsilon_{\text{push}}(M) = \sum_{i,j\neq i} \sum_{l \in \text{impostor}} [1 + \|x_i - x_j\|_M^2 - \|x_i - x_l\|_M^2] \\
= \sum_{i,j\neq i} \sum_{l=1}^{y_{il}} (1 - y_{il}) \left[ 1 + D_M(x_i, x_j) - D_M(x_i, x_l) \right]
\]
where \( y_{il} = 1 \) if \( x_i = y_l \) and \( y_{il} = 0 \) otherwise, and \( [w]_+ = \max(w, 0) \). This term is basically the sum of all margin violations by impostors: for an example \( x_i \) and its target neighbor \( x_j \), if \( 1 + D_M(x_i, x_j) > D_M(x_i, x_l) \) for an impostor \( x_l \), it adds a penalty of how much the impostor \( x_l \) violates its margin to neighborhood of \( x_i \) with respect to target neighbor \( x_j \).

Having defined these two terms, the overall loss function is defined as the weighted sum of \( \epsilon_{\text{pull}} \) and \( \epsilon_{\text{push}} \):
\[
\epsilon(M) = (1 - \mu)\epsilon_{\text{pull}}(M) + (\mu)\epsilon_{\text{push}}(M)
\]

Hence, the goal of learning becomes to minimize the loss function defined above. Note that the same loss function above can also be defined in terms of matrix \( L \), if we replace \( D_M(x_i, x_j) \) by \( D_L(x_i, x_j) = \|L(x_i - y)\|_2^2 \), where \( L^T L = M \). However, in this setting, which is the original formulation of the authors in [1], the problem becomes hard to solve since the problem is non-convex, and thus it may have convergence issues. On the other hand, formulating the minimization of objective above as an optimization problem in matrix \( M \), which must be a positive-semidefinite matrix, leads to a solvable optimization problem which is convex. More precisely, handling the hinge loss \([w]_+ \) by introducing a non-negative slack variable \( s_{ijl} \) for each \((i,j,l)\) triplet corresponding to an example \( x_i \), its target neighbor \( x_j \), and impostor \( x_l \), to be able to measure the margin violation, the problem becomes:
\[
\begin{align*}
\text{minimize} & \quad (1 - \mu) \sum_{i,j\neq i} (x_i - x_j)^T M (x_i - x_j) + \mu \sum_{i,j\neq i} (1 - y_{il}) s_{ijl} \\
\text{subject to} & \quad (x_i - x_j)^T M (x_i - x_j) - (x_i - x_l)^T M (x_i - x_l) \geq 1 - s_{ijl}, \\
& \quad s_{ijl} \geq 0, \\
& \quad M \succeq 0.
\end{align*}
\]
This formulation is now a semidefinite programming and it can be solved by on the shelf programming.

B. Information Theoretic Metric Learning

In this section, we briefly provide the underlying idea incorporated in information theoretic metric learning approach introduced in [2], and the corresponding optimization-based formulation of the problem. The core idea in this approach is to use Kullback-Leibler divergence to measure the distance between two Mahalanobis distance functions. More precisely, in Davis et. al, the high-level objective in this approach is to learn a positive definite matrix \( A \) that satisfies the desired similarity constraints and is close to a given prior positive definite matrix \( A_0 \). Thus, in learning process, they measure the distance/similarity between examples by the Mahalanobis distance defined by matrix \( A \) to be learned, and the distance between Mahalanobis distances, defined for example by matrices \( A \) and \( A_0 \), by Kullback-Leibler divergence of corresponding two Gaussian distributions \( p(x; A) \) and \( p(x; A_0) \) that will be defined below.

To this end, a simple bisection between the set of Mahalanobis distances and the set of equal-mean Gaussian distributions. Formally, for a Mahalanobis distance \( d_A(\cdot, \cdot) \) defined by \( A \), they define its corresponding multivariate Gaussian as \( p(x; A) = \frac{1}{\sqrt{2\pi \det A}} \exp(-\frac{1}{2}d_A(x, \mu)) \), where \( \mu \) is the common mean, \( Z \) is normalizing constant, and \( A^{-1} \) is the covariance of the distribution. Under this setting, they define the distance between two Mahalanobis distance functions defined by matrices \( A \) and \( A_0 \) as
\[
KL(p(x; A) || p(x; A_0)) = \int p(x; A_0) \log \frac{p(x; A_0)}{p(x; A)} dx.
\]
Given pairs of similar and dissimilar examples \( S \) and \( D \), respectively, distance metric learning problem can be defined as follows:
\[
\begin{align*}
\text{minimize} & \quad KL(p(x; A) || p(x; A_0)) \\
\text{subject to} & \quad d_A(x_i, x_j) \leq u \quad (i,j) \in S \\
& \quad d_A(x_i, x_j) \geq l \quad (i,j) \in D
\end{align*}
\]
The authors then show the way that the information theoretic objective above is expressed as a particular type of Bregman divergence. The metric learning problem above is then solved by adapting Bregman’s method in [10]. More precisely, the metric learning problem is expressed as LogDet optimization. To this end, for \( n \times n \) matrices \( A, A_0 \), let
learning is equivalently expressed as

$$KL(p(x; A)||p(x; A_0)) = \frac{1}{2} D_{ld}(A, A_0)$$

Using this identity, optimization problem for metric learning is equivalently expressed as

$$\text{minimize} \quad D_{ld}(A, A_0)$$

subject to

$$\text{tr}(A(x_i - x_j)(x_i - x_j)^T) \leq u \quad (i, j) \in S$$

$$\text{tr}(A(x_i - x_j)(x_i - x_j)^T) \geq l \quad (i, j) \in D$$

$$A \succeq 0$$

In order to avoid from scenarios in which there may not exist solution to the optimization formulation above, slack variables $s_{c(i,j)}$ corresponding to $c(i,j)$-th constraint are incorporated. Let $s$ denote the vector representation of slack variables, and let $s_0$ denote the initial slack vector whose $c(i,j)$-th component is $u$ if $(i,j) \in S$, and $l$ if $(i,j) \in D$. Then, the final version of the optimization formulation of information theoretic objective described above as LogDet optimization becomes

$$\text{minimize} \quad D_{ld}(A, A_0) + \gamma \cdot D_{ld}(diag(s), diag(s_0))$$

subject to

$$\text{tr}(A(x_i - x_j)(x_i - x_j)^T) \leq s_{c(i,j)} \quad (i, j) \in S$$

$$\text{tr}(A(x_i - x_j)(x_i - x_j)^T) \geq s_{c(i,j)} \quad (i, j) \in D$$

where, $\gamma$ is weighting between minimizing the objective and satisfying the constraints. The formulation representing the high-level objective of this approach above is a convex optimization problem in its variables. Thus, efficient local-free-minima algorithms can be used to solve it. However, since the scale of the datasets we experiment ITML approach on are large, general purpose solvers remain inefficient in terms of computation time. That is why, we use the special-purpose solver implemented by the authors for our experiments.

C. Hybrid Metric Learning

We use the same notation and setup as described in other metric learning approaches above. Let’s first describe the high-level idea: suppose we are given two learned linear transformations $L_1, L_2 : R^m \rightarrow R^d$ from training examples $\{x_i\}_{i=1}^n$ in $R^m$ with pairs of similar and dissimilar sets $S$ and $D$. Note that each of $M_1 = L_1^T L_1$ and $M_2 = L_2^T L_2$ defines a Mahalanobis distance in the usual way. Let $d_{M_1}(\cdot, \cdot)$ and $d_{M_2}(\cdot, \cdot)$ denote these Mahalanobis distances defined by $M_1$ and $M_2$, respectively. Define $m \times d$ matrix $L_h(\alpha) = \alpha L_1 + (1 - \alpha) L_2$ for scalar $0 \leq \alpha \leq 1$, and $M_h(\alpha) = L_h(\alpha)^T L(\alpha)$. Let $d_{M_h}(\cdot, \cdot)$ denote the Mahalanobis distance defined by matrix $M_h(\alpha)$. We call

$$d_{M_h}(\cdot, \cdot) = \sqrt{(x - y)^T M_h(\alpha)(x - y)}$$

the hybrid metric of two metrics $d_{M_1}(\cdot, \cdot)$ and $d_{M_2}(\cdot, \cdot)$ with respect to hybrid weight $\alpha$.

With this setting, our call is to learn the best $\alpha$ to combine these two learned metrics into one. In other words, we want to optimize the value of $\alpha$ in a way that hybrid metric inherits and combine good properties explaining similarity relation of each of its base metrics. We adapt a optimization formulation that is similar to the one in cosine similarity learning. Given $L_1, L_2$, and training examples, we formulate our objective as the following optimization problem:

$$\text{minimize} \quad \sum_{(i,j) \in S} D_{M_h}(\alpha)(x, y) - \beta \sum_{(i,j) \in D} D_{M_h}(\alpha)(x, y)$$

subject to $0 \leq \alpha \leq 1$

where $D_{M_h}(\alpha)(x, y) = d_{M_h}(\cdot, \cdot)^2$ as in LMNN approach, and $\beta$ is a balancing factor between the contributions of similar and dissimilar samples to the margin we want to maximize. We set $\beta$ to be $\frac{|S|}{|D|}$ in our experiments. It is important to note here that the optimization problem above is quadratic programming in the optimization variable $\alpha$. Thus, it can be solved very efficiently by QP solvers. Moreover, we try to optimize a scalar optimization variable $\alpha$, which enables solver tools to find the optimal solution even more efficiently. Let $\alpha_h$ denote the optimal solution to the optimization problem above. Let $d_{h}(\cdot, \cdot)$ denote the Mahalanobis distance defined by $M_h(\alpha_h)$. We call $\alpha_h$ as hybrid weight and, call $d_{h}(\cdot, \cdot)$ as hybrid metric. For the sake of completeness of notation in the rest of our discussion, let $M_h$ and $L_h$ denote the matrices $M_h(\alpha_h)$, and $M_h(\alpha_h)$ computed at the optimal $\alpha_h$.

IV. Experiment Design

A. Gabor Filter Bank

We used 40 Gabor filters in total, at 5 different scales and 8 different orientations. We have taken the center frequencies and orientations as

$$f_u = \frac{f_{\max}}{2^u/2}, \quad 0 \leq u \leq 4,$$

$$\theta_v = \frac{\pi}{8} v, \quad 0 \leq v \leq 7.$$
We also took $f_{\text{max}} = 0.25$, as in [16]. The magnitudes and real parts of the filters are plotted in Figures 1 and 2, respectively.

**Fig. 1:** Magnitudes of the Gabor Filters; each row represents a particular scale

**Fig. 2:** Real Parts of the Gabor Filters; each row represents a particular scale

**B. DFT as a Baseline Method**

**C. Dataset Selection**

Three datasets have been carefully selected for performance evaluation of this method.

**Yale Face Dataset:** The Yale face dataset consists of 165 grayscale images of 15 individuals. There are 11 images per subject, one for each different facial expression or configuration: center-light, with glasses, happy, left-light, without glasses, normal, right-light, sad, sleepy, surprised, and wink. Each image is $32 \times 32$ pixels.

**AT&T Dataset:** The AT&T face dataset has 400 grayscale images of 40 individuals. There are 10 images per subject, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses). Each image is $112 \times 92$ pixels.

**ORL Dataset:** The ORL face dataset is essentially the same as the AT&T dataset, however each image is cropped to exclude other features such as hair, and concentrates only on the face. Each image is $64 \times 64$ pixels.

**D. Metric Learning**
V. RESULTS

We compare the performance of metric learning algorithms using Gabor and DFT features in terms of recognition rate, similarity/clustering performance and visual recognition accuracy. For all of the datasets described in section IV-C we evaluate our hybrid metric learning algorithm using kNN classification with $k = 4$, and the results presented in this paper are showing the mean of five consecutive runs.

Table I shows the test set face recognition rates using the Gabor features and the three metric learning algorithms. We see that, the recognition rates are significantly high using LMNN and Hybrid algorithms. Hybrid metric learning is slightly better compared to the LMNN, and ITML produces the worst results among the three algorithms. We achieve the highest recognition rate of 100% on Yale faces data-set with the Hybrid method, since it is a relatively easier data-set. Moreover, the general recognition rates are satisfactory with LMNN and Hybrid method, so we conclude that using a bank of Gabor filters for feature extraction we can achieve high face recognition rates using LMNN and Hybrid metric learning methods. Next we compare the recognition rate using Gabor features to recognition rate using DFT features.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>LMNN</th>
<th>ITML</th>
<th>Hybrid</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORL</td>
<td>0.9750</td>
<td>0.9100</td>
<td><strong>0.9880</strong></td>
</tr>
<tr>
<td>Yale Face</td>
<td>0.9700</td>
<td>0.9030</td>
<td><strong>1.0000</strong></td>
</tr>
<tr>
<td>ATT Face</td>
<td>0.9880</td>
<td>0.9220</td>
<td><strong>0.9910</strong></td>
</tr>
</tbody>
</table>

**TABLE I**: Test set face recognition rates of large margin nearest neighbors, information theoretic metric learning and the hybrid method on common face datasets using Gabor features.

Table II shows the test set face recognition rates using the DFT features and the three metric learning algorithms. Compared with table I the recognition rates are significantly low. Which concludes that, selection of good features is very important and Gabor filter bank is a good method for feature extraction. The worst recognition rate we obtain using DFT features is 46.70% in Yale dataset for ITML algorithm, using the Gabor features however, for the same learning algorithm we achieve a recognition rate of 90.30% which is a huge difference. Since the DFT features are not as informative as Gabor features, and ITML performs poorly the Hybrid method cannot improve the recognition rate further, and performs nearly the same as LMNN.

**TABLE II**: Test set face recognition rates of large margin nearest neighbors, information theoretic metric learning and the hybrid method on common face datasets using DFT features.

For clustering performance comparison, that we achieve using Gabor features and DFT features we show figure 3 and 4 which shows the most similar face indexes(closest neighbors) for a given face for ORL face dataset. Since this data set has 10 different images for 40 people, each face should be similar to the 9 of the faces, i.e. face #1 should have nearest neighbors in the interval [2-10] along the vertical axis. This means we should see a line with a positive unit slope in both figures. We see that in figure 3 which is obtained with Gabor features, the 4th, and 5th nearest neighbors the faces are occasionally clustered to wrong intervals i.e. face numbers around 250 and 350 but the majority of the faces are clustered correctly.

![Fig. 3: Five closest face candidates to a given face index using Gabor features. Horizontal axis is the face index, and vertical axis is the closest face for a given face.](image-url)
Fig. 4: Five closest face candidates to a given face index using DFT features. Horizontal axis is the face index, and vertical axis is the closest face for a given face.

Fig. 5: ATT dataset queried for a face image on topleft, 8 of closest recognized faces are returned by the algorithm that uses Gabor features.

Fig. 6: Yale dataset queried for a face image on topleft, 8 closest recognized faces are returned by the algorithm that uses Gabor features.

Fig. 7: ORL dataset queried for a face image on topleft, 8 of closest recognized faces are returned by the algorithm that uses Gabor features.

Fig. 8: ATT dataset queried for a face image on topleft, 8 of closest recognized faces are returned by the algorithm that uses FFT features.
REFERENCES


