ABSTRACT

Title of dissertation: MULTIMODAL APPROACHES TO COMPUTER VISION PROBLEMS

Christopher Reale, Doctor of Philosophy, 2017

Dissertation directed by: Professor Rama Chellappa
Department of Electrical
and Computer Engineering

The goal of computer vision research is to automatically extract high-level information from images and videos. The vast majority of this research focuses specifically on visible light imagery. In this dissertation, we present approaches to computer vision problems that incorporate data obtained from alternative modalities including thermal infrared imagery, near-infrared imagery, and text. We consider approaches where other modalities are used in place of visible imagery as well as approaches that use other modalities to improve the performance of traditional computer vision algorithms. The bulk of this dissertation focuses on Heterogeneous Face Recognition (HFR). HFR is a variant of face recognition where the probe and gallery face images are obtained with different sensing modalities. We also present a method to incorporate text information into human activity recognition algorithms.

We first present a kernel task-driven coupled dictionary model to represent the data across multiple domains for thermal infrared HFR. We extend a linear coupled dictionary model to use the kernel method to process the signals in a high

dimensional space; this effectively enables the dictionaries to represent the data non-linearly in the original feature space. We further improve the model by making the dictionaries task-driven. This allows us to tune the dictionaries to perform well on the classification task at hand rather than the standard reconstruction task. We show that our algorithms outperform algorithms based on standard coupled dictionaries on three datasets for thermal infrared to visible face recognition.

Next, we present a deep learning-based approach to near-infrared (NIR) HFR. Most approaches to HFR involve modeling the relationship between corresponding images from the visible and sensing domains. Due to data constraints, this is typically done at the patch level and/or with shallow models to prevent overfitting. In this approach, rather than modeling local patches or using a simple model, we use a complex, deep model to learn the relationship between the entirety of cross-modal face images. We describe a deep convolutional neural network-based method that leverages a large visible image face dataset to prevent overfitting. We present experimental results on two benchmark data sets showing its effectiveness.

Third, we present a model order selection algorithm for deep neural networks. In recent years, deep learning has emerged as a dominant methodology in machine learning. While it has been shown to produce state-of-the-art results for a variety of applications, one aspect of deep networks that has not been extensively researched is how to determine the optimal network structure. This problem is generally solved by ad hoc methods. In this work we address a sub-problem of this task: determining the breadth (number of nodes) of each layer. We show how to use group-sparsity-inducing regularization to automatically select these hyper-parameters. We

demonstrate the proposed method by using it to reduce the size of networks while maintaining performance for our NIR HFR deep-learning algorithm. Additionally, we demonstrate the generality of our algorithm by applying it to image classification tasks.

Finally, we present a method to improve activity recognition algorithms through the use of multitask learning and information extracted from a large text corpora. Current state-of-the-art deep learning approaches are limited by the size and scope of the data set they use to train the networks. We present a multitask learning approach to expand the training data set. Specifically, we train the neural networks to recognize objects in addition to activities. This allows us to expand our training set with large, publicly available object recognition data sets and thus use deeper, state-of-the-art network architectures. Additionally, when learning about the target activities, the algorithms are limited to the information contained in the training set. It is virtually impossible to capture all variations of the target activities in a training set. In this work, we extract information about the target activities from a large text corpora. We incorporate this information into the training algorithm by using it to select relevant object recognition classes for the multitask learning approach. We present experimental results on a benchmark activity recognition data set showing the effectiveness of our approach.

MULTIMODAL APPROACHES TO COMPUTER VISION PROBLEMS

by

Christopher Reale

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2017

Advisory Committee:
Professor Rama Chellappa, Chair/Advisor
Professor Behtash Babadi
Professor Joseph JaJa
Professor Nasser M. Nasrabadi
Professor Larry Davis

Dedication

To my family.

Acknowledgments

First and foremost, I would like to thank my advisor, Professor Rama Chellappa, for his supervision and guidance. I would not have made it if not for his constant support and belief in me. For that I am truly thankful.

I would like to thank Professor Nasser M. Nasrabadi. Nasser mentored through at a crucial formative point during my PhD studies. From him I learned how to conduct research at both a high-level such as coming up with new solutions to problems and a low-level such as the nitty gritty details of writing and publishing papers.

I would like to thank the other members of the committee: Professor Behtash Babadi, Professor Joseph JaJa, and Professor Larry Davis. It was honor to have them on my committee and thank them for taking the time to serve on it.

I would like to acknowledge my coworkers at ARL. Specifically, I would like to thank Prudhvi, Heesung, Sungmin, Hyungtae, Sean, Ben, Nathan, and Matt for creating a positive environment to conduct research.

I would also like to thank other members of Professor Chellappa's group, including Garrett, Tho, Vishal, Dave, Priyanka, Swami, Heng, and Pouya.

Finally, I would like to thank my family for providing me with stability. Without them I would not have made it.

Table of Contents

List of Figures

List of Abbreviations

CD	Coupled Dictionaries
CNN	Convolutional Neural Network
HFR	Heterogeneous Face Recognition
IR	Infrared
KCD	Kernel Coupled Dictionaries
KTCD	Kernel Task-driven Coupled Dictionaries
NIR	Near-infrared
PLS	Partial Least Squares
TCD	Task-driven Coupled Dictionaries
TSN	Temporal Segment Networks

Chapter 1: Introduction and Contributions

1.1 Introduction

The main goal of computer vision research is to design algorithms to extract high-level information from images in videos. This can be achieved in many ways such as recognizing and localizing objects in images (object detection), identifying a person by an image of their face (face recognition), and identifying what a person in a video is doing (activity recognition). Modern approaches to these problems train models on large data sets of examples and then use them to extract information from new data samples. In most cases, the models are both trained and evaluated on visual imagery. This is due to the abundance of visual imagery sensors and their relevance to the problems being solved.

While visual imagery is rightfully at the core of the data used to solve computer vision problems, there are instances when other data modalities can supplement visual training data to improve the performance of computer vision algorithms. For example, classification of remote sensing images can be aided by additional (non-visible) frequency bands by using hyperspectral imaging [1]. In this case, the extra frequency bands provide more information about the scene captured by the imaging system. In some circumstances, non-visible sensors can capture more information

than visible sensors. A classic example of this is in scenes with low ambient lighting. Visible sensors rely on ambient light reflected off of objects in a scene in order to capture them. If there is not enough ambient light, visible cameras cannot capture meaningful information. On the other hand, infrared sensors do a much better job of capturing information in low-light scenarios. This is because infrared light is generally emitted from rather than reflected off of objects of interest. In scenarios such as these, with the right algorithms, high-level information can be extracted from infrared imagery whereas it cannot be from visible imagery.

In this dissertation, we focus on multimodal (i.e. using more than one data modality) approaches to two computer vision problems: heterogeneous face recognition (HFR) and activity recognition. HFR is a variant of face recognition where the probe and gallery face images are obtained with different sensing modalities. This is of practical importance in situations where the dominant sensing modality (visible imagery) is ineffective or unavailable and other modalities such as infrared and drawn sketeches can capture some information from the subjects face. In this dissertation, we present approaches to two specific examples of this problem: thermal infrared HFR and near-infrared HFR.

Activity recognition is the problem of recognizing human activities in videos. We frame the problem as a classification task. Thus, we have a list of target activities and, given a video, we must determine which of those activities it depicts. This can be applied to robotics (helping robots to understand their environment and thus react to it better), surveillance (sift through a large number of video streams to detect abnormal behavior), and video-tagging (automatically tag videos to make

them easier to find). We do not consider the problem of activity detection and localization, and thus assume that the majority of the video clip depicts the target activity. In this work we attempt to extract general knowledge from text resources and use it to guide activity recognition training algorithms.

1.2 Organization and Contributions

This dissertation contains four core chapters (Chapter 2, Chapter 3, Chapter 4, and Chapter 5). Each chapter contains an introduction to the specific problem it addresses, work related to the contents of the chapter, the method used to solve the problem, and experiments validating the approach. Here we briefly summarize the chapters and the key contributions put forward in each.

1.2.1 Kernel and Task-Driven Coupled Dictionaries for Thermal Infrared HFR

In Chapter 2, we propose a method for HFR in the thermal IR domain. Our goal is to enable the matching of thermal infrared probe images to visible light gallery images. This is useful in scenarios with little to no ambient illumination. Our baseline approach uses the coupled dictionaries of Yang et al. [2] to enable cross-modal matching. We make two improvements to the model by incorporating the kernel method and task-driven supervision. The main contributions are as follows,

- We incorporate the kernel method into a coupled dictionary model.
- We incorporate task-driven supervision into coupled dictionary learning.

- We apply these models to thermal IR heterogeneous face recognition.

1.2.2 Deep CNN Approach for Near-Infrared HFR

In Chapter 3, we propose a method for HFR in the near-infrared domain. Near-infrared sensors are useful in scenes with some ambient illumination, but not enough for traditional visible cameras. We attempt to solve this problem by using deep convolutional neural networks to map IR and visisble face images into a common domain where they can be directly compared. The main contributions are as follows,

- We present the first method to use a deep model to learn global features for HFR.
- We learn coupled deep convolutional neural networks to map visible and NIR faces into a domain-independent latent feature space where they can be compared directly.
- We demonstrate how to leverage a large visible face recognition dataset to prevent overfitting of the networks.

1.2.3 Neural Network Breadth Selection

In Chapter 4, we present a pruning algorithm to select the breadth of network layers in deep neural networks. Deep neural networks currently provide state-of-the-art results on virtually all machine learning problems. One unanswered question related to deep networks is how to choose network hyperparameters (e.g. depth and breadth). This chapter presents a method to automatically select the breadth

(i.e. number of neurons) of each layer in a network. The main contributions of this chapter are as follows,

- We present a method to automatically determine the breadth of neural network layers.
- We do this by applying group sparsity regularization.
- We show the generality of this method by applying it to near-infrared heterogeneous face recognition networks as well as networks trained for image classification tasks.

1.2.4 Text Guided Activity Recognition

In Chapter 5, we present a method to incorporate information from text resources into activity recognition algorithms. Most state-of-the-art approaches to activity recognition rely on deep convolutional neural networks. One of the main limitations of CNNs is their reliance on very large training datasets which are expensive to collect and annotate. In this work, we propose to leverage a large dataset collected for another problem (object recognition) through multitask learning to minimize the need for a large activity recognition dataset. We then leverage information extracted from text to help select the optimal objects to learn to recognize in the multitask framework. The main contributions of this chapter are as follows,

- We present a multi-task learning approach to allow activity recognition algorithms to use deeper network architectures without overfitting.

- We show how to use information from a large text corpora to guide the multi-task learning approach by ensuring the secondary task is relevant to the primary task.

- We achieve state-of-the-art performance on activity recognition benchmark data sets.

Chapter 2: Kernel and Task-Driven Coupled Dictionaries for Thermal Infrared HFR

2.1 Introduction

Over the past decade, there has been a significant amount of research in the signal processing community dedicated to sparse coding and dictionary learning. Sparse coding refers to representing a signal as a linear combination of a small subset of a large pool of components. It allows for a compact representation of a rich set of signals. Dictionary learning is the process of learning a large pool of components (dictionary atoms) that can accurately represent a set of signals in this manner.

In recent years, sparse coding has been extended to represent phenomena observed through multiple sensing modalities. Typically this is done by using a single set of sparse coefficients to represent the object under observation. Domain specific dictionaries then serve as a link between the object's sparse codes and its representations when viewed with the corresponding sensing modalities. Not only does this allow the typical compactness and richness afforded by sparse coding, it also creates a *domain independent* representation for the signals. Depending on

the method used to train the dictionaries, this can facilitate the comparison of phenomena across modalities through their sparse codes. The dictionary learning algorithm presented by Yang et al. [3] seeks to train Coupled Dictionaries that maximize this cross-modal ability for image superesolution.

In this work, we improve upon the coupled dictionaries presented by Yang et al. [3] and apply them to thermal IR heterogeneous face recognition (HFR). Thermal HFR is the problem of identifying a thermal IR face image using a gallery of visible light face images. This is a very difficult problem because visible light and thermal IR face images capture drastically different underlying phenomena; visible light cameras capture light *reflected* off of the face while thermal IR cameras capture light *emitted* from the face. Causes of variation in one domain may be completely undetectable in the other. For example, while it is well known that ambient illumination can have a drastic effect on visible light images, it does not affect thermal imagery at all. Conversely, thermal IR face images are very dependent on the temperature of the subject. A warmer face yields higher intensity values in IR images, while visible images are essentially invariant to temperature changes. Despite their vast differences, the two sensing methods do capture some shared information. For example, as shown in Figures 2.1 and 2.2, the overall shape of the face and some edge information are constant across domains. In this work, we learn a pair of coupled dictionaries to capture how this shared information manifests itself across domains. We can then harness this information to directly compare thermal probe images to visible gallery images. We do this by computing the sparse codes of thermal and visible face images, which, after training their corresponding

Figure 2.1: Sample images from the WSRI dataset. The visible images (top) correspond to the thermal images (bottom) directly below them.

dictionaries, are assumed to lie in a domain-independent latent feature space for our application. Figure 2.3 shows a visual example of the amount of information that can be transferred across domains.

We improve upon coupled dictionaries by applying two methods: kernelization and task-driven supervision. In the first improvement, we learn dictionaries whose atoms combine in a non-linear manner. While standard linear dictionaries have been shown to provide good results, depending on the data, the linearity assumption can be too restrictive. Kernelization allows for a richer model that can more effectively represent real data where this is the case. To do this, for all computations, we map

Figure 2.2: Sample images from the UND dataset. The visible images (top) correspond to the thermal images (bottom) directly below them.

the signals and dictionary atoms into a high-dimensional space using a non-linear function. We then train the dictionaries to encode signals in the high-dimensional space. While the dictionaries are linear in the new space, they are effectively non-linear in the original feature space. Depending on the dimensionality of the high-dimensional space, learning the dictionaries may not be computationally feasible using the original algorithm. We use Mercer kernels [4] to ensure the dictionary

learning optimization problem is tractable even when the dimension of the space is very large (possibly infinite).

The second improvement applies to scenarios where the dictionaries are used for some objective other than the standard reconstruction task. Historically, they have been used mainly for superresolution and image synthesis which are reconstructive in nature. In our case, the dictionaries will be used to extract features for face recognition (classification). Typically, this would be accomplished by first training the dictionaries to perform accurate reconstruction and subsequently training a classifier on the sparse codes obtained with the dictionaries. While this provides good results [5], the dictionaries are not optimized for classification. Our end goal is not to reconstruct the data, but rather to correctly identify faces. In this work, we extend coupled dictionaries to follow the task-driven approach of Mairal et al. [6] to optimize the dictionaries for our specific problem rather than reconstruction. This is done by including the classifier loss function in the coupled dictionary learning objective. Thus, when training the dictionaries, they are specifically altered to improve the classification performance. This necessitates concurrently training the dictionaries and classifier.

In summary, the main contributions of this chapter are as follows,

- We incorporate the kernel method to a coupled dictionary model.
- We incorporate task-driven supervision to coupled dictionary learning.
- We apply these models to thermal IR HFR.

The two proposed improvements (the kernel method and task-driven supervi-

sion) are orthogonal. Thus we can improve upon the original Coupled Dictionaries (CD) with Kernel Coupled Dictionaries (KCD), Task-Driven Coupled Dictionaries (TCD), and the combination of the two, Kernel Task-Driven Coupled Dictionaries (KTCD).

The remainder of the chapter is organized as follows. In Section 2.2, we review literature related to our problem formulation and application. In Section 2.3, we present the optimization formulation we use to learn the dictionaries. In Section 2.4, we describe the algorithms used to solve said optimization problems. In Section 2.5, we present experimental results validating the effectiveness of our approach. We conclude this chapter in Section 2.6.

2.2 Related Work

In this section, we give an overview of the most relevant work on the topics of coupled dictionaries, kernel dictionaries, task-driven dictionaries, visible face recognition, thermal IR face recognition, and heterogeneous face recognition.

2.2.1 Coupled Dictionaries

Coupled dictionaries were first introduced by Yang et al. [7] to solve the problem of super-resolution. They extended their approach by first adding one [3] and then a second [2] L1 minimization bi-level constraint to their optimization formulation. Since their initial publication, many others have extended their model and/or applied it to different problems [8,9]. Zheng et al. [9] applied coupled dictionaries to

cross-view action recognition. Jia et al. [10] loosened the restriction of having equal sparse codes across domains to merely have the same support across domains. They applied their model to a variety of problems. Wang et al. [8] allowed for a linear transformation between sparse codes across domains and applied their formulation to image super-resolution and photo-sketch synthesis. He et al. [11] utilized a Beta process prior to training coupled dictionaries. In this chapter we extend the work presented in [2], but our improvements could also be applied to most other coupled dictionary models as well.

2.2.2 Kernel Dictionaries

Many linear machine learning and data representation techniques can achieve better results by first non-linearly mapping the data into a higher dimensional feature space. This makes the linear techniques behave as non-linear techniques in the original feature space. For certain mapping functions, the kernel trick can then be used to ensure the tractability of the computations [4]. This applies to sparse coding and dictionary learning. Zhang et al. [12] use kernels for sparse coding for a variety of applications. Gao et al. [13, 14] apply kernels to sparse coding and dictionary learning for face recognition, image classification, and low-rank matrix approximation. Nguyen et al. [15] do the same with a slightly different model for digit recognition. Shrivastava et al. [16] extend kernel dictionaries further to learn the optimal kernel function by using a multiple kernel learning framework. The kernel model we use is most similar to the one used in [13].

2.2.3 Task-Driven Dictionary Learning

While much of the sparse coding and dictionary learning literature has focused on reconstructive dictionaries (i.e., dictionaries optimized to reconstruct signals as accurately as possible), more recently dictionaries have been explicitly trained to perform other tasks (e.g., classification, regression). Mairal et al. [6] proposed tuning a dictionary for supervised classification, semi-supervised classification, and regression tasks. They did this by jointly optimizing the dictionary and classification/regression model parameters. Jiang et al. [17] further build on this model by optimizing the dictionary to enforce label consistency for the atoms (i.e each atom is active for relatively few classes). Wang et al. [18] attempt to improve upon sparse representation classification (SRC) [19] by formulating the dictionary learning and classification as a margin maximization problem. Our task-driven model extends the algorithm in [6] to coupled dictionaries.

2.2.4 Visible Face Recognition

Visible face recognition has been extensively researched for several decades. Prior to 2014, shallow methods (methods that don't use deep-learning) were the prominent methodology, many of them focusing on sparse coding and dictionary learning. In a seminal paper, Wright et al. [19] use a lasso formulation to classify face images. Yang et al. [20] modify [19] to use gabor features in order to both improve performance and reduce computation time. Lu et al. [21] extend the work of [19] to take locality into account. Arandjelovic and Cipolla [22] formulate a

weak photometric model and a learn generic face invariant for unconstrained face recognition in videos. Lu et al. [23] use a manifold-based method for face recognition with only a single gallery sample per subject.

It's worth noting that in the past few years, most face recognition algorithms use deep CNNs trained on large datasets [24–28]. Although they are very effective, these methods require significant amounts of training data, which, due to the relatively small number of thermal IR cameras, is not available for our problem.

2.2.5 Thermal Face Recognition

Many papers have focused on thermal IR face recognition as well. Ma et al. [29] proposed a non-rigid face registration method using Gaussian fields criterion and apply it to infrared face registration. Hermosilla et al. [30] evaluate a variety of recognition methods under varying conditions for thermal face recognition. Ghiass et al. [31] use a dual-dimension active appearance model ensemble. Hermosilla et al. [32] combine thermal face imagery with vascular network information to improve results of a variety of recognition methods. A more thorough review can be found in surveys by Ghiass et al. [33, 34].

2.2.6 Heterogeneous Face Recognition

While face recognition has received a tremendous amount of attention in the visible domain [35] and the IR domain [33, 34], thermal to visible HFR has not been researched as much.

Bourlai et al. [36] evaluate the effectiveness of different image preprocessing and feature extraction methods. They preprocess the images using the Single-Scale Self Quotient Image and Difference of Gaussian Filtering. They then hand-design features (based on Local Binary Patterns) in an attempt to make them more robust to the sensing modality shift. They compute the similarities/differences of the features using several standard distance metrics (chi-squared, L2, L1). Choi et al. [37, 38] also bridge the domain gap with hand-crafted features. They preprocess the images with Difference of Gaussian Filtering and extract Histograms of Oriented Gradient as features. They then use Partial Least Squares (PLS) combined with nearest neighbor to classify the images. They increase the classification rate by using thermal images of non-gallery subjects as negative examples for all classes during the PLS training. These methods ultimately rely on hand-crafted features. The main advantage and disadvantage for these methods is the same: the same features are used regardless of sensor. This saves on computation time as new features don't need to be relearned for a new sensor, but produces worse results as the features aren't optimized specifically for the data.

Klare and Jain [39] utilize a variety of standard preprocessing (DoG Filtering, Center-Surround Divisive Normalization, Gaussian Filtering) and feature-extraction (SIFT, LBP) techniques. The novelty in their work lies in the assumption that a similarity measure within one domain corresponds to similarity within another domain. While this assumption may be reasonable for some domains (NIR, sketch), it is not a good assumption for thermal infrared. Due to the different underlying physical phenomena, a person's face can look wildly different in two images due to

changes in ambient lighting while maintaining a constant thermal signature. Similarly, a person's face can produce different thermal images while looking the same in visible images. With this assumption in mind, they use the kernel similarities of a sample to a fixed training set as the sample's features for matching.

It is worth noting that although thermal to visible face recognition has not specifically received much attention, the more general problem of HFR has. These methods mostly focus on recognizing faces from near infrared (NIR) images or drawn sketches using a visual gallery [8, 40–45]. While these papers propose many interesting algorithms, they are designed for domain shifts that are less drastic than the thermal to visible problem [46]. In Section 2.5 we compare our method to two of those methods [8, 45] on thermal IR datasets.

2.3 Coupled Dictionary Design

Here we formulate the coupled dictionary design problems we solve to learn the dictionaries.

2.3.1 Unsupervised Coupled Dictionary Design

For both the linear and the kernel versions, we learn the visible dictionary $\mathbf{D}_x \in \mathbb{R}^{d_x \times K}$ and the infrared dictionary $\mathbf{D}_y \in \mathbb{R}^{d_y \times K}$ by solving the following optimization problem,

$$
\begin{aligned}
&\min_{\mathbf{D}_x,\mathbf{D}_y} \sum_{i=1}^{N} R(\mathbf{x}_i,\mathbf{D}_x,\mathbf{z}_i^x) + R(\mathbf{y}_i,\mathbf{D}_y,\mathbf{z}_i^y) + \gamma_1 \|\mathbf{z}_i^x - \mathbf{z}_i^y\|_2^2 \\
&\text{s.t. } \mathbf{z}_i^x = \arg\min_{\mathbf{z}} R(\mathbf{x}_i,\mathbf{D}_x,\mathbf{z}) + \lambda\|\mathbf{z}\|_1, \forall i \\
&\qquad \mathbf{z}_i^y = \arg\min_{\mathbf{z}} R(\mathbf{y}_i,\mathbf{D}_y,\mathbf{z}) + \lambda\|\mathbf{z}\|_1, \forall i \\
&\qquad \|\mathbf{D}_x(:,k)\|_2 = 1, \forall k \in \{1,2,\ldots,K\} \\
&\qquad \|\mathbf{D}_y(:,k)\|_2 = 1, \forall k \in \{1,2,\ldots,K\},
\end{aligned}
\tag{2.1}
$$

where $\{(\mathbf{x}_i,\mathbf{y}_i)\}_{i=1}^N$ is a set of N pairs of features extracted from corresponding training samples from domains x and y, respectively. The features are of dimension d_x and d_y. The parameter λ is a tuning parameter used to adjust the trade-off between the representative ability and the sparsity of the sparse codes $\mathbf{z}_i^* \in \mathbb{R}^K$. Similarly, γ_1 can be used to trade-off between representative ability and continuity across domains. For both the linear and kernel versions, $R(\mathbf{x},\mathbf{D},\mathbf{z})$ is the squared 2-norm of the reconstruction error when representing the signal $\mathbf{x}$ with sparse codes $\mathbf{z}$ and dictionary $\mathbf{D}$. The two versions differ based on the space which is used to represent the signals and dictionary atoms. In the linear version, we represent the signals in the Euclidean space. Thus $R(\mathbf{x},\mathbf{D},\mathbf{z}) = \|\mathbf{x} - \mathbf{D}\mathbf{z}\|_2^2$. This corresponds to the model introduced in [2]. In the kernel version, we map the signals $\mathbf{x}_i$ and $\mathbf{y}_i$ (and the dictionary atoms of $\mathbf{D}_x$ and $\mathbf{D}_y$) into a higher dimensional feature space with a mapping function Φ. Thus $R(\mathbf{x},\mathbf{D},\mathbf{z}) = \|\Phi(\mathbf{x}) - \Phi(\mathbf{D})\mathbf{z}\|_2^2$. Although Φ can map the signals and dictionary atoms into a space of very high (possibly infinite) dimension, the optimization problem becomes tractable by applying the kernel trick [4]. Though any Mercer Kernel would suffice, for our experiments, we use the Gaussian radial

basis function (RBF) kernel.

2.3.2 Task-Driven Coupled Dictionary Design

We formulate the Task-Driven Coupled Dictionaries in a similar manner. In addition to the training samples $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$, we are also given the gallery samples $\{\hat{\mathbf{x}}_j\}_{j=1}^M$ from only one modality (e.g. visible) with corresponding labels l_j,

$$\begin{aligned}
\min_{\mathbf{D}_x, \mathbf{D}_y, \mathbf{W}} & \sum_{i=1}^{N} R(\mathbf{x}_i, \mathbf{D}_x, \mathbf{z}_i^x) + R(\mathbf{y}_i, \mathbf{D}_y, \mathbf{z}_i^y) + \gamma_1 \|\mathbf{z}_i^x - \mathbf{z}_i^y\|_2^2 \\
& + \gamma_2 \sum_{j=1}^{M} \hat{L}(\hat{\mathbf{z}}_j^x, l_j, \mathbf{w}) \qquad (2.2) \\
\text{s.t. } & \mathbf{z}_i^x = \arg\min_{\mathbf{z}} R(\mathbf{x}_i, \mathbf{D}_x, \mathbf{z}) + \lambda \|\mathbf{z}\|_1, \forall i \\
& \mathbf{z}_i^y = \arg\min_{\mathbf{z}} R(\mathbf{y}_i, \mathbf{D}_y, \mathbf{z}) + \lambda \|\mathbf{z}\|_1, \forall i \\
& \hat{\mathbf{z}}_j^x = \arg\min_{\mathbf{z}} R(\hat{\mathbf{x}}_j, \mathbf{D}_x, \mathbf{z}) + \lambda \|\mathbf{z}\|_1, \forall j \\
& \|\mathbf{D}_x(:,k)\|_2 = 1, \forall k \in \{1, 2, \ldots, K\}, \\
& \|\mathbf{D}_y(:,k)\|_2 = 1, \forall k \in \{1, 2, \ldots, K\},
\end{aligned}$$

where $\hat{L}(\hat{\mathbf{z}}_j^x, l_j, \mathbf{w})$ evaluates the loss function of a classifier with parameters $\mathbf{w}$ for sample j. So for example, for a binary linear support vector machine we would have $\mathbf{w} \in \mathbb{R}^K$ and $\hat{L}(\mathbf{z}, l, [\mathbf{w}\ b]) = \max(l\mathbf{w}^T\mathbf{z} + b, 0) + \|\mathbf{w}\|_2^2/M$. The parameters γ_1 and γ_2 are used to trade-off between representative error, cross-domain discrepancy, and classification ability. Note that, although we optimize the dictionaries and the classifier jointly, the gallery samples cannot be used to better learn the relationship between the domains because we do not have the corresponding thermal domain

test probes.

2.4 Optimization Algorithm

In this section we describe the algorithms used to solve the coupled dictionary design problems formulated in Section 2.3. We also describe all other implementation and engineering details.

2.4.1 Unsupervised Gradient Calculation

As in [2], we solve the optimization problems by stochastic gradient descent. For a batch of unlabeled samples $i \in S$, we update the dictionaries as follows

$$\mathbf{D}_x \Leftarrow \mathbf{D}_x - \nu_t (\nabla L)_{\mathbf{D}_x} / \|(\nabla L)_{\mathbf{D}_x}\|_F, \tag{2.3}$$

$$\mathbf{D}_y \Leftarrow \mathbf{D}_y - \nu_t (\nabla L)_{\mathbf{D}_y} / \|(\nabla L)_{\mathbf{D}_y}\|_F, \tag{2.4}$$

where ν_t is the iteration dependent learning rate. $(\nabla L)_{\mathbf{D}_x} = \frac{1}{|S|} \sum_{i \in S} (\nabla L_i)_{\mathbf{D}_x}$ and $(\nabla L)_{\mathbf{D}_y} = \frac{1}{|S|} \sum_{i \in S} (\nabla L_i)_{\mathbf{D}_y}$ are the average gradients of the loss function $L_i = \frac{1}{2} R(\mathbf{x}_i, \mathbf{D}_x, \mathbf{z}_i^x) + \frac{1}{2} R(\mathbf{y}_i, \mathbf{D}_y, \mathbf{z}_i^y) + \frac{\gamma_1}{2} \|\mathbf{z}_i^x - \mathbf{z}_i^y\|_2^2$ over the set S with respect to $\mathbf{D}_x$ and $\mathbf{D}_y$. We calculate the individual gradients with the chain rule as follows

$$(\nabla L_i)_{\mathbf{D}_x} = \frac{\partial L_i}{\partial \mathbf{D}_x} + \frac{\partial L_i}{\partial \mathbf{z}_i^x} \cdot \frac{\partial \mathbf{z}_i^x}{\partial \mathbf{D}_x}, \tag{2.5}$$

$$(\nabla L_i)_{\mathbf{D}_y} = \frac{\partial L_i}{\partial \mathbf{D}_y} + \frac{\partial L_i}{\partial \mathbf{z}_i^y} \cdot \frac{\partial \mathbf{z}_i^y}{\partial \mathbf{D}_y}. \tag{2.6}$$

The partial derivatives in 2.5 and 2.6 for the linear case can be calculated as follows

$$\frac{\partial L_i}{\partial \mathbf{D}_x} = (\mathbf{D}_x \mathbf{z}_i^x - \mathbf{x}_i)\mathbf{z}_i^{xT}, \quad \frac{\partial L_i}{\partial \mathbf{D}_y} = (\mathbf{D}_y \mathbf{z}_i^y - \mathbf{y}_i)\mathbf{z}_i^{yT}, \tag{2.7}$$

$$\frac{\partial L_i}{\partial \mathbf{z}_i^x} = \mathbf{D}_x{}^T(\mathbf{D}_x \mathbf{z}_i^x - \mathbf{x}_i) + \gamma(\mathbf{z}_i^x - \mathbf{z}_i^y), \tag{2.8}$$

$$\frac{\partial L_i}{\partial \mathbf{z}_i^y} = \mathbf{D}_y{}^T(\mathbf{D}_y \mathbf{z}_i^y - \mathbf{y}_i) - \gamma(\mathbf{z}_i^x - \mathbf{z}_i^y), \tag{2.9}$$

$$\frac{\partial \tilde{\mathbf{z}}_i^x}{\partial \tilde{d}_{rc}^x} = \left(\tilde{\mathbf{D}}_x^T \tilde{\mathbf{D}}_x\right)^{-1}\left(- \frac{\partial \tilde{\mathbf{D}}_x^T \tilde{\mathbf{D}}_x}{\partial \tilde{d}_{rc}^x}\tilde{\mathbf{z}}_i^x + \frac{\partial \tilde{\mathbf{D}}_x^T}{\partial \tilde{d}_{rc}^x}\mathbf{x}_i\right)., \tag{2.10}$$

$$\frac{\partial \tilde{\mathbf{z}}_i^y}{\partial \tilde{d}_{rc}^y} = \left(\tilde{\mathbf{D}}_y^T \tilde{\mathbf{D}}_y\right)^{-1}\left(- \frac{\partial \tilde{\mathbf{D}}_y^T \tilde{\mathbf{D}}_y}{\partial \tilde{d}_{rc}^y}\tilde{\mathbf{z}}_i^y + \frac{\partial \tilde{\mathbf{D}}_y^T}{\partial \tilde{d}_{rc}^y}\mathbf{y}_i\right). \tag{2.11}$$

The kernel version is a bit more involved, but still straight forward. Let us rewrite $\mathbf{D}_x$ in vector row and vector column formats as follows,

$$\begin{aligned}\mathbf{D}_x &= \begin{bmatrix} d_{11}^x & d_{12}^x & \cdots & d_{1K}^x \\ d_{21}^x & d_{22}^x & \cdots & d_{2K}^x \\ \vdots & \vdots & \ddots & \vdots \\ d_{n1}^x & d_{n2}^x & \cdots & d_{nK}^x \end{bmatrix} = \begin{bmatrix} (\mathbf{e}_1^x)^T \\ (\mathbf{e}_2^x)^T \\ \vdots \\ (\mathbf{e}_n^x)^T \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{d}_1^x & \mathbf{d}_2^x & \cdots & \mathbf{d}_K^x \end{bmatrix},\end{aligned} \tag{2.12}$$

and let

$$\mathbf{A}_x = \mathbf{K}(\mathbf{D}_x, \mathbf{D}_x) = \begin{bmatrix} \mathbf{a}_1^x & \mathbf{a}_2^x & \cdots & \mathbf{a}_K^x \end{bmatrix}. \tag{2.13}$$

Lastly, define $\mathbf{g}_x^{(ij)}$ as

$$\mathbf{g}_x^{(rc)} = -\frac{1}{\sigma^2}(\mathbf{e}_r^x - d_{rc}^x). * \mathbf{a}_c^x, \tag{2.14}$$

where .∗ signifies the element-wise product operator. Also define the corresponding variables for the y domain. The gradients in (2.5) and (2.6) for the kernel version can then be computed as follows

$$\frac{\partial L_i}{\partial d_{rc}^x} = z_{ic}^x \mathbf{z}_i^{xT} \mathbf{g}_x^{(rc)} + \frac{1}{\sigma^2}(d_{rc}^x - x_{ir}) k(\mathbf{x}, \mathbf{d}_c^x) z_{ic}^x, \tag{2.15}$$

$$\frac{\partial L_i}{\partial d_{rc}^y} = z_{ic}^y \mathbf{z}_i^{yT} \mathbf{g}_y^{(rc)} + \frac{1}{\sigma^2}(d_{rc}^y - y_{ir}) k(\mathbf{y}, \mathbf{d}_c^y) z_{ic}^y, \tag{2.16}$$

$$\frac{\partial L_i}{\partial \mathbf{z}_i^y} = \mathbf{A}_x \mathbf{z}_i^x - \mathbf{k}(\mathbf{x}_i, \mathbf{D}_x) + \gamma_1(\mathbf{z}_i^x - \mathbf{z}_i^y), \tag{2.17}$$

$$\frac{\partial L_i}{\partial \mathbf{z}_i^y} = \mathbf{A}_y \mathbf{z}_i^y - \mathbf{k}(\mathbf{y}_i, \mathbf{D}_y) - \gamma_1(\mathbf{z}_i^x - \mathbf{z}_i^y) \tag{2.18}$$

$$\frac{\partial \tilde{\mathbf{z}}_i^x}{\partial \tilde{d}_{rc}^x} = \tilde{\mathbf{A}}_x^{-1}\Big(- \frac{\partial \tilde{\mathbf{A}}_x}{\partial \tilde{d}_{rc}^x} \tilde{\mathbf{z}}_i^x + \frac{\partial \mathbf{k}(\mathbf{x}_i, \tilde{\mathbf{D}}_x)}{\partial \tilde{d}_{rc}^x} \Big). \tag{2.19}$$

$$\frac{\partial \tilde{\mathbf{z}}_i^y}{\partial \tilde{d}_{rc}^y} = \tilde{\mathbf{A}}_y^{-1}\Big(- \frac{\partial \tilde{\mathbf{A}}_y}{\partial \tilde{d}_{rc}^y} \tilde{\mathbf{z}}_i^y + \frac{\partial \mathbf{k}(\mathbf{y}_i, \tilde{\mathbf{D}}_y)}{\partial \tilde{d}_{rc}^y} \Big), \tag{2.20}$$

where, with slight abuse of notation, z_{ic}^x and z_{ic}^y are the c$^{\text{th}}$ elements of $\mathbf{z}_i^x$ and $\mathbf{z}_i^y$, respectively, and x_{ir} and y_{ir} are the r$^{\text{th}}$ elements of $\mathbf{x}_i$ and $\mathbf{y}_i$, respectively.

These equations determine the gradients of the non-zero sparse coefficients with respect to the active dictionary atoms. As in previous similar methods [3, 6] the remaining gradient values are set to zero.

2.4.2 Task-Driven Gradient Calculation

As in the unsupervised case, we perform a stochastic gradient descent to solve the dictionary learning optimization. Samples chosen from the training set have no effect on the supervised loss function; therefore the gradients for these samples can still be calculated by (2.5) and (2.6).

Samples chosen from the gallery only affect the supervised loss function $\hat{L}$. We calculate the gradient with

$$(\nabla \hat{L}_j)_{\mathbf{D}_x} = \frac{\partial \hat{L}_j}{\partial \mathbf{D}_x} + \frac{\partial \hat{L}_j}{\partial \mathbf{z}_j^x} \cdot \frac{\partial \mathbf{z}_j^x}{\partial \mathbf{D}_x}. \tag{2.21}$$

The dictionary $\mathbf{D}_x$ has no direct influence on $\hat{L}_j$ (it only affects it indirectly through $\hat{\mathbf{z}}_j^x$). Thus the first term $\frac{\partial \hat{L}_j}{\partial \mathbf{D}_x} = 0$. The first factor of the second term is the gradient of the loss function of a classifier with respect to a data point. This can be calculated for most standard classification algorithms (e.g. SVM, logistic regression, etc.). The second factor of the second term is the same as in the unsupervised setting and is calculated by (2.10) and (2.19) for the linear and kernel cases, respectively.

Samples taken from the gallery exist exclusively in the x domain. The y domain dictionary $\mathbf{D}_y$ does not impact the sparse representation for any of theses samples. Therefore, we have

$$(\nabla \hat{L}_j)_{\mathbf{D}_y} = \mathbf{0}. \tag{2.22}$$

In the unsupervised algorithm, we first learn the dictionaries and then train a classifier to be optimal for those dictionaries. In the task-driven case, we cannot learn the optimal dictionaries without the optimal classifier and vice versa. Therefore we train the coupled dictionaries and the classifier at the same time. Thus, in addition to updating the dictionaries, we must also update the classifier parameters $\mathbf{W}$. We do this according to

$$\mathbf{W} \Leftarrow \mathbf{W} - \nu_t(\nabla \hat{L})_{\mathbf{W}} / \|(\nabla \hat{L})_{\mathbf{W}}\|_F, \tag{2.23}$$

where $(\nabla \hat{L})_{\mathbf{W}} = \frac{1}{|\hat{S}|} \sum_{j \in \hat{S}} (\nabla \hat{L}_j)_{\mathbf{W}}$ is the average gradient of the loss function over the set $\hat{S}$ with respect to the classifier parameters. In this chapter, we use a linear one-versus-all support vector machine as a classifier, so the parameters $\mathbf{W}$ are the coefficients describing the separating hyperplanes for each of the gallery subjects.

2.4.3 Implementation Details

2.4.3.1 Preprocessing and Feature Extraction

We preprocess and extract features from face images with a four stage process initially described in [38]. First we align and crop the images based on four manually labeled fiducial points (the center of each eye, tip of the nose, and center of the mouth). Next, we filter the images with a Difference of Gaussian (DOG) filtering process defined as follows

$$I_{DOG}(x, y|\sigma_0, \sigma_1) = [N(x, y; \sigma_0) - N(x, y; \sigma_1)] * I(x, y), \tag{2.24}$$

where I_{DOG} is the filtered image, I is the input image, $N(x, y, \sigma)$ is a zero-correlation 2D Gaussian kernel with zero mean and standard deviation σ. Third, we normalize the contrast of each image according to the following equation,

$$I_C(x, y) = \tau \tanh\left(\frac{I_{DOG}(x, y)}{\tau}\right), \tag{2.25}$$

where I_C is the contrast normalized image, I_{DOG} is the input filtered image, and τ is a constant (set to 15 in this work). Finally, we extract Histogram of Oriented Gradient (HOG) features [47] from the contrast normalized images. Figures 2.4 and 2.5 show sample images after the first three steps.

2.4.3.2 Patch Based Dictionaries

Ideally we would like to learn dictionaries that can represent entire face images with a single set of sparse coefficients (i.e., the input samples to the dictionary learning algorithm would be face images). Many sparse coding face recognition algorithms take this approach and achieve good results [12, 19, 48]. This works well for the standard face recognition problem because the dictionary is learned on the same subjects that it will be tested on. The dictionary only needs to represent those specific subjects well. In our scenario, the set of training subjects and the set of gallery subjects are disjoint, therefore our dictionaries must generalize to *arbitrary* subjects. To achieve this we would need to provide the dictionary learning algorithm with significantly more training subjects than in previous work. Unfortunately, the large amount of data required prohibits us from taking this approach, and we cannot train coupled dictionaries for the entire face.

Instead, we opt to train patch-wise coupled dictionaries. Rather than having a set of sparse coefficients represent a corresponding pair of face images, we have them represent corresponding patches within those images. Intuitively, we are acknowledging that, for any reasonable amount of training data, faces are too unique

to an individual to be able to train dictionaries that generalize to arbitrary subjects. We use patches instead because they are more general; two very different faces may share similar parts (e.g. eyes, eyebrows, nose, lips, etc). Thus, patch-wise dictionaries learned on one set of subjects could then be usefully applied to another set of subjects.

More concretely, training the dictionaries on patches helps us to provide sufficient data for the training algorithm in two ways. The first is that there are many patches per image; the number of samples in our training set is multiplied by the number of patches per image. The second is that patches have smaller dimension and consequently exponentially fewer possible variations. This allows the coupled dictionaries to consist of fewer atoms with much smaller dimensionality. Since the dictionaries have significantly fewer parameters, fewer data samples are required to obtain good generalization. We further reduce the dimension of the features extracted from patches using the principle component analysis (PCA).

Although we learn the coupled dictionaries from image patches, we still want to train a classifier on the entirety of the face images. Thus, we concatenate the sparse codes of all the patches within an image to form the features used for training the classifier.

2.4.3.3 Optimization Details

We solve the optimization problems using mini-batch stochastic gradient descent. For the unsupervised case, we initialize the dictionaries randomly and nor-

malize their atoms. For each iteration, we estimate the gradients of the objective function using (2.5) and (2.6) with ten randomly chosen pairs of corresponding images from the training set. We then update the dictionaries by taking a step in opposite direction of the gradient with a iteration-dependent step-size. Finally, we renormalize the atoms of both dictionaries. We continue this process until we converge on a solution. The pseudo-code for this algorithm is shown in Algorithm 1.

Algorithm 1 Unsupervised Dictionary Learning

Randomly initialize $\mathbf{D}_x$ and $\mathbf{D}_y$

$t \Leftarrow 1$

repeat

Randomly choose $S \subset \{1, 2, \ldots, N\}$

for $i \in S$ **do**

$(\nabla L_i)_{\mathbf{D}_x} \Leftarrow \frac{\partial L_i}{\partial \mathbf{D}_x} + \frac{\partial L_i}{\partial \mathbf{z}_i^x} \cdot \frac{\partial \mathbf{z}_i^x}{\partial \mathbf{D}_x}$

$(\nabla L_i)_{\mathbf{D}_y} \Leftarrow \frac{\partial L_i}{\partial \mathbf{D}_y} + \frac{\partial L_i}{\partial \mathbf{z}_i^y} \cdot \frac{\partial \mathbf{z}_i^y}{\partial \mathbf{D}_y}$

end for

$(\nabla L)_{\mathbf{D}_x} \Leftarrow \frac{1}{|S|} \sum_{i \in S} (\nabla L_i)_{\mathbf{D}_x}$

$(\nabla L)_{\mathbf{D}_y} \Leftarrow \frac{1}{|S|} \sum_{i \in S} (\nabla L_i)_{\mathbf{D}_y}$

$\mathbf{D}_x \Leftarrow \mathbf{D}_x - \nu_t (\nabla L)_{\mathbf{D}_x} / \|(\nabla L)_{\mathbf{D}_x}\|_F$

$\mathbf{D}_y \Leftarrow \mathbf{D}_y - \nu_t (\nabla L)_{\mathbf{D}_y} / \|(\nabla L)_{\mathbf{D}_y}\|_F$

Normalize atoms of $\mathbf{D}_x$ and $\mathbf{D}_y$

$t \Leftarrow t + 1$

until convergence

The task-driven algorithm, shown in Algorithm 2, is slightly different. As in the unsupervised case, we solve the optimization by batch stochastic gradient descent. Rather than initializing randomly, we initialize with the solution to the unsupervised problem. We then initialize the classifier to be optimal for the unsupervised dictionaries. When selecting the random batch, we choose half of the samples from the training set and half of the samples from the gallery set. We estimate the gradients using (2.5) and (2.6) for the training samples and (2.21) and (2.22) for the gallery samples. In addition to updating the dictionaries, we also update the classifier parameters in each iteration based on the gallery samples in the batch.

We found that running the optimization for 1000 iterations was sufficient for convergence. Following [6], for each iteration t, we set the learning rate to $\nu_t = \rho \min(1, t_o/t)$. This gives us a constant step size of ρ for the first t_0 iterations followed by a slow decrease over time. In our experiments, we use a one-versus-all linear Support Vector Machine (SVM) as a classifier. We found it beneficial to retrain the SVM periodically (every 200 iterations) during the gradient descent. The efficiency of SVM training algorithms [49] enables us to do this without significantly slowing down the optimization.

2.5 Experiments

For all experiments, we partition the dataset into a training set to learn the dictionaries and a testing set (gallery and probe images) to test the classification

Algorithm 2 Task-Driven (Supervised) Dictionary Learning

Initialize $\mathbf{D}_x$ and $\mathbf{D}_y$ with unsupervised dictionaries

$t \Leftarrow 1$

repeat

Randomly choose $S \subset \{1, 2, \ldots, N\}$

for $i \in S$ **do**

$(\nabla L_i)_{\mathbf{D}_x} \Leftarrow \frac{\partial L_i}{\partial \mathbf{D}_x} + \frac{\partial L_i}{\partial \mathbf{z}_i^x} \cdot \frac{\partial \mathbf{z}_i^x}{\partial \mathbf{D}_x}$

$(\nabla L_i)_{\mathbf{D}_y} \Leftarrow \frac{\partial L_i}{\partial \mathbf{D}_y} + \frac{\partial L_i}{\partial \mathbf{z}_i^y} \cdot \frac{\partial \mathbf{z}_i^y}{\partial \mathbf{D}_y}$

end for

Randomly choose $\hat{S} \subset \{1, 2, \ldots, M\}$

for $j \in \hat{S}$ **do**

$(\nabla \hat{L}_j)_{\mathbf{D}_x} = \frac{\partial \hat{L}_j}{\partial \mathbf{D}_x} + \frac{\partial \hat{L}_j}{\partial \mathbf{z}_j^x} \cdot \frac{\partial \mathbf{z}_j^x}{\partial \mathbf{D}_x}$

Calculate $(\nabla \hat{L}_j)_{\mathbf{W}}$

end for

$(\nabla L)_{\mathbf{D}_x} \Leftarrow \frac{1}{|S|} \sum_{i \in S} (\nabla L_i)_{\mathbf{D}_x} + \frac{\gamma_2}{|\hat{S}|} \sum_{j \in \hat{S}} (\nabla \hat{L}_j)_{\mathbf{D}_x}$

$(\nabla L)_{\mathbf{D}_y} \Leftarrow \frac{1}{|S|} \sum_{i \in S} (\nabla L_i)_{\mathbf{D}_y}$

$(\nabla \hat{L})_{\mathbf{W}} \Leftarrow \frac{1}{|\hat{S}|} \sum_{j \in \hat{S}} (\nabla \hat{L}_j)_{\mathbf{W}}$

$\mathbf{D}_x \Leftarrow \mathbf{D}_x - \nu_t (\nabla L)_{\mathbf{D}_x} / \|(\nabla L)_{\mathbf{D}_x}\|_F$

$\mathbf{D}_y \Leftarrow \mathbf{D}_y - \nu_t (\nabla L)_{\mathbf{D}_y} / \|(\nabla L)_{\mathbf{D}_y}\|_F$

$\mathbf{W} \Leftarrow \mathbf{W} - \nu_t (\nabla \hat{L})_{\mathbf{W}} / \|(\nabla \hat{L})_{\mathbf{W}}\|_F$

Normalize atoms of $\mathbf{D}_x$ and $\mathbf{D}_y$

$t \Leftarrow t + 1$

until convergence

	WSRI	UND	NV-MW	NV-LW
PLS [38]	83.7	41.0	82.4	70.4
Deep CNN [45]	42.5	26.2	-	-
SCDL [8]	94.6	**52.1**	87.8	88.3
CD	93.1	50.0	89.5	89.4
TCD	93.1	50.5	90.3	89.1
KCD	95.6	51.4	92.4	93.0
KTCD	**95.9**	**52.0**	**93.2**	**93.3**

Table 2.1: Comparison of algorithm performance by dataset.

performance. In order to mimic real-world scenarios (one likely will not have access to a thermal gallery), none of the subjects in the gallery and probe images are present in the training set. This forces us to generalize the dictionaries to arbitrary subjects rather than overfitting to the ones they are trained on. We evaluate the effectiveness of our approach on three datasets: WSRI Dataset, UND Collection X1 [50, 51], and NVESD Dataset [52].

2.5.1 Wright State

The WSRI Dataset contains corresponding visible light and mid-wave infrared (MWIR) images. It was acquired at the Wright State Research Institute (WSRI) at Wright State University using a Basler A202k visible camera and a FLIR Systems SC6700 MWIR camera. It was initially collected to study how thermal face images (and the underlying physical phenomena that they capture) change after subjects

Dataset	WSRI	UND	NV-MW	NV-LW
Dictionary Size	30	20	20	20
λ	.1	.1	.1	.1
γ_1	8	2	8	8
γ_2	.65	.65	.65	.65
σ	1.5	1.5	.9	.9

Table 2.2: Parameter values used to generate results in Table 1.

perform different tasks (e.g. exercise, counting backwards, time-lapse etc.). We have repurposed a subset of the dataset to be used for Thermal to Visible face recognition.

The portion we use contains 64 subjects. For each subject we have between 14 and 33 (25 on average) pairs of face images (one visible and one IR) taken at the same time. The visible images are 1004×1004 pixels and the IR images are 512×640 pixels. Sample images are shown in Figure 2.1. After preprocessing, aligning, and cropping the faces based on manually labeled fiducial points, the final size of the images (shown in Figure 2.4) from both modalities is 390×470 pixels. We use a grid of 143 overlapping 64×64 pixel patches to extract features. For all experiments on this dataset we randomly choose ten subjects to serve as the dictionary training set and use the remaining 54 subjects as the gallery and test probes. We repeat this process ten times and report the average results over all experiments.

2.5.2 Notre Dame

The University of Notre Dame (UND) Collection X1 [50, 51] contians corresponding visible light and long-wave infrared (LWIR) images. It was collected using a Merlin uncooled LWIR camera and a high resolution visible color camera. Initially the visible and IR images are 1200×1600 pixels and 320×240 pixels, respectively (shown in Figure 2.2). After aligning and cropping, the final images (examples shown in Figure 2.5) are 110×150 pixels. We use a grid of 88 overlapping 24×24 pixel patches to extract features. The lower resolution and relatively poor quality of the LWIR images makes this data set significantly more difficult (and arguably closer to a real-world scenario) than the WSRI dataset. This is especially apparent when comparing the preprocessed images (see Figure 2.5) to those from the WSRI dataset (see Figure 2.4). The WSRI dataset shows significantly more detail, especially around the eyes, nose, and mouth.

The dataset comes split into a training set and a testing set. The training set consists of 159 subjects with one pair of images per subject. The testing set consists of 82 subjects with between 4 and 40 (average 28) image pairs per subject. We use the visible images of the testing set as our gallery and the LWIR images of the testing set as our probe images. We repeat all experiments ten times and report the average results. Although we use the same training and testing sets for each trial, results vary due to the combination of the random dictionary initialization and non-convexity of the optimization problems.

2.5.3 Night Vision

The Night Vision (NV) dataset [52] is a portion of the data collected in a joint effort between the Night Vision Electronic Sensors Directorate (NVESD) of the U.S. Army Communications-Electronics Research, Development and Engineering Center (CERDEC) and the U.S. Army Research Laboratory. The portion we used contains 50 captured with visible, mid-wave infrared, and long-wave infrared cameras. Half of the subjects have two images from each sensor each while the other half have four images from each sensor for a total of 150 image pairs. After alignment and cropping, the images are 174×174 pixels. We use a grid of 169 overlapping 24×24 pixel patches to extract features. We randomly partition the dataset into a ten-subject training set to train the dictionaries and a 40-subject gallery set to test their performance. As with our other experiments, we perform the experiment ten times and report the average results.

2.5.4 Discussion

The results of our methods and the parameters used to achieve them are presented in Tables 2.1 and 2.2 respectively. In addition to comparing the four variants described in this chapter (CD, TCD, KCD, and KTCD), we also compare to three other methods. The first method we compare to is the PLS-based approach [38]. The PLS-based approach suffers from not explicitly learning the relationship between domains. Although they attempt to learn it implicitly by training with negative thermal samples along with visible ones, the model simply cannot learn as

much information about the domain shift as dictionary-based methods can. Thus the PLS method performs worse than the dictionary learning methods.

The second method we compare to is a deep convolutional neural network approach [45]. Although it produces state-of-the-art results for near infrared face images, it performs very poorly on the thermal infrared datasets we use in this chapter. This is because the method assumes a certain level of similarity between the two domains. While this is fine for near IR imagery, thermal IR imagery is too different than visible light imagery, and causes the method to fail. We could not evaluate this algorithm on the Night Vision datasets because they have too few images to train deep convolutional neural networks without overfitting.

The third method, semi-coupled dictionary learning (SCDL) [8], is a variant of dictionary learning that allows for a linear transformation between the sparse coefficients of the two domains rather than fixing them together. As one would expect, this improves upon standard coupled dictionaries (CD) on the larger datasets (WSRI and UND) due to the extra degrees of freedom allowed by the linear transformation. On the other hand, the extra degrees of freedom allows the model to overfit on the smaller datasets (NV-MW, NV-LW), and it ends up performing slightly worse than standard coupled dictionaries. The method presented in this chapter (KTCD) provides the benefits of both algorithms. By using kernels, it allows for a more complex representation of the data. At the same time, due to the kernel trick, it has the same number of degrees of freedom as linear dictionaries, which prevents them from overfitting. Additionally, learning task-driven dictionaries incorporates extra discriminative information specific to the gallery. The combination of the two

allows KTCD to perform the best across all datasets.

2.5.5 Parameters

As in most machine learning algorithms, our method requires the selection of several parameters. In this section, we examine the effect that varying their values has on the face recognition performance.

2.5.5.1 Dictionary Size

In Figure 2.6 we examine the performance variation for different dictionary sizes. Ultimately, our algorithms are not very sensitive to this parameter. Even with dictionaries as small as five atoms, the results are virtually unchanged. This is due to the patch-wise nature of our approach. Even though each patch is encoded by a few sparse coefficients, the concatenation of the codes of all patches provides a rich representation for the face images. This robustness allows us to not spend time identifying the optimal dictionary size when applying our algorithms to new datasets.

2.5.5.2 Parameter γ_1

Figure 2.7 shows the effects of varying γ_1 on the WSRI and UND datasets, respectively. These plots show the importance of the coupling effect of the dictionary learning algorithm. As γ_1 tends toward zero, the coupling weakens, and the identification rate drops.

2.5.5.3 Training Subjects

As shown in Figure 2.8, the dictionary learning algorithms need very little data to reach the saturation point of their performance. On the WSRI dataset (Figure 2.8 (a)), the performance reaches that point with only six subjects. This is because each subject provides about 25 image pairs on average, and each image pair provides 143 patch pairs. Thus six subjects provide more than 20,000 training samples. On the other hand, the UND dataset (Figure 2.8 (b)) requires 70 subjects worth of data. This is because there is only one image pair for each training subject.

2.5.5.4 Parameter σ

Of all the parameters, the performance is most sensitive to changes in the kernel width σ. As shown in Figure 2.9 small changes to σ can cause a significant drop in performance. In general, finding an optimal kernel and its parameter(s) for a particular dataset is a problem unto itself [53]. We leave the issue of optimal kernel learning for coupled dictionaries to future work.

2.5.5.5 Parameter λ

As shown in Figure 2.10, varying λ over commonly used values has a moderate effect on the performance. Greater λ values (and sparser codes) tend to decrease the performance. This is because the data is simply to complex to be represented by codes that are that sparse. While denser codes produce better results, they also slow down the learning algorithm by increasing the dimension of the matrix $\tilde{\mathbf{D}}_x^T \tilde{\mathbf{D}}_x$

which needs to be inverted. It is worth noting that, while all of the algorithms perform worse as λ increases, the kernel methods are more robust to this variation than their linear counterparts.

2.5.5.6 Parameter γ_2

The parameter γ_2 controls the trade-off between the unsupervised objective function and the classifier objective function. As shown in Figure 2.11, changing it has little effect on the overall performance. This is because the task-driven improvement only modestly increases the overall performance. There are two reasons for this. The first reason is the use of patches rather than entire face samples. As described in subsection 2.4.3.2, we are designing general dictionaries that can be applied to all patches for arbitrary subjects. Unfortunately, features that are discriminative for a patch at one location on one subject's face may not be discriminitive at another location on another subject's face. It's not always possible to learn features for patches that are both general and discriminative because they may not exist. Essentially our need to have general dictionaries handcuffs the task-driven algorithm and dampens the improvement it provides.

The second reason is that we must balance between two objectives: classification and consistency across domains. Consider the work of Mairal et al. [6]. They optimize dictionaries for the task at hand. This works for the single domain case because the dictionary has only one objective. Standard dictionaries have one objective (reconstruction) and Mairal's dictionaries have one objective (task at hand).

Alternatively, coupled dictionaries are trained to optimize for two objectives: reconstruction error *and* cross-modal discrepancy. While we would like to focus our dictionaries on classification, we still must ensure the dictionaries enforce consistency of sparse codes across domains. If we sacrifice too much of the cross-domain consistency, the chasm between the two sensing modalities will cripple our classification performance regardless of how low we make its objective function.

2.5.6 Alignment Sensitivity

Face alignment is a key component of virtually all face recognition algorithm. In this chapter, we rely on four manually labeled landmarks (two eyes, nose, and mouth) to align our images. In real-world scenarios, manually labeling each image may not be feasible and a landmark detection algorithm must be used. In order to simulate this setting, we run experiments on images that have been aligned based on incorrect landmark points. Specifically, we add zero mean Gaussian noise to each coordinate of each manually labeled landmark, realign the images based on those, and then reevaluate the performance. Tables 2.3 and 2.4 show the performance degradation for various levels of noise. Note that the UND images have a much lower resolution than the WSRI images, so a single pixel (and thus a single pixel error) represents a larger portion of the face. Thus UND images are more sensitive to misalignment than the WSRI images. Additionally, when only using four landmarks for alignment, an error in one coordinate of one point can cause significant performance problems. In a real-world scenario, a larger set of landmarks could be

Noise std. (pixels)	CD	TCD	KCD	KTCD
0	93.1	93.1	95.6	95.9
2	92.2	92.4	94.1	94.3
4	88.6	88.7	90.2	90.5
6	82.1	82.3	84.7	84.8

Table 2.3: Alignment sensitivity on the WSRI dataset.

Noise std. (pixels)	CD	TCD	KCD	KTCD
0	50.0	50.5	51.4	52.0
1	43.2	43.6	44.9	45.1
2	31.0	31.0	33.1	33.8
3	23.3	24.1	25.2	25.6

Table 2.4: Alignment sensitivity on the UND dataset.

used to mitigate the effects of a few erroneous values.

2.5.7 Path Forward

Although progress has been made towards practical thermal face recognition algorithms, there are still many steps to be taken to get there. One of the most glaring weaknesses in current thermal face recognition research is the reliance on substandard datasets. Compared to other domains (visible light, near IR), thermal IR datasets are very small in both number of subjects and total images. Not only does this more closely mimic real world scenarios, it also enables the use of powerful algorithms that require large amounts of training data such as convolutional neural

networks. Additionally, current datasets are obtained in controlled settings with limited pose variation. Only so much value can be obtained from datasets that don't accurately represent real-world scenarios. Larger, more challenging datasets is the best path forward towards the application of thermal face recognition systems.

2.6 Conclusion

In this chapter, we have presented two improvements to coupled dictionaries. In the first, we map the data into a high-dimensional feature space before learning the dictionaries. This enables us to learn dictionary atoms that combine non-linearly. We then use the kernel trick to make the dictionary learning tractable. In the second improvement, we fine tune the dictionaries for classification rather than reconstruction. We do this by concurrently training the dictionaries and the classifier.

We have applied our dictionary learning algorithms to three thermal to visible face recognition datasets: WSRI, UND-X1, and NV. We present results that measure the improvement that our algorithms provide over the original coupled dictionaries, semi-coupled dictionaries, a deep neural network approach and a PLS-based thermal to visible recognition algorithm. While our algorithms provide some improvement, the performance on the more realistic UND dataset is still quite poor. There is still much to be done to solve this challenging problem.

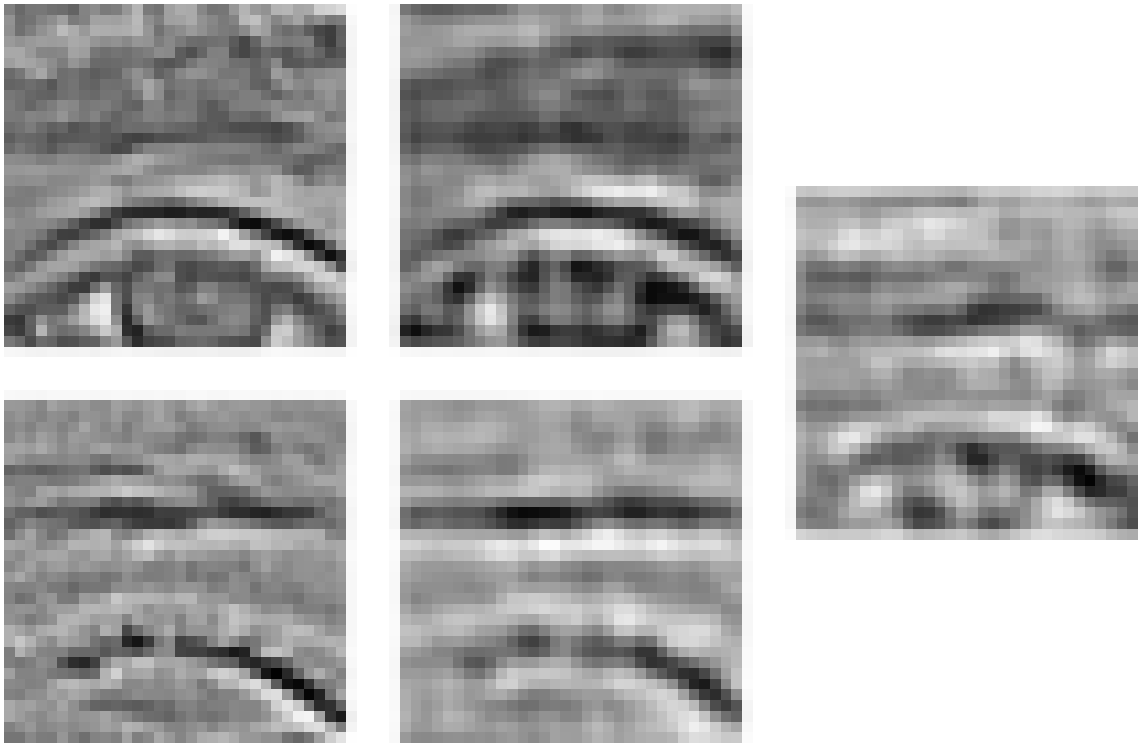
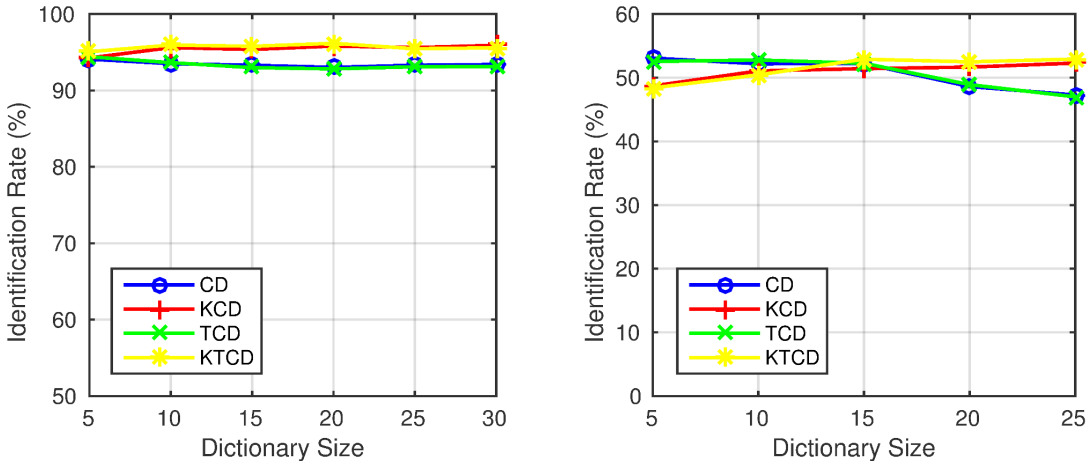


Figure 2.3: For this figure we trained coupled dictionaries on pixel values of visible and infrared image patches. We use the pixel values as features (as opposed to the HOG features used in the rest of the experiments) in order to demonstrate the ability of coupled dictionaries visibly. The left most column shows the preprocessed visible (top) and corresponding IR (bottom) patches. The middle column shows the reconstructions of the left-column patches from their respective sparse codes and domain dictionaries. The image on the right shows the reconstruction of the visible patch from the sparse codes of the IR patch using the visible domain dictionary. The similarity between the right and top middle patches demonstrates the ability of coupled dictionaries.

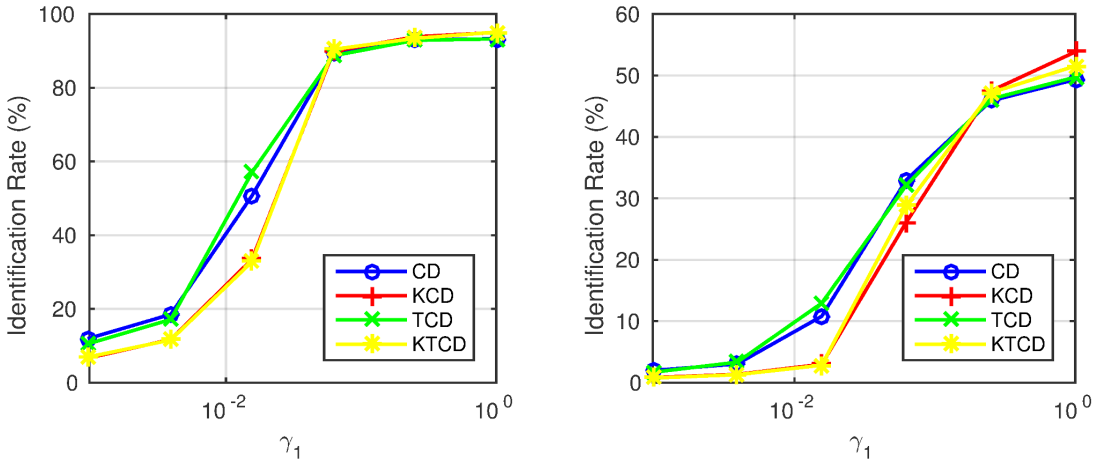
Figure 2.4: Preprocessed and aligned sample images from the WSRI dataset. The visible images (top) correspond to the thermal images (bottom) directly below them.

Figure 2.5: Preprocessed and aligned sample images from the UND dataset. The visible images (top) correspond to the thermal images (bottom) directly below them.



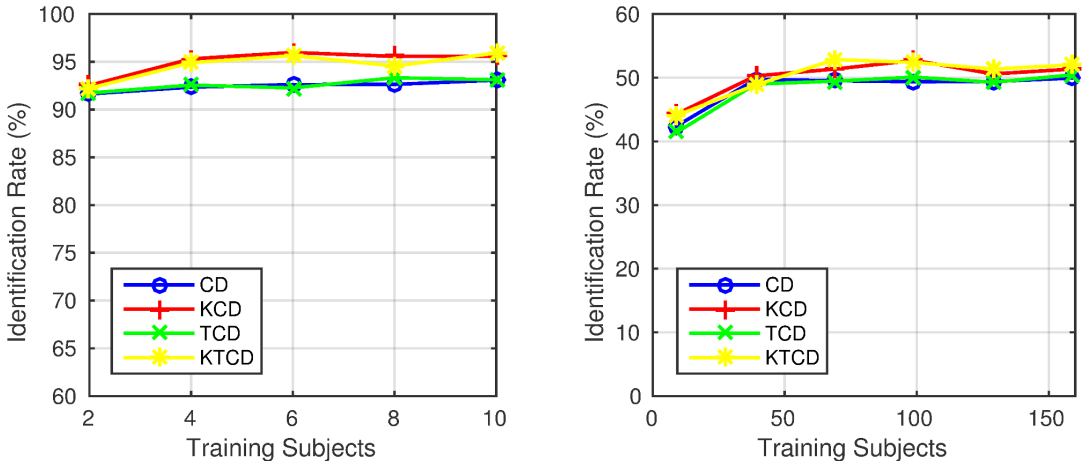
(a) (b)

Figure 2.6: Performance for varying number of dictionary atoms on the WSRI (a) and UND (b) datasets.



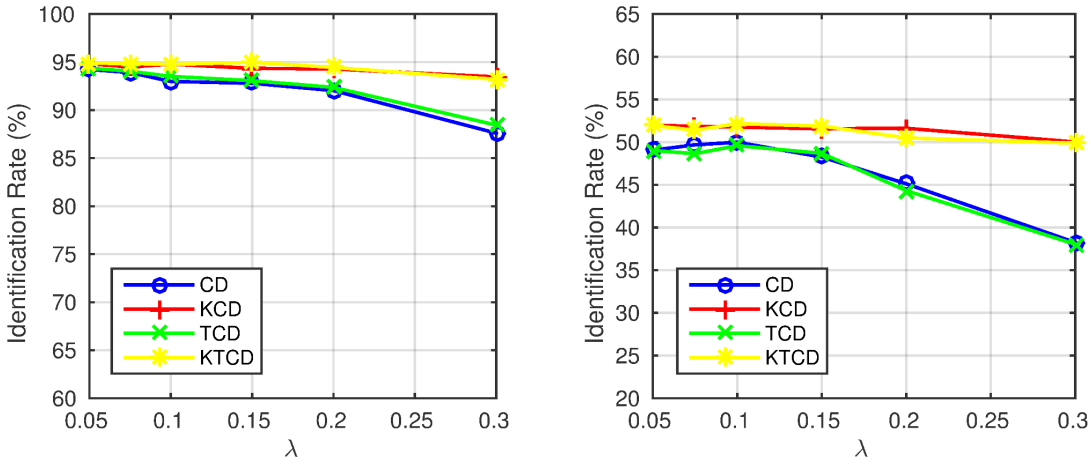
(a) (b)

Figure 2.7: Performance for varying values of γ_1 on the WSRI (a) and UND (b) datasets.



(a) (b)

Figure 2.8: Performance for varying number of training subjects on the WSRI (a) and UND (b) datasets.



(a) (b)

Figure 2.9: Performance for varying values of λ on the WSRI (a) and UND (b) datasets.

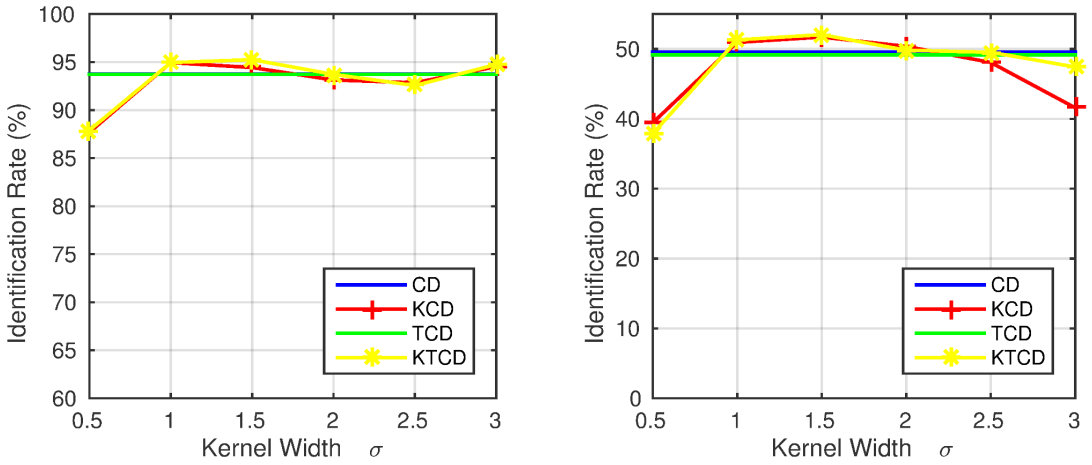


Figure 2.10: Performance for varying kernel widths on the WSRI (a) and UND (b) datasets.

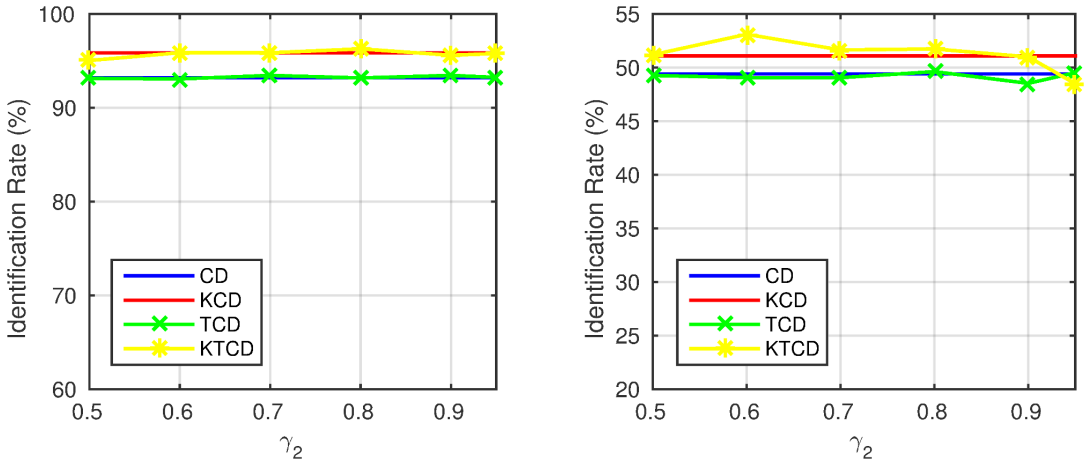


Figure 2.11: Performance for varying values of γ_2 on the WSRI (a) and UND (b) datasets.

Chapter 3: Deep CNN Approach for Near-Infrared HFR

3.1 Introduction

In recent years, a significant amount of research in the computer vision community has focused on Heterogeneous Face Recognition [54]. The objective of HFR is to be able to perform face recognition with probe images captured via alternative sensing modalities. Due to the ubiquity of visible cameras, virtually all face galleries are comprised of visible light images. Thus the main challenge of HFR is enabling the cross-domain comparison of probe images to visible-light gallery images. Most works address this problem by selecting [55, 56] or learning [57, 58] features that, among other things, are more invariant across domains than raw pixels. While these methods achieve some degree of success, due to data constraints, the features used are almost always local in nature or learned using shallow models.

Considering that most HFR datasets have large feature dimension (at least 100x100 pixels) and only a moderate number of images for training (fewer than 10000), any attempt to learn global features with deep models will likely overfit to the training set. To get around this problem, HFR algorithms learn features to represent patches rather than whole images. This alleviates the data constraints in two ways: by increasing the number of training samples (because there are multiple

patches per image) and by decreasing the feature dimension (because patches are smaller than whole images). Although some works do consider global features [57, 58], the limited training data restricts them to using simple models.

While learning local features and using simple models make HFR more tractable, recent computer vision research has shown that global features and deep models tend to outperform local features and shallow models. Even just using a simple spatial pyramid approach (i.e. a naive globalization of a local histogram feature) can improve system performance for many applications [59]. This is especially true for current state-of-the-art algorithms in face recognition, which use deep networks to extract global representations for face images. This allows for a richer representation that can model the face as a whole rather than as a collection of parts. In this work, we show how to use deep learning to leverage a large visible face recognition dataset to learn global features for HFR.

Deep networks perform well due to their ability to learn information from extremely large (sometimes unlabeled) datasets. In fact, the recent surge in deep learning research was spurred in part by the breakthrough method of Hinton et al. [60] to initialize deep networks. They proposed to greedily pretrain the layers of a network with a generative model. This helps the stochastic gradient descent learning algorithm by guiding it towards better local minima [61]. In the case where there is limited training data, it also helps to prevent overfitting. In this work, we propose to adapt the same paradigm to HFR. Like [60], we initialize networks using the abundant data (labeled visible faces) and then tweak them with the scarcer application-specific data (labeled/corresponding visible and infrared faces).

More specifically, our approach is as follows. We first train a deep convolutional neural network on a large visible face dataset for identification. We then use the trained network to initialize networks that will be used to extract features from visible and near-infrared images for HFR. Finally, we further optimize the HFR networks on the HFR training data to couple their output features, making them suitable for cross-modal face recognition.

The main contributions of this chapter are as follows:

- We present the first method to use a deep model to learn global features for HFR.
- We learn coupled deep convolutional neural networks to map visible and NIR faces into a domain-independent latent feature space where they can be compared directly.
- We show how to leverage a large visible face recognition dataset to prevent overfitting of the networks.

The remainder of the chapter is organized as follows. In Section 3.2 we review relevant HFR, face recognition, and deep learning literature. In Section 3.3 we describe our approach. In Section 3.4 we provide implementation and engineering details. In Section 3.5 we present experimental results and discuss their implications. In Section 3.6 we describe three improvements we made to our baseline method. In Section 3.7 we present experiments we performed to evaluate those improvements. Finally, we conclude the paper in Section 3.8.

3.2 Related Work

3.2.1 NIR-Vis Face Recognition

HFR has been extensively researched over the past decade, with NIR being one of the most prominent alternative sensing modalities. Here we briefly describe some recent works on the subject. Klare and Jain [39] use kernel similarities to a set of training subjects as features. Zhu et al. [56] propose a new feature descriptor and a transductive model for domain-adaptive matching. Yi et al. [58] use restricted Boltzmann machines to reduce the domain difference locally and ignore initial PCA coefficients to do the same globally. Jin et al. [62,63] learn local features to represent images consistenly across domains and discriminatively within each domain. Juefei-Xu et al. [57] use cross spectral joint dictionaries to reconstruct visible light images from near IR images and vice-versa. They then compare images directly. A common drawback of all these methods is they do not use a deep, global representation of face images, which has been shown to produce superior results for face recognition.

3.2.2 Deep Learning Face Recognition

Face recognition has been researched heavily for decades [35]. With few exceptions [64] recent state-of-the-art algorithms have been dominated by deep convolutional neural networks trained on extremely large datasets to produce global feature representations. Taigman et al. [24] were the first to train a deep neural network for face recognition. They trained on the private Social Face Classifica-

tion dataset which contains 4.4 million labeled images of 4030 subjects. They used convolutional, locally connected, and fully connected layers in their network. Additionally they fine-tune the parameters for verification with a siamese network. They extended their work in [25] and increased the dataset size to 500 million images of 10 million subjects. Sun et al. [26, 65–67] make a variety of adjustments to improve the performance of deep networks for face recognition including a joint verification-identification loss function, different network architectures, and Bayesian metric learning. They use the private CelebFaces [65] (202,599 images of 10,177 subjects) and WDRef [68] (99,773 images of 2,995 subjects) datasets to train their networks. They also use convolutional, locally connected, and fully connected layers. Parkhi et al. [28] learn a feature embedding by using a triplet loss function. They also detail the steps they took to create their 2.6 million image dataset. Schroff et al. [27] leveraged a 200 million images of 8 million subjects to train a network. This has the best performance to date on Labeled Faces in the Wild (LFW) [69], a standard unconstrained face recognition benchmark.

3.2.3 Distillation

Distillation was first introduced by Hinton et al. [70] as a means to condense an ensemble of neural networks into a single neural network by extracting information from network logits. It has since been used by Gupta et al. [71] to help train in modalities with limited training data. Su and Maji [72] take a similar approach when training a model for use with low quality data.

3.3 Our Method

3.3.1 Network Structure and Initialization

The network structure we use throughout this work (detailed in Table 3.1) is from the Googlenet [73] family of networks (i.e. deep with small convolutional filters). It is comprised of five major sections connected in series. Each section contains the following in order: a convolutional layer, a rectified linear unit layer, a second convolutional layer, a second rectified linear unit layer, and a max pooling layer. The only exception is that the fifth (and last) pooling layer is an average pooling layer. The output of the last section is fed to a fully connected layer whose output is evaluated by a softmax loss function for identification. We build the network entirely of convolutional layers (as opposed to locally and fully connected layers) to minimize the number of parameters so as to reduce the risk of overfitting. This is not a concern during initial training, but can become an issue when tweaking the network for HFR due to limited HFR training data.

We train the network with the standard stochastic gradient descent (SGD) and backpropagation on the CASIA WebFace Database [74]. It is comprised of 494,414 images of 10,575 subjects, which is large enough to provide sufficient generalization. We held out 10,000 images for validation purposes and, after training, the network was able to classify 80 percent of them correctly. We also test the network on Labeled Faces in the Wild (using the negative distances of pool5 layer features as similarity scores) and achieved 96 percent accuracy. While not state-of-the-art on

LFW, the network is more than good enough to serve as an initialization for our HFR application. Within-domain identification experiments on the HFR training data yield greater than 99.5 percent accuracy for both visible and near-infrared images. Furthermore, it has significantly fewer parameters than state-of-the-art networks, which lowers the risk of overfitting when adapting the network for HFR. For the remainder of this paper, we will refer to this network as IDNet.

3.3.2 HFR Networks

While we would ultimately like to learn a network for identification on infrared faces, we do not have access to infrared face images of probe subjects for training. Instead, we train two networks for cross-modal verification: VisNet (for visible images) and NIRNet (for near infrared images). We initialize these networks as copies of IDNet, with the exclusion of the fully connected softmax classifier. The fully connected layer contains more parameters than the rest of the network combined, so removing it helps to prevent overfitting. Additionally, the outputs of the fully connected layer are each highly tuned for a specific subject in the WebFace dataset, and are not likely to generalize well to arbitrary subjects. After removing layer fc6, the last layer of each network is pool5 and the output of each is one 320-channel pixel.

We train VisNet and NIRNet to couple their output features by creating a siamese network [75, 76] as shown in Figure 3.3. Although we initialize them with the same values, unlike the normal use of a siamese network, we do not force the two

	Name	Type	Filter Size	Stride	Output Size	Params
Section 1	conv11	Convolution	$3 \times 3 \times 32$	1	$100 \times 100 \times 32$	288
	conv12	Convolution	$3 \times 3 \times 64$	1	$100 \times 100 \times 64$	18.4K
	pool1	Max Pooling	2×2	2	$50 \times 50 \times 64$	0
Section 2	conv21	Convolution	$3 \times 3 \times 64$	1	$50 \times 50 \times 64$	36.7K
	conv22	Convolution	$3 \times 3 \times 128$	1	$50 \times 50 \times 128$	73.7K
	pool2	Max Pooling	2×2	2	$25 \times 25 \times 128$	0
Section 3	conv31	Convolution	$3 \times 3 \times 96$	1	$25 \times 25 \times 128$	111K
	conv32	Convolution	$3 \times 3 \times 192$	1	$25 \times 25 \times 192$	166K
	pool3	Max Pooling	2×2	2	$13 \times 13 \times 192$	0
Section 4	conv41	Convolution	$3 \times 3 \times 128$	1	$13 \times 13 \times 128$	221K
	conv42	Convolution	$3 \times 3 \times 256$	1	$13 \times 13 \times 256$	295K
	pool4	Max Pooling	2×2	2	$7 \times 7 \times 256$	0
Section 5	conv51	Convolution	$3 \times 3 \times 160$	1	$7 \times 7 \times 160$	369K
	conv52	Convolution	$3 \times 3 \times 320$	1	$7 \times 7 \times 320$	461K
	pool5	Avg Pooling	7×7	1	$1 \times 1 \times 320$	0
	fc6	Fully Connected			10575	3.38M
	cost	Softmax			10575	0

Table 3.1: IDNet layer details

halves of the network to share weights during training. This allows the networks more freedom to capture features that manifest differently in visible and NIR images. We use the contrastive loss [76] of the outputs of the VisNet and NIRnet as the loss function for the network. The contrastive loss L on vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ can be written as follows,

$$L(\mathbf{x}, \mathbf{y}) = \begin{cases} \|\mathbf{x} - \mathbf{y}\|_2^2 & \text{if } l_x = l_y \\ \max(0, (p - \|\mathbf{x} - \mathbf{y}\|_2))^2 & \text{otherwise} \end{cases} \tag{3.1}$$

where $\mathbf{x}$ and $\mathbf{y}$ have respective labels l_x and l_y, and p is a tuneable parameter. Minimizing the contrastive loss makes the distance between images of the same subject as small as possible while ensuring images of different subjects are at least a distance of p away from each other.

As with IDNet, we train using stochastic gradient descent and calculate the gradients using the backpropagation algorithm. Whereas in the IDNet training a sample consists of a visible image and its corresponding identity label, here, a sample consists of a visible image, an infrared image, and a binary label indicating whether the images depict the same subject.

3.4 Implementation Details

3.4.1 Image Preprocessing

We perform very minimal image preprocessing. We first align all faces using [77] from the Dlib C++ library [78]. This method works well on NIR images in

Figure 3.1: Aligned and cropped Webface images.

addition to visible ones. Next we crop the faces and resize them to be 100×100 pixels. We then convert the images to gray-scale and subtract the mean face image of the WebFace dataset. We do not scale or filter the images in any way. Sample aligned and cropped images (prior to mean subtraction) are shown in Figures 3.1 and 3.2.

3.4.2 IDNet Details

We train IDNet using the Caffe [79] deep learning framework for 200,000 iterations with a batch size of 256 images. We initially set the learning rate to be .01 and reduce it by a factor of ten every 80,000 iterations. We set the momentum to .9

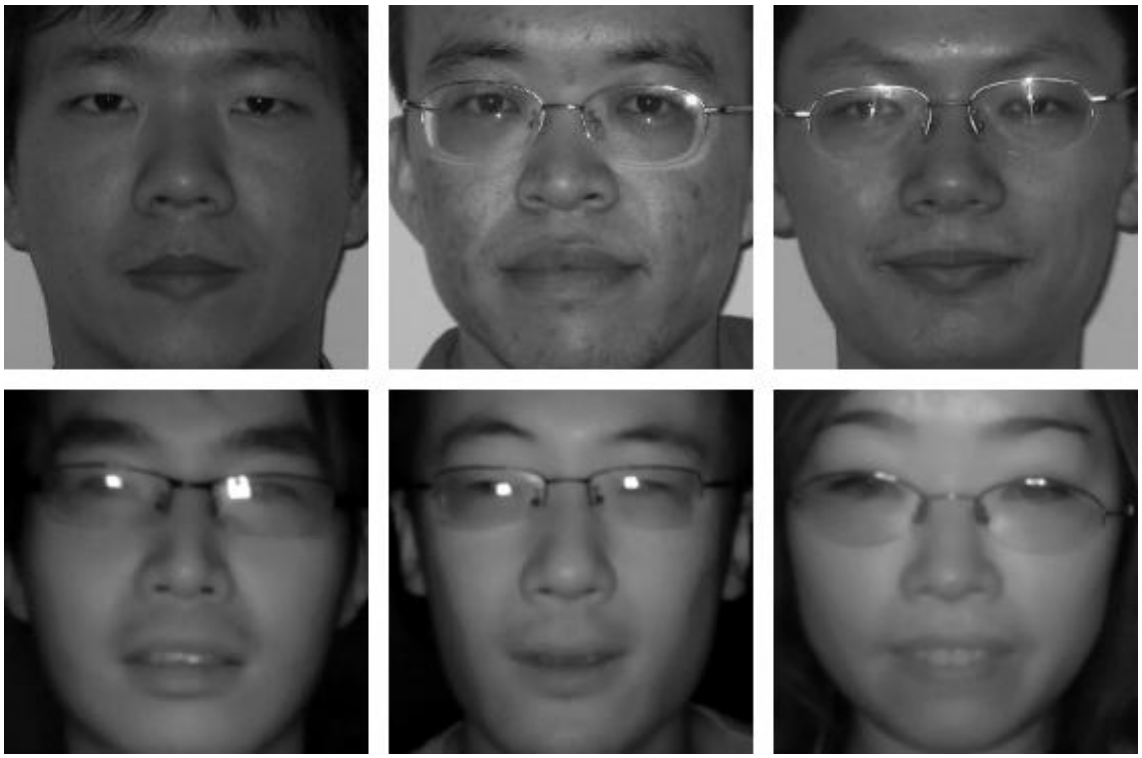


Figure 3.2: Aligned and cropped NIR-VIS 2.0 face images. The top row is visible-light and the bottom row is near-infrared.

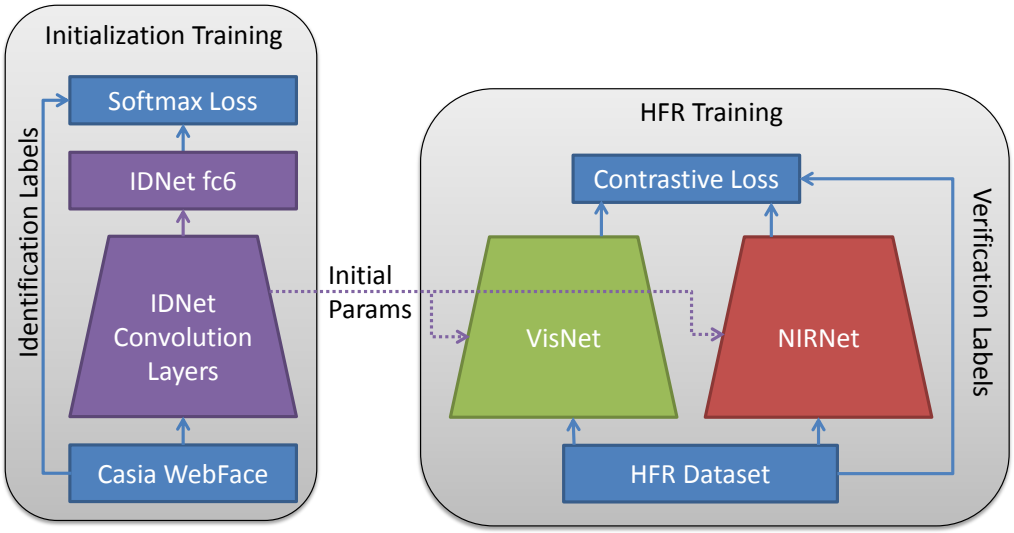


Figure 3.3: Network Diagram

and the weight decay to .0005. We use a single NVIDIA 12GB GeForce GTX Titan X GPU to train IDNet which takes approximately two days.

3.4.3 HFR Details

When training the HFR networks, we are given training data consisting of a set of labeled visible faces and a set of labeled infrared faces. None of the subjects in the testing set are present in the training set. We use all possible same-subject visible-infrared image pairs as positive samples. Because there are significantly more different-subject image pairs, we do not use all of them, but rather we balance our training set by including the same number of negative samples as positive samples. We choose the different-subject pairs randomly.

While training IDNet from scratch takes two days, tweaking VisNet and NIR-

Net can be done much quicker with IDNet as a good initialization point. This allows us to choose some parameters by cross-validation. For all experiments we use a batch size of 64 image pairs. For the learning rate policy, we halve the learning rate every 1000 iterations and set the momentum to .9. We found that both the loss function and performance saturated before 4,000 iterations, allowing us to train the network in about 20 minutes. Parameters we choose with cross-validation include the initial learning rate, the weight decay, the contrastive loss parameter, and the distance metric used for nearest neighbor classification (ℓ^2-norm, ℓ^1-norm, or cosine).

Although we generate as many training samples as possible while maintaining a balanced dataset, the HFR network training is still prone to overfitting due to too little data and too many parameters. We alleviate the overfitting problem by forcing the convolution kernels of some of the network layers to maintain the original IDNet values. This reduces the number of parameters the training algorithm must learn, lessening the amount of training data needed.

Ideally we would determine which layers to fix entirely through cross-validation. Unfortunately, with each half of the network having 10 convolutional layers, there are $2^{10} \times 2^{10} \approx 10^6$ combinations to test (where a combination indicates for each layer whether it will be fixed to its initial value or allowed to float during training). Instead, we use intuition to select a small subset and cross-validate over those. We first assume that if one layer in a section is fixed, then the other must be fixed as well (i.e. if conv32 in VisNet is fixed, then conv31 in VisNet must also be fixed). We then further assume that at least one of the first and last sections must be floating (i.e. not fixed) and that all floating layers in a network must form a contiguous

NIR-VIS 2.0	Rank 1	Std. Dev.	FAR=.001
IDNet	58.6	2.0	39.0
CDFL [62]	71.5	1.4	55.1
LMCFL [63]	75.7	2.5	55.9
[57]	78.5	1.67	85.8
C-CBFD [80]	81.8	2.3	47.3
[58]	86.2	0.98	81.3
Our Method	**87.5**	0.83	76.1

Table 3.2: Performance on View 2 of CASIA NIR-VIS 2.0 Face Database

block. Our assumptions reduce the combinations from more than a million down to the 81 shown in Table 3.4.

3.5 Experiments and Discussion

3.5.1 NIR-VIS 2.0

The CASIA NIR-VIS 2.0 database [82] is the largest publicly available NIR HFR dataset. It contains 17,580 total images of 725 subjects. The dataset contains two views: View 1 for algorithm development and parameter tuning, and View 2 for performance reporting. View 1 contains separate testing (probe/gallery) and training sets with different subjects. The training set has 2,480 visible images and 6,270 NIR images of 357 subjects. The testing set has 6123 NIR probe images of the remaining 358 subjects and one visible gallery image per subject. View 2 has 10

HFB	Rank 1	FAR=.01	FAR=.001
IDNet	80.9	70.4	36.2
P-RS [39]	87.8	98.2	95.8
C-DFD [81]	92.2	85.6	65.5
THFM [56]	99.28	99.66	98.42
[58]	99.38	-	92.25
Our Method	97.58	96.9	85.0

Table 3.3: Performance on View 2 of CASIA HFB Face Database

NIR-VIS 2.0 View 1		Floating NIRNet Sections								
Identification Rate		1	1-2	1-3	1-4	All	2-5	3-5	4-5	5
Floating VisNet Sections	1	71.8	81.4	86.5	87.8	87.8	87.9	87.4	85.8	83.4
	1-2	80.7	81.5	85.7	87.6	87.5	87.7	87.6	85.6	82.3
	1-3	86.2	85.7	83.4	82.5	83.0	83.1	81.8	79.9	78.9
	1-4	86.1	86.7	82.0	77.0	73.5	73.7	72.6	70.7	70.2
	All	85.8	86.8	82.0	72.5	61.1	61.2	60.1	58.8	60.9
	2-5	85.7	86.6	81.7	73.3	61.2	61.2	59.7	58.6	60.9
	3-5	86.0	86.2	80.9	72.0	60.3	60.7	59.8	58.6	60.6
	4-5	85.5	86.0	80.4	71.0	61.1	61.1	59.7	60.9	60.0
	5	84.9	85.5	82.3	71.5	62.0	61.3	62.2	59.8	60.1

Table 3.4: View 1 performance variation based on which sections of NIRNet and VisNet are altered during training. The rest of the sections are fixed to their initial IDNet values.

sub-experiments each of which has the same setup as View 1 with slightly different numbers of images. NIR-VIS 2.0 is by far the most difficult NIR HFR dataset. In particular, the combination of pose variation, large gallery size, and single gallery image per subject make the dataset challenging. We view this dataset as the most relevant to real-world scenarios due to these challenges. Table 3.2 shows our method outperforms all others on View 2 of this dataset. Additionally, the ROC curve on View 2 is shown in Figure 3.4.

3.5.2 HFB

The CASIA HFB dataset [83] is an older and thus more widely used NIR HFR dataset. It has 5,098 total images from 200 subjects. It has a similar protocol setup to NIR-VIS 2.0 (View 1 for tuning parameters, ten View 2 experiments for reporting results). View 1 (and each experiment in View 2) split the dataset into 100 training subjects and 100 testing subjects. In View 1, the training set contains 1,036 visible images and 1,438 NIR images, and the testing set contains 1,059 visible gallery images and 1,542 NIR probe images. The experiment protocols in View 2 have similar statistics. HFB is less challenging than NIR-VIS 2.0 because there are fewer gallery subjects and multiple gallery images per subject. Additionally, there is significantly less pose variation in HFB as there is in NIR-VIS 2.0. These factors make HFB less relevant to real world applications. It's worth noting one aspect of HFB that makes it more difficult to train with: the relatively small number of training subjects. Models are more likely to fit to those specific subjects rather than

generalizing. Our results on View 2 of HFB and those of other methods are shown in Table 3.3 and our ROC curve is shown in Figure 3.5.

3.5.3 Layer Fixing Cross-validation

Table 3.4 shows the results of a cross-validation experiment used to choose which VisNet and NIRNet layers to alter during training. In addition to useful information for parameter selection, it also provides insight into the strengths and weaknesses of learning local and global features (see Table 3.5 for the localness of the features output by different network layers). The configurations that train only local features (the four in the upper left corner) perform significantly worse than those that consider global features. This supports our claim that it is important to learn features with a deep, global model. In the same vein, the configurations that produce the best results almost all allow learning in section 4 or section 5 in either VisNet or NIRNet. This shows that learning global features with a deep model can improve HFR performance. Additionally, the bottom right quadrant of the table shows the difficulty in learning global features and why they are generally not used in HFR. Any configuration where both nets have one of their two highest level sections (4 and 5) floating causes the network to overfit and perform poorly. This is how any deep approach to HFR would perform given the limited training data. We avoid this problem by fixing the high-level layers of one of the network halves to the strong initialization, yielding the performance shown in the upper right and bottom left quadrants. Without the IDNet initialization (i.e. with random initialization),

we would have to train every layer of both VisNet and NIRNet, which yields an identification rate of 61.1 percent.

Another point that can be inferred from Table 3.4 is the relative importance of altering VisNet versus altering NIRNet. While the difference is usually small, it is generally better to alter more layers of NIRNet and leave more layers of VisNet fixed. This can be seen by comparing the identification rate of a combination with the that of the combination with swapped NIRNet and VisNet floating sections (swapping the row and column values on the table). For example, the performance when allowing all NIRNet sections and VisNet sections 1-2 to float is 87.5 percent, while the swapped version (NIRNet 1-2 and all sections of VisNet float) correctly identifies at a slightly lower rate (86.8 percent). Intuitively this makes sense as IDNet was initially trained to extract discriminative features from visible images and thus does not need to be tweaked as much when being used for that purpose.

3.5.4 Gallery Size

While the NIR-VIS 2.0 and HFB databases provide excellent benchmarks for NIR HFR, they do not cover all real-world scenarios. For example, it is possible to have a gallery of thousands or more subjects. In this section, we examine how the performance degrades with increasing gallery size. To accomplish this task, we append visible face images (one per subject) from the WebFace dataset used to train IDNet to the NIR-VIS 2.0 galleries. This provides a simulation of a situation with significantly more subjects. The results in Figure 3.6 show that our method is fairly

Layer	Region Size
Input	1×1
pool1	6×6
pool2	16×16
pool3	36×36
pool4	76×76
pool5	100×100

Table 3.5: Sizes of image regions that affect features in a given layer. Each feature in a layer in the first column encapsulates visual information from an image region of corresponding size from the second column.

robust to the increased gallery size.

3.6 Improvements

In this section, we describe three methods that improve the performance of our baseline approach. First, we use cross-modal distillation to leverage more information from the initialization network. In the baseline approach, the networks are trained so that images of the same subject are close together and images of different subjects are farther apart. Cross-modal distillation attempts to enforce the following principle: if a subject looks similar/dissimilar to another subject in the visible domain, then the same *degree of similarity* should hold in the IR domain. As shown in Figure 3.7, this is achieved by training the IR network to reproduce the visible network's logits (classification scores before the softmax output) which can

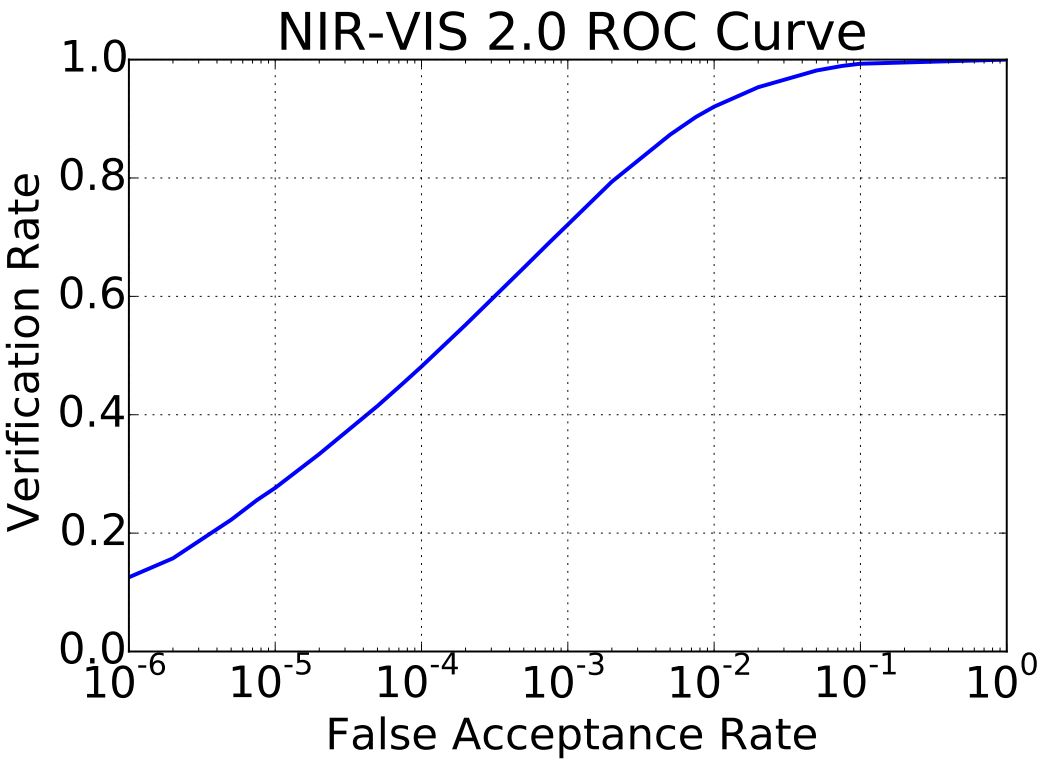


Figure 3.4: ROC Curve for CASIA NIR-VIS 2.0 dataset.

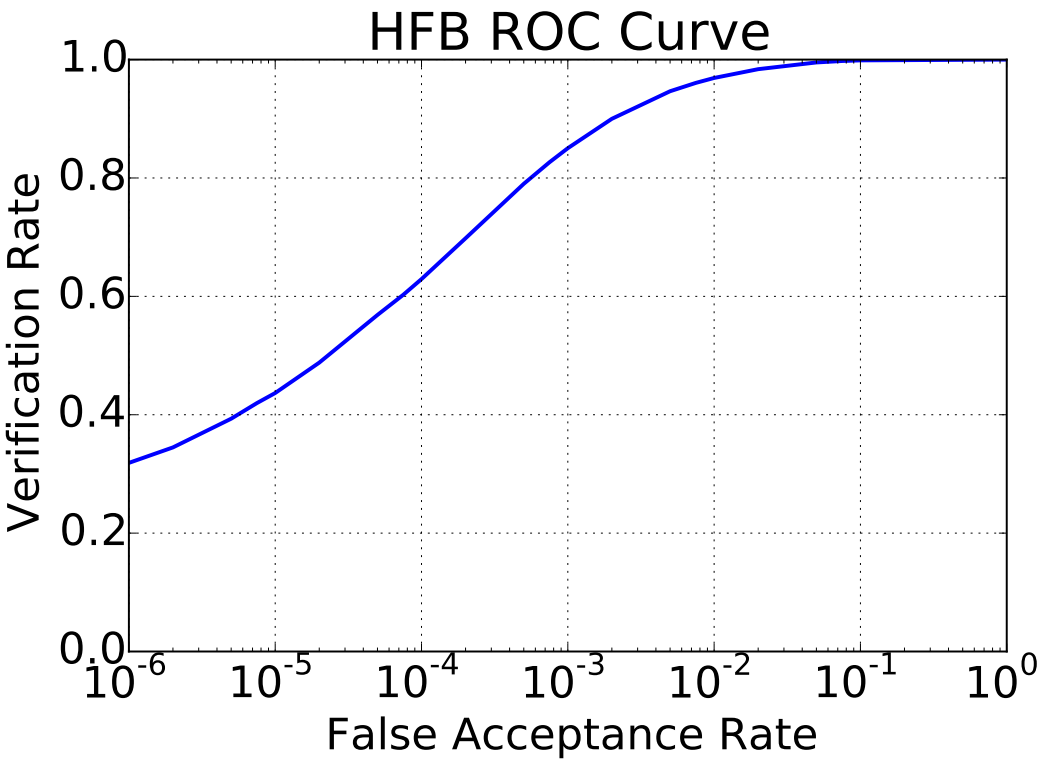


Figure 3.5: ROC Curve for CASIA HFB dataset.

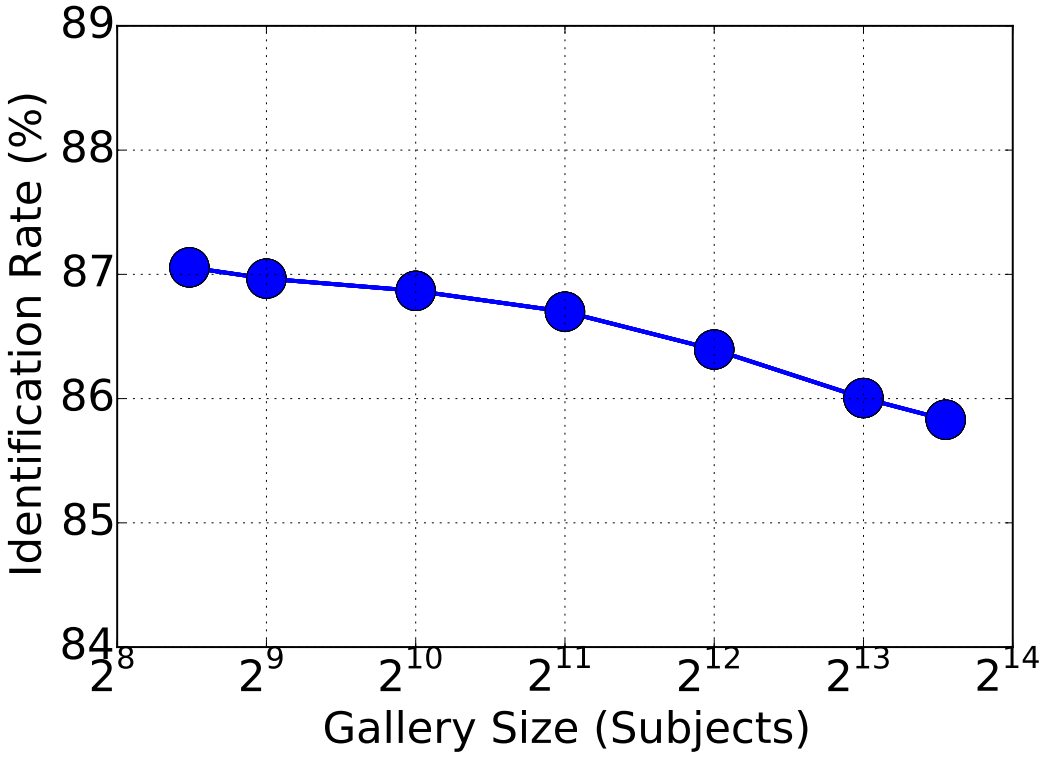


Figure 3.6: Performance degradation on NIR-VIS 2.0 View 2 with increased gallery size.

be thought of as similarity scores to the initialization subjects. This allows the IR network to "distill" information from the visible domain network.

The second proposed improvement is to replace the ℓ_2 distance metric with the ℓ_1 distance metric in the contrastive loss during training. This prevents a few feature components from dominating the optimization and allows all the features learned by the initialization network to have comparable effects during the HFR training.

The third improvement we present is to use a better initialization network. We accomplish this by training the initialization network for over twice as many iterations. This yields a more thoroughly trained network with more meaningful, discriminative features for identifying faces.

3.6.1 Cross-modal Distillation

The first method we use to improve performance is cross-modal distillation. The main assumption is that if a subject looks similar or dissimilar to another subject in visible imagery, that should carry over into the near-infrared imagery. Cross-modal distillation does this by using information about the initialization subjects in the pretrained network. While that information is not included in any dataset, we can infer it from the initialization network output logits.

The initialization network is trained to recognize 10,575 celebrities from their face images. Given a face, this is done by calculating similarity scores (sometimes called logits) for each possible celebrity in the dataset. The network classifies the

face image as belonging to the celebrity with the highest score. The network can perform this operation on any face image, regardless of the subject in the image. While it is not useful for recognition of subjects which are not in the initialization set (like our test and training set subjects), the logits still contain valuable information about the appearance of the face. They can be thought of as a rough estimate for how similar the input face is compared to the celebrity faces in the visible domain. A high logit indicates the input face looks very similar to the celebrity while a low logit indicates the opposite.

As shown in Figure 3.8, we incorporate this extra information into our algorithm by adding a second contrastive loss function that takes the logits as inputs. This contrastive loss operates only on same-subject pairs, so it essentially performs regression on the logits. Basically, we're trying to ensure that the infrared network can recreate the logits that are generated by the visible network. The idea is that in order to accomplish this task, the network is forced to learn more disciminative features in the convolutional layers of the network. Also note that, as with the convolutional weights, the weights of the fully connected layers are initialized with the corresponding values in the initialization network. Without the pretrained values, the computed logits would be meaningless.

3.6.1.1 Engineering Details

We make a few notable changes in the training algorithm due to the incorporation of cross-modal distillation. First, due to the large number of parameters in the

fully connected layers (see Table 3.1), we do not allow them to be adjusted during HFR optimization. Additionally, we found that when training with distillation on the HFB dataset, we did not have to fix as many of the convolutional layers (i.e. the model didn't overfit as easily with distillation). We attribute this to relatively low number of training subjects (100) in the HFB dataset. The additional information from the CASIA WebFace subjects helped to improve generalization.

3.6.2 Contrastive Loss Metric

The second adjustment we make to improve performance is a change in the metric used by the contrastive loss. In the original work, only the Euclidean norm is used with the contrastive loss. On the other hand, when testing, Reale et al. [45] cross-validated over three distance metrics to compare face images (ℓ_1, ℓ_2, and cosine). We thought it might be beneficial to cross-validate over the distance metric used in the contrastive loss as well.

In this work, we only cross-validate over the ℓ_1 and ℓ_2 norms. When using the ℓ_1 norm, the constrastive loss is computed as follows,

$$L(\mathbf{x}, \mathbf{y}) = \begin{cases} \|\mathbf{x} - \mathbf{y}\|_1 & \text{if } l_x = l_y \\ \max(0, p - \|\mathbf{x} - \mathbf{y}\|_1) & \text{otherwise,} \end{cases}.$$

This gives the training algorithm more flexibility to tailor the networks for the data. It turns out that in virtually all experiments, the ℓ_1 norm provides the best results. We believe it tends to be a better fit for this data because it prevents a

few feature components from dominating the optimization. By this we mean that the gradient passed backwards by a given feature component no longer depends on the components magnitude. For example, consider the ℓ_2 norm. One component of gradient of the ℓ_2 norm squared is equal to that component itself. Because of this, components that naturally have a larger discrepancy across domains will have an undue effect on the network parameters when their gradients back-propagated. The gradient of the ℓ_1 norm, on the other hand, is merely the sign of each component and does not depend on the magnitude of the components at all. This prevents a few components from dominating the optimization, allowing for a better solution.

Since we are using two contrastive loss functions (the original one and the one for distillation), we also used cross-validation to determine the distance metric for the distillation loss function. In that case, the ℓ_2 distance yielded better results.

3.6.3 Improved Initialization

The third adjustment we make to improve performance is by providing the networks with a better initialization. Generally, a simple way to improve performance of any deep network is to train it for longer. Usually this will achieve a small increase in performance. In our case, extra training of the HFR networks does not change the performance much because of the limited amount of training data available.

While we do not have sufficient cross-modal training data to warrant training the HFR networks for more iterations, we do have a large initialization dataset

(CASIA Webface). Thus we train the initialization network for a longer period of time (450,000 iterations vs 200,000 iterations). This makes the initialization learn more effective features for recognizing faces (83% vs 80% recognition rate on a validation set from CASIA Webface) and therefore increases the HFR performance.

3.7 Improved Results

We present the results of evaluations of five variants of our method (each method individually, all three together, and the original method as a baseline) in Tables 3.6 and 3.7. From the results, it is clear that all three methods have a positive effect on the recognition performance.

Of the three methods, distillation provides a more modest increase in performance, though it is comparable to the other two on the HFB dataset. We attribute this to the smaller number training subjects and images in HFB. This makes information about additional subjects (through distillation) more valuable. On the other hand, NIR-VIS 2.0 provides more training subjects and images. Thus, the training algorithm does not benefit as much from information about additional subjects. The other two methods (ℓ_1 metric and better initialization) help the training on both datasets, with the better initialization adding a noticeably bigger performance boost in both cases. This demonstrates the remarkable ability of deep networks to soak up information even after they have already been trained for a long time.

Finally, it is clear that the best performance is achieved when all three methods are used together. The recognition rates are improved from 97.58% to 99.52%

HFB	Rank 1	FAR=.01	FAR=.001
Baseline [45]	97.58	96.9	85.0
Distill	98.55	94.7	81.2
L1 Loss	98.68	95.0	82.6
Better Init.	99.08	97.7	87.7
All Three	**99.52**	98.6	91.8

Table 3.6: Performance of our algorithms on View 2 of CASIA HFB Face Database

NIR-VIS 2.0	Rank 1	Std. Dev.	FAR=.001
Baseline [45]	87.1	0.88	74.5
Distillation	87.5	1.04	76.1
L1 Loss	89.4	1.23	79.5
Better Init.	90.8	0.79	77.9
All Three	**92.6**	0.64	81.6

Table 3.7: Performance of our algorithms on View 2 of CASIA NIR-VIS 2.0 Face Database

and from 87.1% to 92.6% on the HFB and NIR-VIS 2.0 datasets respectively. Additionally, the verification rates increase from 85.0 to 91.8 and from 74.5 to 81.6.

3.8 Conclusion

In conclusion, we have presented a novel approach to NIR HFR. We have proposed to use convolutional neural networks to learn deep, global features that

capture discriminative information in NIR and visible face images. Moreover, we coupled the networks so they produce domain-independent features that can be compared directly. We prevented overfitting by initialization with a visible face identification network trained on a very large visible face dataset. We evaluated our approach on two benchmark databases and additionally investigated how our method scales with larger gallery sizes. We proposed three improvements to our baseline method and evaluated their effectiveness as well.

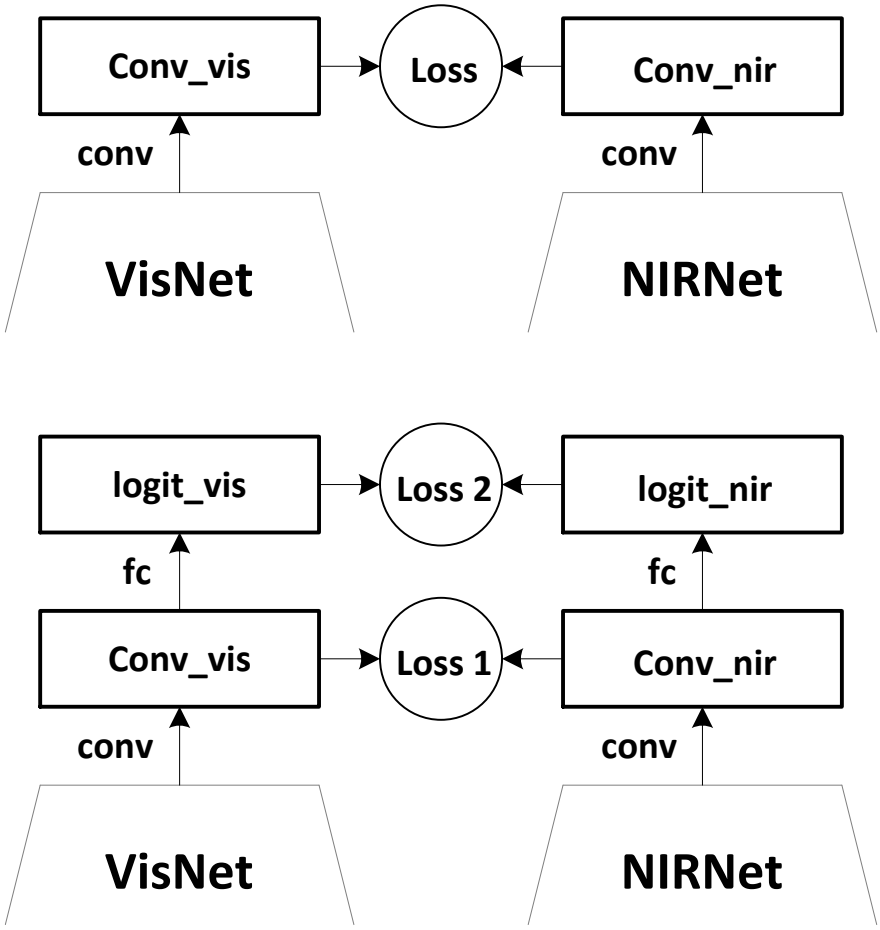


Figure 3.7: Top: Architecture of baseline method. Bottom: Architecture of the proposed network adding a second loss measuring difference of the logits of the two networks to the baseline architecture.

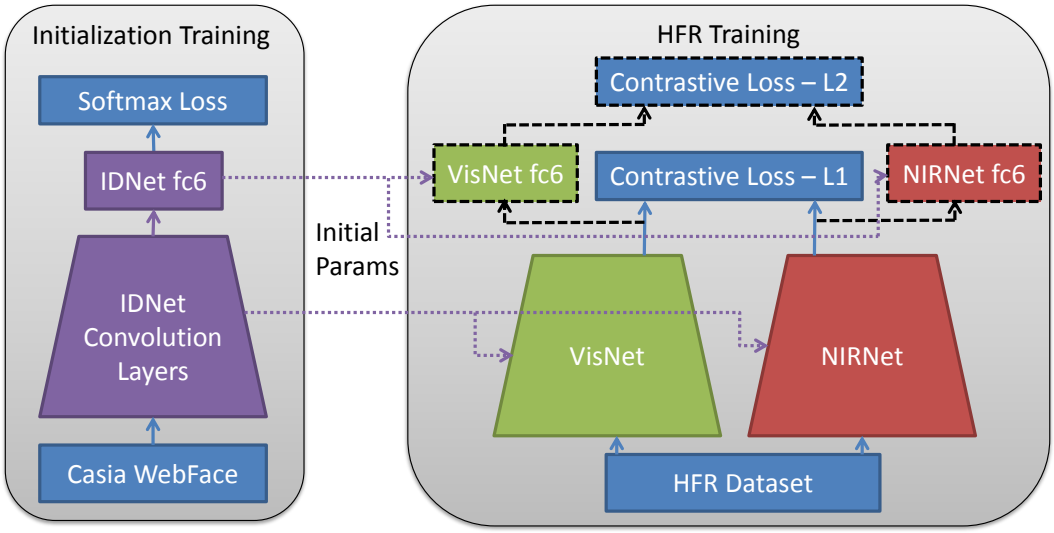


Figure 3.8: Network Diagram: This figure shows a visualization of our training algorithm and how our improvements affect it. The gray section on the left shows the initialization training setup. One of the contributions of this work is to improve this initialization by training it for longer. The gray section on the right shows the HFR training. Black dashed lines indicate any components added to include cross-modal distillation.

Chapter 4: Neural Network Breadth Selection

4.1 Introduction

Virtually all machine learning methods entail choosing a model and using a training algorithm to learn model parameters from data. In most cases, one must set the values of one or more hyper-parameters for the model and/or algorithm. Deep networks are no different. When there are only a few hyper-parameters and training is relatively fast, cross-validation can be used to select optimal values for the hyper-parameters. If cross-validation is too slow, ad hoc methods must be used instead. Deep learning methods fall into the latter category as they have several hyper-parameters for both the model (layer types, nodes per layer) and training algorithm (learning rate, learning policy, weight decay, momentum, etc.) and generally take a significant amount of time to train.

In this work, we provide a method to reduce the complexity of hyper-parameter selection in deep networks. More specifically, we show how to use group lasso regularization [84] to select the appropriate breadth (number of hidden nodes) for each convolutional and fully connected layer of the network. The group lasso penalty encourages parameters to be group sparse (i.e. blocks of the parameters are zeroed). Thus, by incorporating it into the standard backprop [85] stochastic gradient descent

(SGD) and organizing the parameters into blocks based on the hidden node they compute, the optimization will automatically zero out unnecessary hidden nodes. As shown in Figure 4.1, we can then simply remove the zeroed out nodes to obtain a smaller network.

Choosing the appropriate number of nodes for each layer is important. having too few nodes can overly simplify the model, restricting it from capturing the full complexity of the task at hand. This can cause the network to perform sub-optimally. On the other hand, unnecessary nodes require extra computational power to calculate and necessitate more space in memory to store both the extra model parameters and the extra features for each sample. Additionally, having too many nodes yields an overly complex model that can be more prone to over-fitting. This is of particular importance when fine-tuning with limited data as in the problem we focus on: NIR Heterogeneous Face Recognition.

4.2 Related Work

Despite the ubiquity of deep neural networks and the importance of this problem, to the best of our knowledge it has not been addressed beyond an ad hoc approach. Some methods have attempted to solve the related problem of making deep networks smaller. Liu et al. [86] sparsely decompose the filters of convolutional neural networks in order to reduce the redundancy of their parameters. Yang et al. [87] use kernel methods in place of fully connected layers to greatly reduce the necessary number of parameters. Moczulski et al. [88] introduce a new module

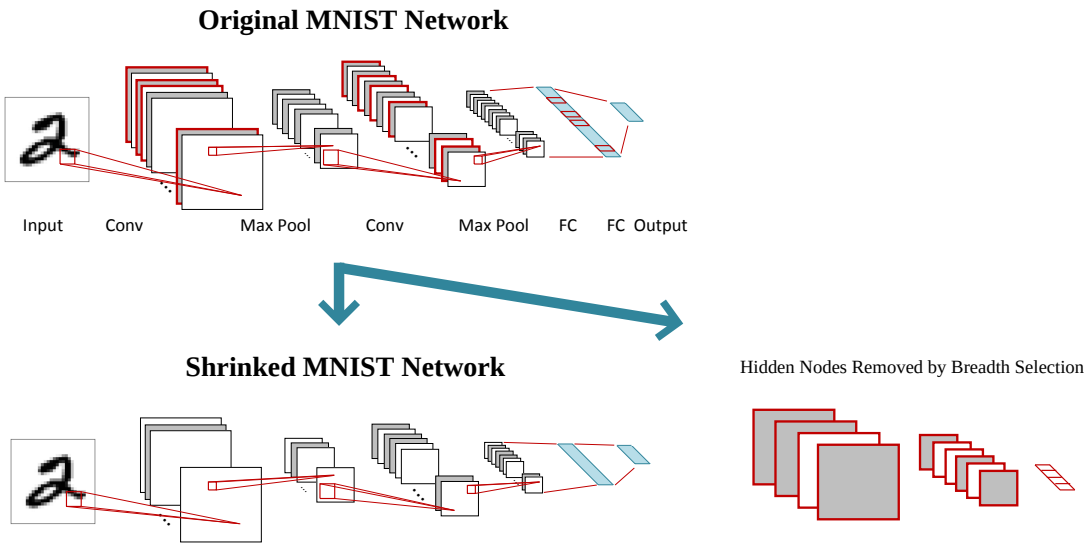


Figure 4.1: A visualization of our method.

consisting of diagonal matrices and the discrete cosine transform to reduce the number of parameters in networks while maintaining good performance. Lebedev and Lempitsky [89] use a method similar to ours to speed up convolution by sparsifying the footprints of convolution filters. Our method can also be used shrink/speed up deep networks, though our primary objective is to determine the optimal network structure.

4.3 Our Method

4.3.1 Formulation

Deep networks are trained by using SGD to solve the following optimization problem,

$$\min_W \frac{1}{M} \sum_{i=1}^{M} L(x_i, W) + \lambda R(W),$$

where W represents the model parameters, M is the number of training samples, x_i is a training sample, L is a loss function, R is a regularization function, and λ is a constant referred to as the weight decay. Let W be the set of parameter matrices $\mathbf{W}^{(i)} \in \mathbb{R}^{N_i \times K_i}$ where N_i is the number of hidden nodes in layer i and K_i is number of parameters required per node for layer i. Traditionally, the squared Euclidean norm is used for the regularization function R. In this case, R can be written as follows,

$$R(W) = \sum_{\mathbf{W}^{(i)} \in W} \frac{1}{2} \|\mathbf{W}^{(i)}\|_F^2 = \sum_{\mathbf{W}^{(i)} \in W} \sum_{j=1}^{N_i} \frac{1}{2} \|\mathbf{w}_j^{(i)}\|_2^2,$$

where $\mathbf{w}_j^{(i)} \in \mathbb{R}^{K_i}$ is the jth row of $\mathbf{W}^{(i)}$.

While the squared Euclidean norm generally works well for optimizing a given network, we want to encourage our training algorithm to use fewer nodes than it begins with. To do this, instead of using the Euclidean norm, we propose to use the group lasso penalty (sometimes called the mixed ℓ_1/ℓ_2 norm) as follows,

$$R(W) = \sum_{\mathbf{W}^{(i)} \in W} 2\|\mathbf{W}^{(i)}\|_{\ell_1/\ell_2} = \sum_{\mathbf{W}^{(i)} \in W} \sum_{j=1}^{N_i} 2\|\mathbf{w}_j^{(i)}\|_2.$$

The group lasso penalty encourages the solutions to be group sparse (i.e. entire blocks of parameters are set to zero). In our case we have organized the parameters so that each block corresponds to a single hidden node. Thus, training with the group lasso penalty will cause the parameters entire nodes to be zeroed out, after

which they can simply be removed from the network.

Our learning algorithm can be summarized as follows. We train a given network using group lasso regularization. We then remove any zeroed hidden nodes. Finally, we continue training the altered network using the squared Euclidean norm regularization to tweak the network and boost performance.

4.3.2 Explanation

The optimization is set up as a balancing act between reducing the loss function and reducing the number of nodes with the weight decay determining how far that balance should skew towards one side or another. The parameters of nodes that do not contribute much to keeping the loss function low have relatively small loss function gradients. Over time, the regularization function gradient dominates for these parameters and they eventually go to zero. On the other hand, the parameters of nodes which greatly contribute to maintaining a small loss have much larger loss function gradients. Over the course of the optimization, the loss function gradients of these parameters dominate and they will not be set to zero.

It is informative to compare the gradients of the squared Euclidean and the group lasso regularization functions to gain a more detailed understanding of our method. In the squared Euclidean case, the gradient for $\mathbf{w}_j^{(i)}$ (the parameters of a given node) is calculated as follows,

$$\nabla \frac{1}{2} \|\mathbf{w}_j^{(i)}\|_2^2 = \mathbf{w}_j^{(i)}.$$

The squared Euclidean gradient effectively reduces each parameter by a fixed percentage (hence the term weight decay). Without the loss function gradients, the optimization will simply step towards $\mathbf{0}$ a fixed fraction of the distance. This has the effect of ensuring no parameter gets too big while also offering very little incentive to shrink already small parameters.

The group lasso gradients are calculated as follows,

$$\nabla 2\|\mathbf{w}_j^{(i)}\|_2 = \frac{\mathbf{w}_j^{(i)}}{\|\mathbf{w}_j^{(i)}\|_2}.$$

As with the squared Euclidean gradient, the group lasso gradient steps towards $\mathbf{0}$. The difference is in the magnitude of that step. The squared Euclidean regularization mandates stepping a fixed percentage of the distance, whereas the group lasso simply steps a fixed amount regardless of the distance to $\mathbf{0}$. In the Euclidean case, as parameters of less important nodes shrink, the regularization affects them less and so they stop shrinking before they get to $\mathbf{0}$. On the other hand, group lasso always effects all filters the same amount, so less important filters will continue shrinking until they are zeroed out. A side-effect of the the group lasso penalty is that nodes with large parameters are not reined in as much as they would be in the Euclidean case. This leads to some filters with larger parameters. To counteract this, we use both the group lasso and squared euclidean norms for regularization.

4.3.3 Gradient Existence

The gradient of the group lasso penalty does not exist when the norm of a block is zero. This can be problematic as the goal of our method is to ensure some blocks have that exact property. We found it easiest to avoid this problem by adding a small constant ϵ to all norm calculations as follows,

$$\nabla 2\|\mathbf{w}_j^{(i)}\|_2 \approx \frac{\mathbf{w}_j^{(i)}}{\|\mathbf{w}_j^{(i)}\|_2 + \epsilon}.$$

For small ϵ, this leaves the gradient virtually unchanged most of the time (when the norm is not close to zero) and ensures it is defined at $\mathbf{w}_j^{(i)} = \mathbf{0}$.

4.4 Experiments

As a sanity check, we first test our method on the MNIST handwritten digit classification dataset. We then use it to shrink a visible face recognition network, which in turn we use as an initialization network for Near-infrared Heterogeneous Face Recognition.

4.4.1 MNIST

The MNIST handwritten digit recognition dataset [90] consists of 70,000 (60,000 for training, 10,000 for testing) 28x28 pixel images labeled based on the numerical digit (0-9) they depict. We use the network structure and solver parameters from the mnist example in the Caffe deep learning framework. The network structure is

shown in Table 4.4.1.

We run experiments with a range of ten different weight decays (log space between .0001 and .004). We take results of these experiments and produce plots showing the trade-off between classification rate and resource usage (shown in Figures 4.2 and 4.3).

Figure 4.2 shows that the number of parameters can be reduced by about two thirds with very little change in performance. Figure 4.3 shows a slightly different picture, where the performance tails off a bit more quickly with less computational power (measured in multiplication operations required to classify a single image). This is because there are significantly more unnecessary nodes in the fully connected layer of the original network compared to the convolution layers and fully connected layers use more parameters but require less computation than convolution layers. The breakdown of the extent to which each layer shrinks is shown in Figure 4.4 where a larger fraction of convolution nodes are retained compared to fully connected nodes. The main take-away from this experiment is that if a network has too many nodes in a layer, Group Lasso regularization can automatically identify and shrink the layer without negatively affecting performance.

4.4.2 Face Recognition Initialization

While our ultimate goal is to improve the performance of deep heterogeneous face recognition from Chapter 2, we accomplish this by first shrinking the initialization network provided to the algorithm. This removes unnecessary filters, making

Layer Type	Dimensions	Stride
Conv	5×5×20	1
Max Pool	2×2	2
Conv	5×5×50	1
Max Pool	2×2	2
Fully Connected	500	-
ReLU	-	-
Fully Connected	10	-
Softmax	10	-

Table 4.1: Caffe MNIST Network

the network less prone to overfitting.

As in Chapter 2, we initially train the deep convolutional neural network shown in Table 3.1 to perform face recognition on the CASIA WebFace Database [74] for 200,000 iterations. Before using this network for Cross-Spectrum Face recognition, we first examine the properties of the trained and shrunk networks. Table 4.2 shows which layers shrank and which did not. It is interesting that the layers that compute lower-level features (i.e. towards the bottom of the network) seem to be more susceptible to being shrunk. This is most likely due to the fact that high-level features are significantly more complicated than low-level features and thus there are many more possible features for them to recognize. This is despite the fact that the lower level layers have fewer nodes to begin with.

Nodes Remaining After Shrinkage					
Weight Decay	0	.0005	.0010	.0015	.0020
conv11	32	17	8	7	7
conv12	64	34	17	14	13
conv21	64	60	35	22	19
conv22	128	128	115	71	51
conv31	96	94	94	92	83
conv32	192	192	192	192	192
conv41	128	106	106	105	105
conv42	256	256	256	255	233
conv51	160	123	123	119	116
conv52	320	320	320	320	320

Table 4.2: Face recognition layer-wise shrinkage

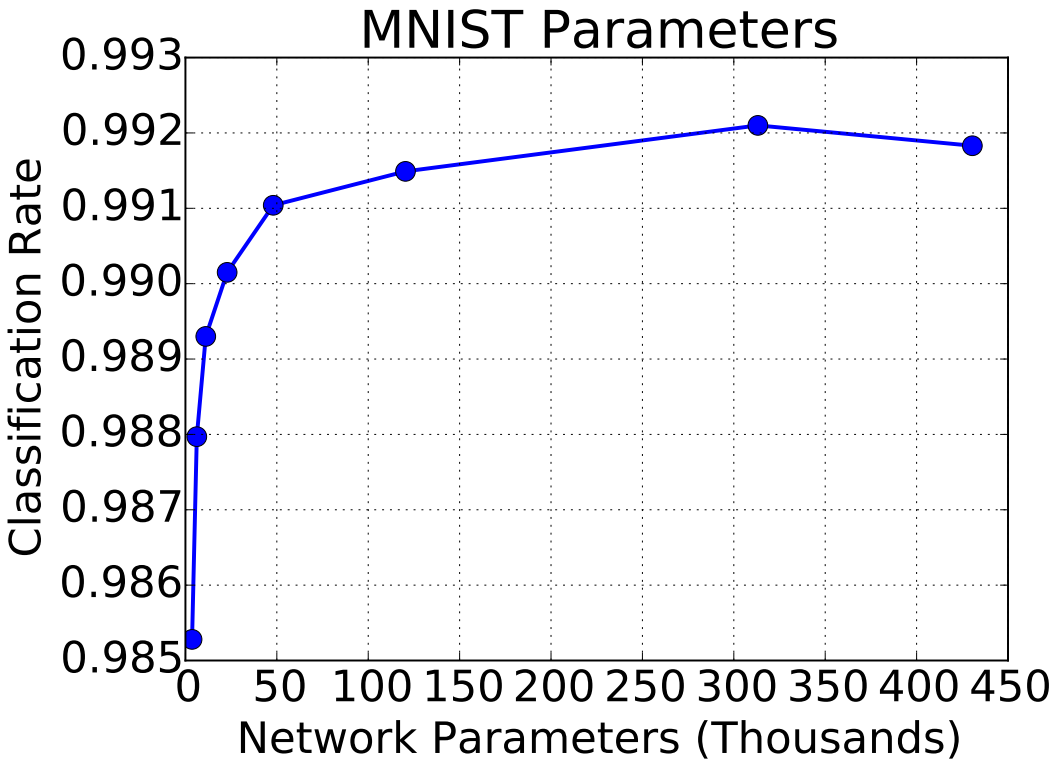


Figure 4.2: Trade-off between the number of network parameters and classification rate on the MNIST dataset. Note the number of parameters in the original network is 430,500.

To ensure that the shrinkage does remove too many nodes from the networks, we examine their performance on a visible face recognition task. Before training the networks, we removed a 10,000 image validation set from the CASIA WebFace dataset. Figure 4.5 shows the ability of the networks to classify the validation set images for varying weight decays. It is clear that even the smallest non-zero weight decay value (.0005) removes enough nodes to hamper recognition. This is most likely

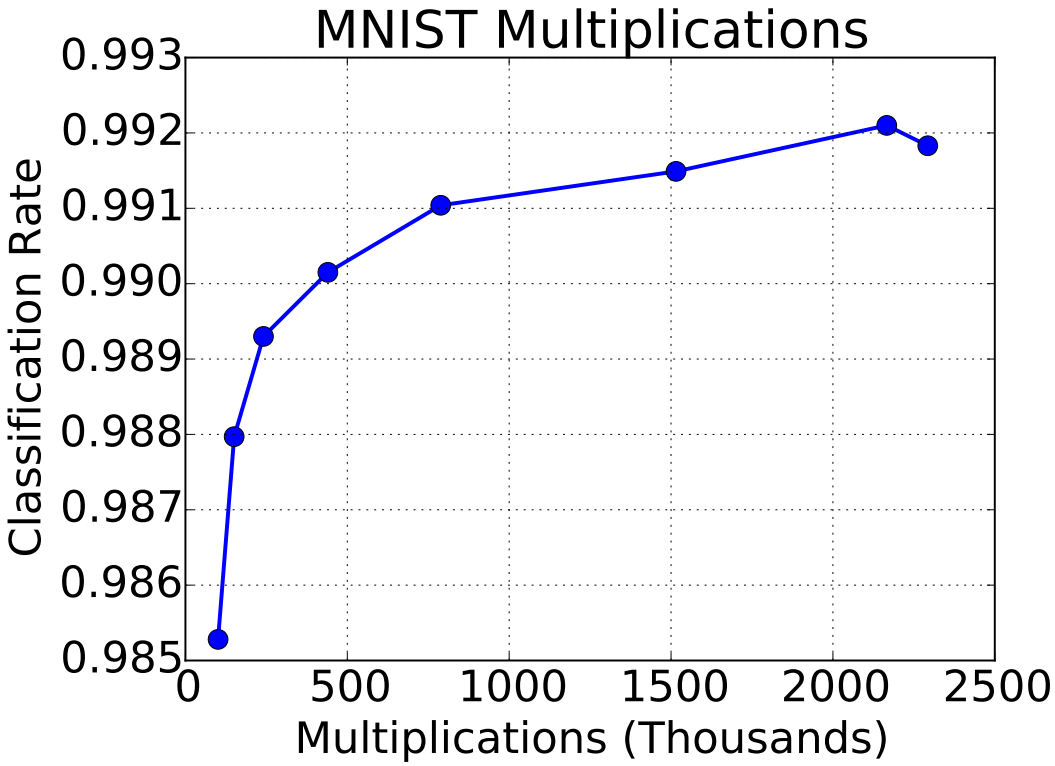


Figure 4.3: Trade-off between the number of multiplications per image and classification rate on the MNIST dataset. Note the number of multiplications required in the original network is 2,293,000.

due to the large number of subjects in the dataset (10,575), which necessitates fine-grained classification and thus a complicated network. As we will show in Section 4.4.3, this is not an issue for HFR because the fine-grained classification problem is dwarfed by the cross-domain gap. Also, having a smaller network helps to reduce overfitting in the data-constrained HFR scenario.

4.4.3 NIR Heterogeneous Face Recognition

We test our algorithm on the two Near-infrared Heterogeneous Face Recognition (HFR) datasets used in Chapter 2: the CASIA NIR-VIS 2.0 database [82] and the CASIA HFB dataset [83]. After training our initialization networks and removing nodes, we follow the rest of the algorithm from Chapter 2 and perform heterogeneous face recognition on two datasets. Performance on View 1 of the datasets are shown in Figures 4.6 and 4.7. For both datasets, it is clear that for the optimal weight decay (.0005), there is a negligible effect on the data set performance. As the weight decay increases and the initialization networks get smaller, the performance eventually decreases.

After optimizing hyper-parameters on View 1, we then evaluate our approach on View 2 of both datasets. Table 4.3 shows that we achieve the best performance to date on the NIRVIS 2.0 dataset. We outperform the previous state-of-the-art by 2.8 percent. Table 4.4 shows that, although we do not achieve state-of-the-art results, we do improve upon the previous deep learning approach of Chapter 2.

4.5 Conclusion

We have presented a method that uses the group lasso penalty to remove unnecessary hidden nodes from a deep neural network. It works by enforcing group sparsity in the network parameters to zero out hidden nodes which do not provide much contribution. The method can be used to make a network smaller without sacrificing performance and to help determine the optimal number of hidden nodes

NIR-VIS 2.0	Rank 1	Std. Dev.	FAR=.001
CDFL [62]	71.5	1.4	55.1
LMCFL [63]	75.7	2.5	55.9
[57]	78.5	1.67	85.8
C-CBFD [80]	81.8	2.3	47.3
[58]	86.2	0.98	81.29
Chapter 2	87.5	0.83	76.1
Our Method	**89.7**	1.06	77.0

Table 4.3: Performance on View 2 of CASIA NIR-VIS 2.0 Face Database

HFB	Rank 1	FAR=.001
P-RS [39]	87.8	95.8
C-DFD [81]	92.2	65.5
THFM [56]	99.28	98.42
[58]	99.38	92.25
Chapter 2	97.58	85.0
Our Method	98.34	87.5

Table 4.4: Performance on View 2 of CASIA HFB Face Database

for each layer. Finally, we used our method to shrink the initialization network of a NIR HFR algorithm and improved upon the state-of-the-art for that problem.

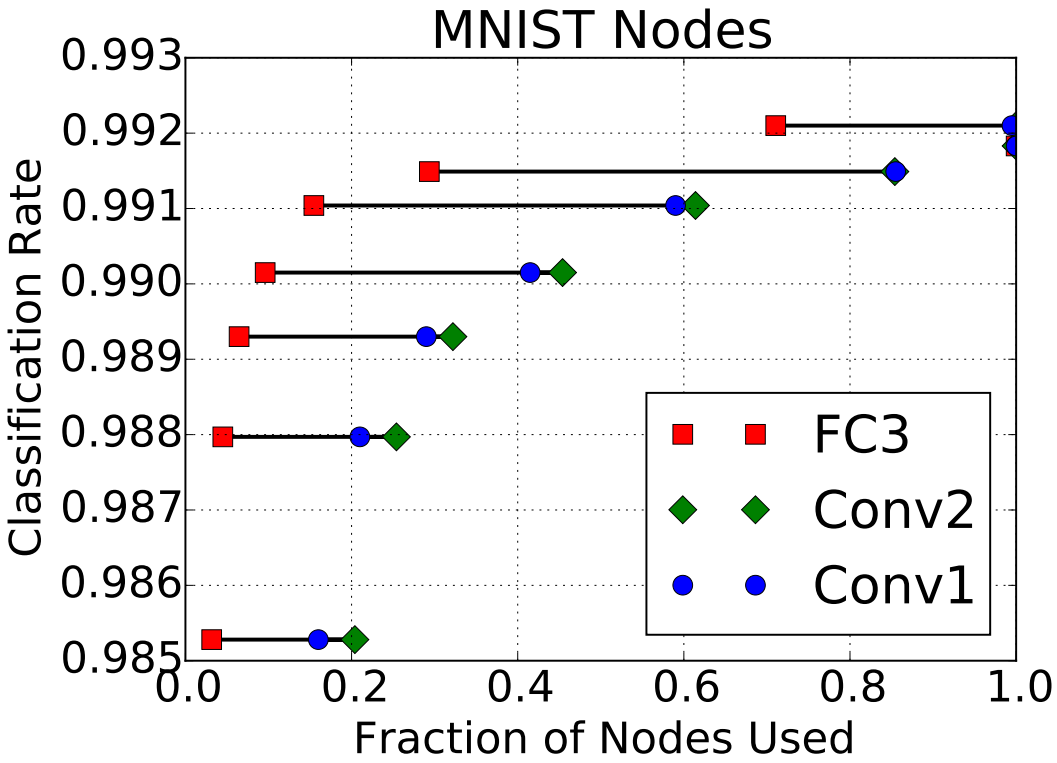


Figure 4.4: Trade-off between the number of nodes in each layer and classification rate on the MNIST dataset. Black lines connect data points collected from the same network. Note that the initial number of nodes for the FC3, Conv2, and Conv1 layers are 500, 50, and 20 respectively.

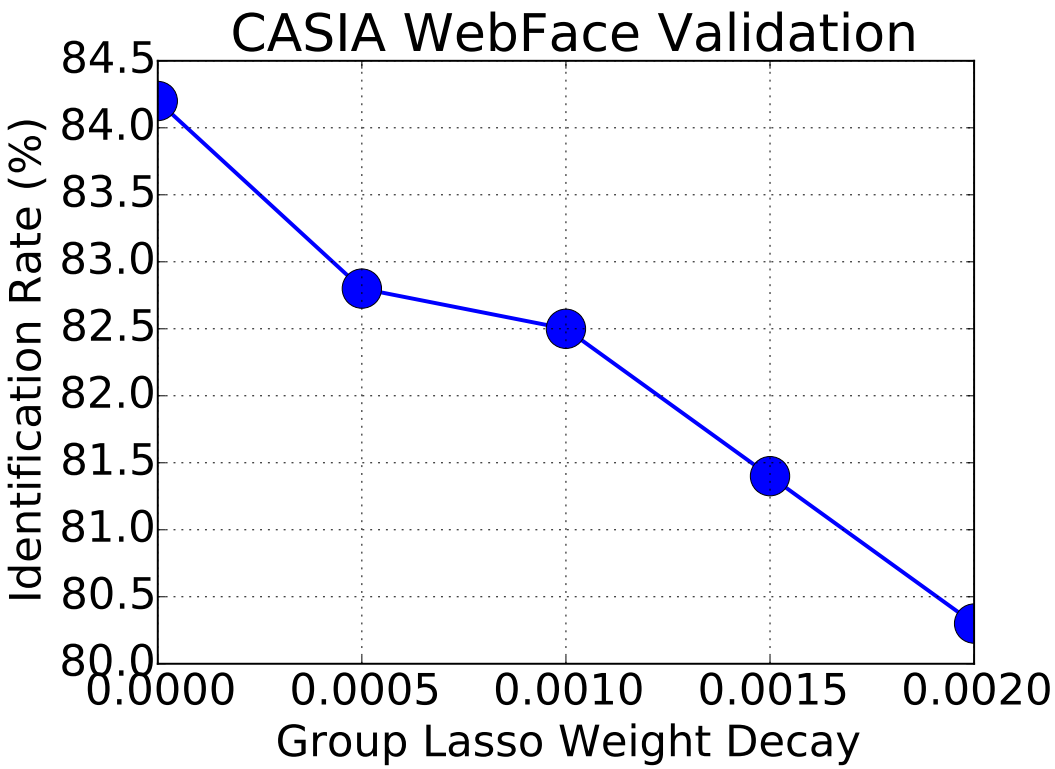


Figure 4.5: Trade-off between the recognition rate of the initial network on the validation set and weight decay on the CASIA WebFace dataset.

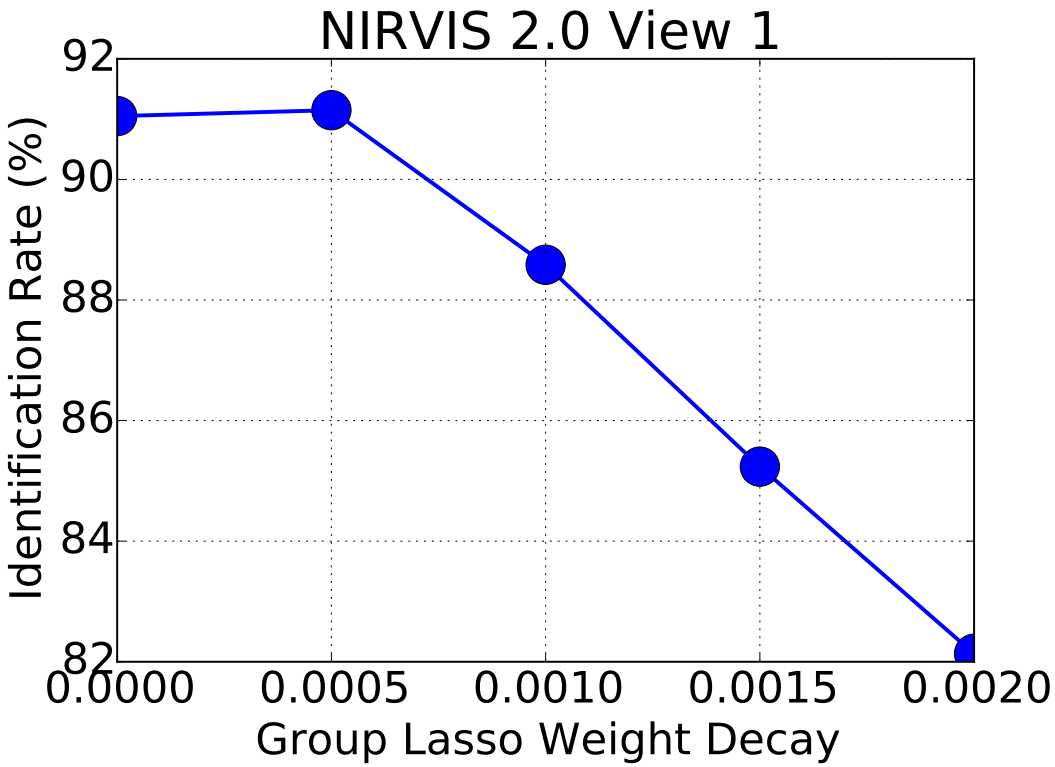


Figure 4.6: Trade-off between the recognition rate and initialization network weight decay on View 1 of the NIR Vis 2.0 dataset.

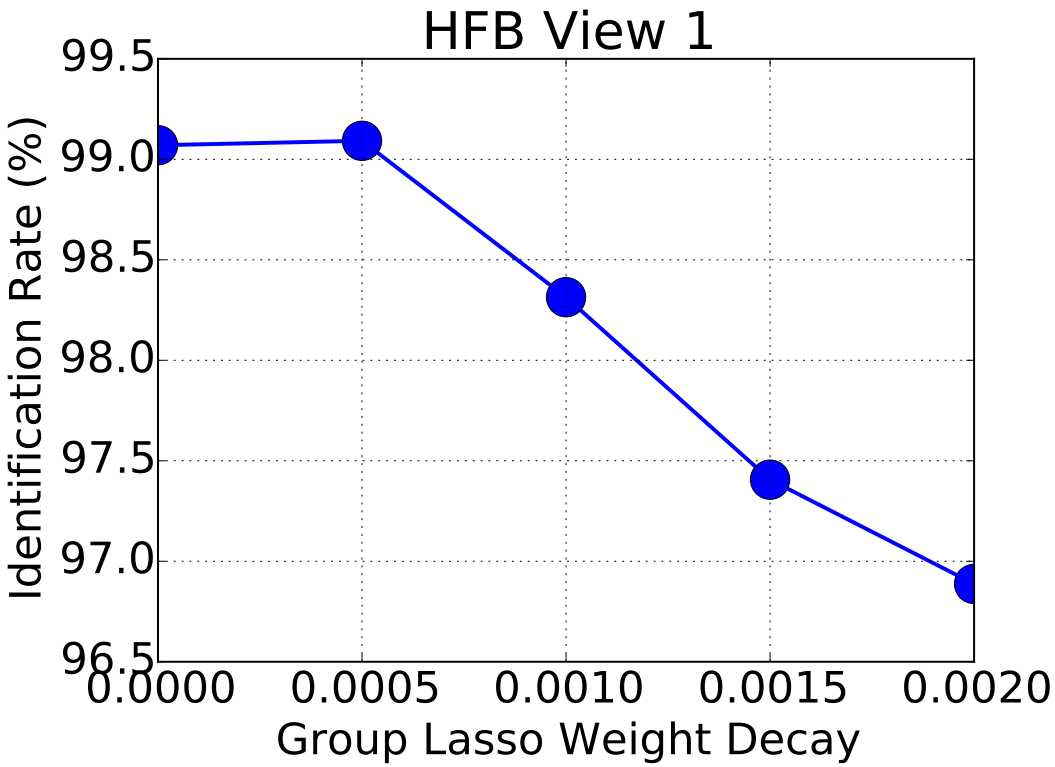


Figure 4.7: Trade-off between the recognition rate and initialization network weight decay on View 1 of the HFB dataset.

Chapter 5: Text Guided Activity Recognition

5.1 Introduction

In recent years, a significant amount of research in the computer vision community has focused on human activity recognition. The objective of this research is to be able to automatically recognize and understand what humans depicted by a video are doing. In this work, human activity recognition is formulated as a classification problem (i.e. given a short video clip, which activity in a given set is depicted). This problem is important for several applications including robotics (helping robots to understand their environment and thus react to it better), surveillance (sift through a large number of video streams to detect abnormal behavior), and video-tagging (automatically tag videos to make them easier to find).

Most state-of-the-art approaches to this problem train deep convolutional neural networks (CNN) to classify videos based on their raw pixels and/or extracted features. One prominent method, the two-stream approach [91], trains one network to classify single frames and a second network to classify short snippets of optical flow features. At test time, the outputs of the two networks are fused by averaging the prediction probabilities. In this work, we build off of the two-stream method.

The main limitation of deep learning approaches to this problem is their de-

pendence on the size and scope of the training data set. It is important to have a large labeled training set to fully take advantage of the power of deep neural networks. In some scenarios, it may not be feasible to attain a large enough data set because it is slow and expensive to collect and label videos. Although state-of-the-art methods achieve good results on benchmark data sets, due to limited data, they cannot use the deepest network architectures (state-of-the-art for many other problems) without overfitting. Furthermore, regardless of the size of the data set, it is virtually impossible to guarantee that all variations of the target activities are captured by the data set.

In this work, we attempt to address these problems with text-guided multitask learning. Text-guided multitask learning helps the neural network training algorithm in two ways. First it allows us to use a large image recognition data set to boost the amount of training data we have. Second, it allows us to incorporate general knowledge about target activities that may not be fully apparent from the training videos.

More specifically, we train our network to simultaneously perform object recognition in addition to activity recognition. This allows us to leverage the ImageNet data set [92], which provides us with significantly more training data. We further improve upon the multitask learning approach by mining a large text corpora to determine the most relevant objects to our activity recognition task. Essentially, we are trying to incorporate general knowledge about the activities from an external source (text corpora) into our training algorithm. Our text-guided multitask learning approach allows us to use much deeper networks than current methods in the

literature without overfitting and thus achieve higher recognition rates.

The remainder of this chapter is organized as follows. In Section 5.2 we detail other relevant published methods. In Section 5.3 we describe our method in more detail. In Section 5.4 we describe experiments we ran to evaluate our method and to compare it to other methods in the literature.

5.2 Related Work

In this section we detail some work related to activity recognition and the approach we take to solve it. This section is by no means exhaustive, but provides a good sampling of other previously published methods.

5.2.1 Activity Recognition

Due to the rise in popularity of deep learning, many recent activity recognition approaches have used deep convolutional neural networks. We base our method off of the two-stream approach [91], which uses two networks trained to classify frames and optical flow features. Temporal Segment Networks [93], which also use the two-stream approach, incorporate longer term temporal information by grouping frames from different portions of the video during training. We incorporate temporal segment networks into our approach as well. Karpathy et al. [94] apply deep learning to a very large dataset. In addition standard convolutional neural networks, three-dimensional convolutional neural networks [95–97] have been used for activity recognition to great effect.

5.2.2 Multitask Learning

Multitask learning was first proposed by Caruana [98]. Multitask learning works by traing a single model to perform multiple related tasks, resulting in improved performance on both tasks. It has been used for a variety of machine learning problems including face recognition [99], tracking [100], and image classification [101]. Since it is such a general approach, it also has been applied to problems outside of computer vision such as in bioinformatics [102].

5.2.3 Imagery/Text Related Work

There has been a significant amount of work done that examines the relationship between images/video and text. Much work has been done on the task of video captioning [103,104]. Ths work attempts to automatically describe the contents of a video with text. Recently, visual question answering [105] has become a very popular problem in the computer vision community. This involves generating text answers to text questions about the content of images. Text grounding [106] in images has also received some attention. The goal of grounding is to localize the portion of an image that represents a text phrase. One commonality of all text related work in the computer vision community is that it uses text related to a specific piece of data (i.e. image, video, image set, etc.). In this chapter, we propose using text to learn general information about the task at hand, and then use that information to improve performance. To the best of our knowledge, we are the first to investigate this approach.

5.3 Our Method

In this section, we describe our approach in more detail. First, we describe the baseline two-stream approach. Next, we describe how we incorporate multi-task learning into that approach. After that, we discuss how we incorporate general knowledge from text to improve the multi-task learning framework. Finally, we describe more specific technical and engineering details about our algorithm.

5.3.1 Two-stream Approach

As implied by its name, the two-stream approach trains two networks to classify a video based on two different data streams: the spatial stream (raw frames) and the temporal stream (optical flow features). The spatial network is trained to classify a single frame, while the temporal network is trained to classify short (10 frames) snippets of optical flow features. The original two-stream approach used the VGG 2048-M network architecture as shown in Figure 5.1, but since then many other network architectures have been used within the same approach to great effect.

At test time, a set of equally spaced frames and optical flow snippets are extracted from a test video and evaluated by the networks to generate prediction probabilities. The probabilities are then fused by a weighted average (generally slightly giving more weight to the temporal network) and the activity is selected by taking the probability with the maximum value.

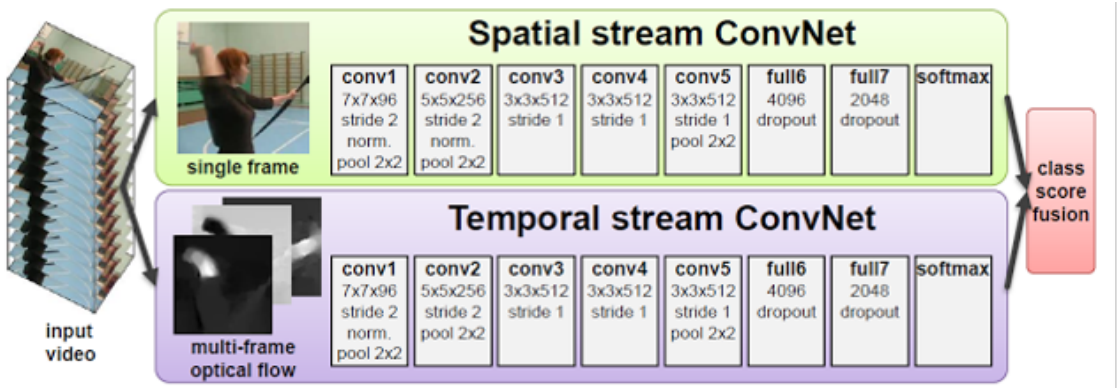


Figure 5.1: Baseline two-stream approach.

5.3.2 Multitask Learning

In order to bolster the amount of training data available to our training algorithm, we use multitask learning when training the spatial network. Specifically, we train it to perform object recognition on ImageNet images in addition to activity recognition on frames. As shown in Figure 5.2, we share all layers of the network between the two tasks aside from the task-specific softmax classifiers.

Multitask learning was originally formulated to learn multiple tasks on the same set of data, but it is common to associate only a subset of the target tasks with portions of the data if they are not fully annotated for each task. In our case, our data sets are only annotated for a single task (videos frames for activity recognition and ImageNet images for object recognition), thus each data sample is only associated with a single loss function. While this aspect may not be in spirit with the original formulation of multitask learning, it still helps to improve activity recognition performance due to the large number of shared layers (all but one) and the relevance of the two tasks to each other.

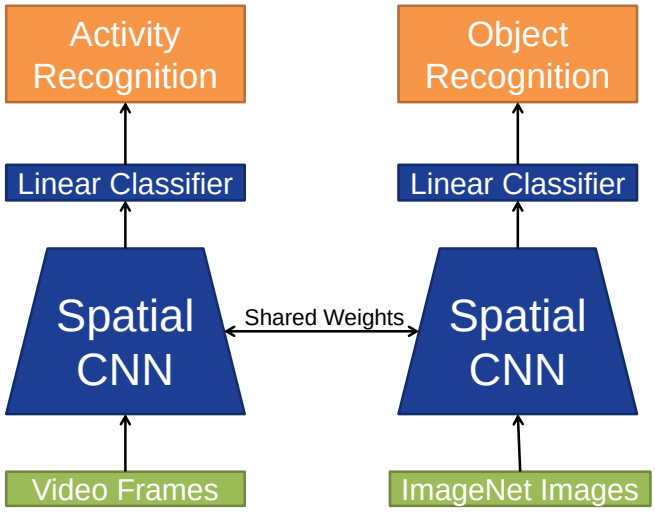


Figure 5.2: Multitask approach.

Another way to think about the multitask learning approach is as an extension of the standard pretraining method. Rather than training a network from scratch (randomly initialized parameter weights), virtually all computer vision deep learning algorithms (including activity recognition) initialize with a network pretrained to perform object recognition on ImageNet images. In our case, when we begin training the network for activity recognition, we also continue the object recognition pretraining. This acts as a regularization for the network parameters, preventing them from overfitting to the activity recognition task.

5.3.3 Incorporation of Text

We further improve upon our multitask learning approach by extracting information from text resources and incorporating it into the training algorithm. Specifically, we use Word2Vec [107] to determine which objects in the ImageNet data set

are most commonly associated with our target activities. We then train the object recognition task to only recognize those relevant objects.

Word2Vec is an algorithm that embeds words and phrases into a vector space based on their usage in a large text corpora. Words that are used in similar contexts will be closer together in the vector space. In this work we use this embedding to determine which ImageNet objects are associated with our target activities.

In order to select the relevant objects, we compute the similarity between the text label of each target activity and each ImageNet class. Let f be a function learned by Word2Vec to map strings into a vector space. We approximate the relevance between a target activity x and an ImageNet class y with the cosine similarity of their vector space representations as follows,

$$s(x, y) = \frac{f(x) \cdot f(y)}{\|f(x)\|_2 \|f(y)\|_2}.$$

We then approximate the overall relevance of an ImageNet class y to the set of target activities X as the sum of the relevances of y to each activity $x \in X$,

$$r(y) = \sum_{x \in X} s(x, y).$$

We do this for all ImageNet classes. We then select the most relevant classes (those whose relevance score is numerically highest) to use in our training algorithm and discard the rest.

5.3.4 Engineering Details

In this work, we use the ResNet family of networks for our network architecture. We run experiments using ResNet 50, ResNet 100, and ResNet 152. This is in contrast to previous approaches which use shallower networks. Our multitask approach acts as a regularization, enabling us to use the deeper, better-performing ResNet networks. All networks are initialized by pretraining on the 1000 ImageNet challenge classes.

When using text to select the most relevant ImageNet classes to recognize, one must determine how many of them to select. There is no clear answer to this question, so we err on the side of caution and use the 1000 most relevant classes. We also experiment with using fewer (500) and more (2000) and found there was little change to the recognition rate.

We train our networks with stochastic gradient descent on single GPU (NVIDIA TITAN X or NVIDIA 1080) systems. Due to the depths of the networks used and the memory limitations of the GPUs (12 GB and 8 GB), we were forced to use small batch sizes of 64, 48, and 32 frames/images for ResNet 50, ResNet 101, and ResNet 152 respectively. When training in the multitask setting, we split the batch size between activity recognition frames and ImageNet images. We found that splitting the batch approximately evenly between the two (i.e. giving equal weight to the two objectives) provided the best activity recognition performance. And, of course, when training the baseline approach (without multitask learning), each batch consists entirely of activity recognition frames.

We incorporate the Temporal Segment Network (TSN) of [93] training method when training our networks. This introduces some long-term temporal information about the videos, and thus improves the performance. One side effect of this method is that the size the activity recognition frames portion of the batch must be a multiple of the selected number of segments. Through experimentation, we determine the optimal number of segments to be three, and thus the size of the activity recognition portion of the batch must be a multiple of three. So for example, when training the ResNet 50 network, our total batch size is 64. Ideally we would split it evenly to have 32 activity recognition samples and 32 ImageNet samples, but 32 is not a multiple of three. Instead we use 33 activity recognition samples and 31 ImageNet samples.

When training ResNet 50, we initialize the learning rate to .001. We divide it by 10 after 10,000 and 13,000 iterations and train for 15,000 iterations in total. Due to the smaller batch sizes, we initialize the learning rates for the ResNet 101 and 152 network architectures to .0005. For ResNet 101, we divide it by 10 after 13,000 and 18,000 iterations and train for 20,000 iterations in total. For ResNet 152, we divide it by 10 after 28,000 and 36,000 iterations and train for 40,000 iterations in total. For all experiments, we set the weight decay to be .0001. During training of all three architectures, we place dropout layers just before the final softmax classifiers. We set the dropout rate to .25 (i.e. keep 75% of features).

We only perform two preprocessing on video frames. First we subtract a mean pixel from each pixel in the image. Then we select a random window from the target frame. The window's width and height are randomly and independently selected

(from a uniform distribution) to be between 168 and 256 pixels. Once the width and height are selected, the location of the window within the image is selected at random (again, from a uniform distribution). Finally, the window is resized to 224×224 pixels and fed into the network. The random window selection process helps to generate more variation in the training data to reduce the risk of overfitting. We perform slightly different preprocessing to the ImageNet images. We still subtract the pixel mean, but select a sub-image by simply choosing a random 224×224 window from the image. We can use a simpler window selection with ImageNet because it contains many more images which are uncorrelated (unlike video frames which are highly correlated). At test time, we use the standard approach of generating predictions for 25 evenly spaced frames. For each frame, we generate predictions from 10 different 224×224 pixel windows: one from each corner of the frame, one from the center of the frame, and then a horizontally flipped version of each of those. For each video we end up with 250 probability predictions for each of the classes. We average them and predict the activity with the highest value.

In this work, we use a pre-trained Word2Vec model. It was trained on an internal Google data set of news articles containing a billion words [107].

5.4 Experiments

We evaluate the performance of our approach on the UCF 101 dataset [108]. It contains a variety of activities including playing various sports, playing musical instruments, and many common indoor and outdoor tasks.

Network	Baseline	Multitask	Multitask + Text
ResNet 50	81.3	84.0	85.1
ResNet 101	82.6	85.3	86.9
ResNet 152	83.1	86.0	**87.5**
TSN [93]	85.7	-	-

Table 5.1: Performance of our algorithm on the UCF 101 data set.

We run experiments on three different ResNet networks (ResNet 50, ResNet 101, and ResNet 152) under three different settings (baseline, multitask, multitask with text). The baseline approach is the standard method without multitask learning. The multitask approach randomly selects 1000 ImageNet categories to learn to recognize for the secondary objective. The multitask with text approach uses Word2Vec to select the 1000 most relevant ImageNet categories as described in Section 5.3.3.

The prefusion results for the spatial stream are shown in Table 5.1. From the results, it is clear that using the ResNet networks with the baseline approach provides worse performance than the state-of-the-art methods. This is because the state-of-the-art methods use shallower networks which are not as prone to overfitting. When we incorporate multitask learning, the performance increases to about the current state-of-the-art. And finally, when we also incorporate text-guided supervision, we are able to outperform the state-of-the-art by a few percent.

5.5 Conclusion and Future Work

In this chapter we have shown how to leverage information extracted from text resources to improve the performance of deep convolutional neural network based activity recognition classifiers. We do this by using Word2Vec to identify objects most commonly associated with the target activities and then training the network to recognize those objects in addition to classifying the frames of activity videos.

Chapter 6: Summary and Future Work

6.1 Summary

While much of computer vision research focuses exclusively on visible light imagery, there are many problems for which alternative data modalities can be used either in place of or as a supplement to visible images and videos. In this dissertation we presented several algorithms for scenarios for which this is the case. Specifically, we focused on heterogeneous face recognition and activity recognition.

In Chapter 2, we presented a method to represent data across multiple domains for thermal infrared HFR. Specifically, we extended a linear coupled dictionary model to use the kernel method. Additionaly, we altered the training algorithm to incorporate task-driven supervision. The kernel task-driven coupled dictionaries generated by these two improvements outperformed algorithms based on standard coupled dictionaries on thermal infrared to visible face recognition.

In Chapter 3, we presented a deep learning-based approach to NIR HFR. This deep convolutional neural network-based approach modeled faces as a whole rather than a collection of patches. We showed how to leverage a large visible image face dataset to prevent overfitting of the model and presented experimental results on two benchmark data sets showing its effectiveness.

In Chapter 4, we presented a method to automatically determine the number of nodes in each layer of a neural network. We did this by replacing the standard weight decay with a group lasso penalty. We discussed some engineering details of using this method and applied it to deep networks for NIR HFR as well as other computer vision problems.

In Chapter 5, we presented a method to improve activity recognition algorithms through the use of multitask learning and information extracted from a large text corpora. We first used multitask learning to train neural networks to recognize objects in addition to activities. We then used information about the target activities from a large text corpora to determine the best object classes for the secondary task to recognize.

6.2 Future Work

6.2.1 Deep Heterogeneous Face Recognition

To this point, our work on deep heterogeneous face recognition has only used discriminating feed forward neural networks. Recently, a significant amount of research in the machine learning community has focused on learning generative models with generative adversarial networks (GAN) [109]. In the future, we plan to extend our approach by exploring the use of deep generative models. This could allow us to, at the very least, generate artificial training data to improve our algorithm's performance. It could potentially even allow us to synthesize faces across domains.

6.2.2 Text Guided Multitask Learning

Going forward, we will attempt to improve upon our text guided activity recognition method in several ways. The most obvious way is to apply our method to the temporal stream in addition to the spatial stream. It cannot be directly applied as is because no publicly available data sets exist for optical flow features of objects. While it cannot be directly applied, there are reasons to believe it could be applied with some alterations. Previous work [110] has shown that fusing the two-streams part of the way through the networks can increase the activity recognition performance. In that case, our multitask learning approach could be used to regularize the fused network instead of just the spatial network. Additionally, it has been shown that initializing the temporal network with the spatial network can increase the temporal recognition rate [93]. This suggests that there are some similarities between the two domains, and, since multitask learning helps the spatial domain, it may also help the temporal domain.

In addition to applying our method to the temporal stream, in the future we will also attempt to incorporate other tasks in the multitask learning framework. For example, human detection would almost certainly help in recognizing human activities. Another task that could be beneficial is semantic segmentation. Finally, we could simply incorporate audio data into the recognition algorithm. This could be a third stream or perhaps be fused with the other two streams. Finally, we will also try to apply this approach to solve other problems. While we have shown some success in applying this method to activity recognition, it may also help improve

deep learning-based algorithms for other computer vision problems such as object localization.

Bibliography

[1] Farid Melgani and Lorenzo Bruzzone. Classification of hyperspectral remote sensing images with support vector machines. *IEEE Transactions on geoscience and remote sensing*, 42(8):1778–1790, 2004.

[2] Jianchao Yang, Zhaowen Wang, Zhe Lin, Xianbiao Shu, and Thomas Huang. Bilevel sparse coding for coupled feature spaces. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2360–2367. IEEE, 2012.

[3] Jianchao Yang, Zhaowen Wang, Zhe Lin, Scott Cohen, and Thomas Huang. Coupled dictionary training for image super-resolution. *Image Processing, IEEE Transactions on*, 21(8):3467–3478, 2012.

[4] Bernhard Scholkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[5] C Reale, N.M. Nasrabadi, and R. Chellappa. Coupled dictionaries for thermal to visible face recognition. In *Image Processing (ICIP), 2014 21st IEEE International Conference on*, Oct 2014.

[6] Julien Mairal, Francis Bach, and Jean Ponce. Task-driven dictionary learning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(4):791–804, 2012.

[7] Jianchao Yang, John Wright, Thomas S Huang, and Yi Ma. Image super-resolution via sparse representation. *Image Processing, IEEE Transactions on*, 19(11):2861–2873, 2010.

[8] Shenlong Wang, Lei Zhang, Yan Liang, and Quan Pan. Semi-coupled dictionary learning with applications to image super-resolution and photo-sketch synthesis. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2216–2223. IEEE, 2012.

[9] Jingjing Zheng, Zhuolin Jiang, P Jonathon Phillips, and Rama Chellappa. Cross-view action recognition via a transferable dictionary pair. In *BMVC*, pages 1–11, 2012.

[10] Kui Jia, Xiaogang Wang, and Xiaoou Tang. Image transformation based on learning dictionaries across image spaces. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(2):367–380, 2013.

[11] Li He, Hairong Qi, and R. Zaretzki. Beta process joint dictionary learning for coupled feature spaces with application to single image super-resolution. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 345–352, June 2013.

[12] Qiang Zhang and Baoxin Li. Discriminative k-SVD for dictionary learning in face recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2691–2698. IEEE, 2010.

[13] Shenghua Gao, Ivor Wai-Hung Tsang, and Liang-Tien Chia. Kernel sparse representation for image classification and face recognition. In *Computer Vision–ECCV 2010*, pages 1–14. Springer, 2010.

[14] Shenghua Gao, I.W. Tsang, and Liang-Tien Chia. Sparse representation with kernels. *Image Processing, IEEE Transactions on*, 22(2):423–434, Feb 2013.

[15] HV Nguyen, Vishal M Patel, Nasser M Nasrabadi, and Rama Chellappa. Kernel dictionary learning. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 2021–2024. IEEE, 2012.

[16] A. Shrivastava, V.M. Patel, and R. Chellappa. Multiple kernel learning for sparse representation-based classification. *Image Processing, IEEE Transactions on*, 23(7):3013–3024, July 2014.

[17] Zhuolin Jiang, Zhe Lin, and Larry S Davis. Label consistent k-SVD: learning a discriminative dictionary for recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(11):2651–2664, 2013.

[18] Zhaowen Wang, Jianchao Yang, Nasser Nasrabadi, and Thomas Huang. A max-margin perspective on sparse representation-based classification. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1217–1224. IEEE, 2013.

[19] John Wright, Allen Y Yang, Arvind Ganesh, Shankar S Sastry, and Yi Ma. Robust face recognition via sparse representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(2):210–227, 2009.

[20] Meng Yang, Lei Zhang, Simon C.K. Shiu, and David Zhang. Gabor feature based robust representation and classification for face recognition with gabor occlusion dictionary. *Pattern Recognition*, 46(7):1865 – 1878, 2013.

[21] Can-Yi Lu, Hai Min, Jie Gui, Lin Zhu, and Ying-Ke Lei. Face recognition via weighted sparse representation. *Journal of Visual Communication and Image Representation*, 24(2):111 – 116, 2013. Sparse Representations for Image and Video Analysis.

[22] Ognjen Arandjelovi and Roberto Cipolla. Achieving robust face recognition from video by combining a weak photometric model and a learnt generic face invariant. *Pattern Recognition*, 46(1):9 – 23, 2013.

[23] J. Lu, Y. P. Tan, and G. Wang. Discriminative multimanifold analysis for face recognition from a single training sample per person. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):39–51, Jan 2013.

[24] Y. Taigman, Ming Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1701–1708, June 2014.

[25] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Web-scale training for face identification. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 2746–2754, June 2015.

[26] Yi Sun, Ding Liang, Xiaogang Wang, and Xiaoou Tang. Deepid3: Face recognition with very deep neural networks. *arXiv preprint arXiv:1502.00873*, 2015.

[27] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 815–823, June 2015.

[28] Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. *Proceedings of the British Machine Vision*, 2015.

[29] Jiayi Ma, Ji Zhao, Yong Ma, and Jinwen Tian. Non-rigid visible and infrared face registration via regularized gaussian fields criterion. *Pattern Recognition*, 48(3):772 – 784, 2015.

[30] Gabriel Hermosilla, Javier Ruiz del Solar, Rodrigo Verschae, and Mauricio Correa. A comparative study of thermal face recognition methods in unconstrained environments. *Pattern Recognition*, 45(7):2445 – 2459, 2012.

[31] R. S. Ghiass, O. Arandjelovi, H. Bendada, and X. Maldague. Illumination-invariant face recognition from a single image across extreme pose using a dual dimension aam ensemble in the thermal infrared spectrum. In *The 2013 International Joint Conference on Neural Networks (IJCNN)*, pages 1–10, Aug 2013.

[32] Gabriel Hermosilla, Javier Ruiz-del Solar, and Rodrigo Verschae. An enhanced representation of thermal faces for improving local appearance-based face recognition. *Intelligent Automation & Soft Computing*, pages 1–12, 2015.

[33] R.S. Ghiass, O. Arandjelovic, H. Bendada, and X. Maldague. Infrared face recognition: A literature review. In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–10, Aug 2013.

[34] Reza Shoja Ghiass, Ognjen Arandjelovi, Abdelhakim Bendada, and Xavier Maldague. Infrared face recognition: A comprehensive review of methodologies and databases. *Pattern Recognition*, 47(9):2807 – 2824, 2014.

[35] Wenyi Zhao, Rama Chellappa, P Jonathon Phillips, and Azriel Rosenfeld. Face recognition: A literature survey. *Acm Computing Surveys (CSUR)*, 35(4):399–458, 2003.

[36] Thirimachos Bourlai, Arun Ross, Cunjian Chen, and Lawrence Hornak. A study on using mid-wave infrared images for face recognition. In *SPIE Defense, Security, and Sensing*, pages 83711K–83711K. International Society for Optics and Photonics, 2012.

[37] Jonghyun Choi, Shuowen Hu, S. Susan Young, and Larry S. Davis. Thermal to visible face recognition. volume 8371, pages 83711L–83711L–10, 2012.

[38] Shuowen Hu, Jonghyun Choi, Alex L. Chan, and William R. Schwartz. Thermal to visible face recognition using partial least squares. *Journal of the Optical Society of America - A*, Under Review.

[39] Brendan F. Klare and Anil K. Jain. Heterogeneous face recognition using kernel prototype similarities. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(6):1410–1422, 2013.

[40] Wei Zhang, Xiaogang Wang, and Xiaoou Tang. Coupled information-theoretic encoding for face photo-sketch recognition. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 513–520. IEEE, 2011.

[41] Meina Kan, Shiguang Shan, Haihong Zhang, Shihong Lao, and Xilin Chen. Multi-view discriminant analysis. In *Computer Vision–ECCV 2012*, pages 808–821. Springer, 2012.

[42] Xiaogang Wang and Xiaoou Tang. Face photo-sketch synthesis and recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(11):1955–1967, 2009.

[43] Zhifeng Li, Dihong Gong, Yu Qiao, and Dacheng Tao. Common feature discriminant analysis for matching infrared face images to optical face images. *Image Processing, IEEE Transactions on*, 23(6):2436–2445, June 2014.

[44] Zhen Lei, Shengcai Liao, A.K. Jain, and S.Z. Li. Coupled discriminant analysis for heterogeneous face recognition. *Information Forensics and Security, IEEE Transactions on*, 7(6):1707–1716, Dec 2012.

[45] Christopher Reale, Nasser M. Nasrabadi, Heesung Kwon, and Rama Chellappa. Seeing the forest from the trees: A holistic approach to near-infrared heterogeneous face recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2016.

[46] Jonghyun Choi, Abhishek Sharma, David W Jacobs, and Larry S Davis. Data insufficiency in sketch versus photo face recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 1–8. IEEE, 2012.

[47] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[48] Vishal M Patel, Tao Wu, Soma Biswas, P Jonathon Phillips, and Rama Chellappa. Dictionary-based face recognition under variable lighting and pose. *Information Forensics and Security, IEEE Transactions on*, 7(3):954–965, 2012.

[49] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[50] Xin Chen, Patrick J. Flynn, and Kevin W. Bowyer. Visible-light and infrared face recognition. In *Proceedings of ACM Workshop on Multimodal User Authentication*, pages 48–55, 2003.

[51] Patrick J Flynn, Kevin W Bowyer, and P Jonathon Phillips. Assessment of time dependency in face recognition: An initial study. In *Audio-and Video-Based Biometric Person Authentication*, pages 44–51. Springer, 2003.

[52] Kenneth A. Byrd. Preview of the newly acquired nvesd-arl multimodal face database. In *Proceedings of the SPIE DSS*, volume 8734 of *ICML '04*, 2013.

[53] Francis R. Bach, Gert R. G. Lanckriet, and Michael I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 6–, New York, NY, USA, 2004. ACM.

[54] Shuxin OuYang, Timothy M. Hospedales, Yi-Zhe Song, and Xueming Li. A survey on heterogeneous face recognition: Sketch, infra-red, 3d and low-resolution. *CoRR*, abs/1409.5114, 2014.

[55] B. Klare and A.K. Jain. Heterogeneous face recognition: Matching nir to visible light images. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 1513–1516, Aug 2010.

[56] Jun-Yong Zhu, Wei-Shi Zheng, Jian-Huang Lai, and S.Z. Li. Matching nir face to vis face using transduction. *Information Forensics and Security, IEEE Transactions on*, 9(3):501–514, March 2014.

[57] Felix Juefei-Xu, Dipan Pal, and Marios Savvides. Nir-vis heterogeneous face recognition via cross-spectral joint dictionary learning and reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 141–150, 2015.

[58] Dong Yi, Zhen Lei, and S.Z. Li. Shared representation learning for heterogenous face recognition. In *Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on*, volume 1, pages 1–7, May 2015.

[59] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2169–2178. IEEE, 2006.

[60] G Hinton, S Osindero, and Y Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, July 2006.

[61] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *The Journal of Machine Learning Research*, 11:625–660, 2010.

[62] Yi Jin, Jiwen Lu, and Qiuqi Ruan. Coupled discriminative feature learning for heterogeneous face recognition. *Information Forensics and Security, IEEE Transactions on*, 10(3):640–652, 2015.

[63] Yi Jin, Jiwen Lu, and Qiuqi Ruan. Large margin coupled feature learning for cross-modal face recognition. In *Biometrics (ICB), 2015 International Conference on*, pages 286–292, May 2015.

[64] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman. Fisher Vector Faces in the Wild. In *British Machine Vision Conference*, 2013.

[65] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *Advances in Neural Information Processing Systems*, pages 1988–1996, 2014.

[66] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation from predicting 10,000 classes. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1891–1898, June 2014.

[67] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deeply learned face representations are sparse, selective, and robust. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 2892–2900, June 2015.

[68] Dong Chen, Xudong Cao, Liwei Wang, Fang Wen, and Jian Sun. Bayesian face revisited: A joint formulation. In *Computer Vision–ECCV 2012*, pages 566–579. Springer, 2012.

[69] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[70] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[71] S. Gupta, J. Hoffman, and J. Malik. Cross modal distillation for supervision transfer. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2827–2836, June 2016.

[72] Jong-Chyi Su and Subhransu Maji. Cross quality distillation. *arXiv preprint arXiv:1604.00433*, 2016.

[73] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.

[74] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014.

[75] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 539–546. IEEE, 2005.

[76] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, volume 2, pages 1735–1742. IEEE, 2006.

[77] Vahdat Kazemi and Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1867–1874. IEEE, 2014.

[78] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[79] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[80] Jiwen Lu, V.E. Liong, Xiuzhuang Zhou, and Jie Zhou. Learning compact binary face descriptor for face recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 37(10):2041–2056, Oct 2015.

[81] Zhen Lei, M. Pietikainen, and S.Z. Li. Learning discriminant face descriptor. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(2):289–302, Feb 2014.

[82] Stan Z Li, Dong Yi, Zhen Lei, and Shengcai Liao. The casia nir-vis 2.0 face database. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*, pages 348–353. IEEE, 2013.

[83] Stan Z Li, Zhen Lei, and Meng Ao. The HFB face database for heterogeneous face biometrics research. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 1–8. IEEE, 2009.

[84] Lukas Meier, Sara Van De Geer, and Peter Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1):53–71, 2008.

[85] Y Le Cun, B Boser, John S Denker, D Henderson, Richard E Howard, W Hubbard, and Lawrence D Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems*. Citeseer, 1990.

[86] Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, and Marianna Pensky. Sparse convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 806–814, 2015.

[87] Zichao Yang, Marcin Moczulski, Misha Denil, Nando de Freitas, Alex Smola, Le Song, and Ziyu Wang. Deep fried convnets. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1476–1483, 2015.

[88] Marcin Moczulski, Misha Denil, Jeremy Appleyard, and Nando de Freitas. Acdc: A structured efficient linear layer. *arXiv preprint arXiv:1511.05946*, 2015.

[89] Vadim Lebedev and Victor Lempitsky. Fast convnets using group-wise brain damage. *arXiv preprint arXiv:1506.02515*, 2015.

[90] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[91] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[92] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[93] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision*, pages 20–36. Springer, 2016.

[94] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[95] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015.

[96] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2013.

[97] Lin Sun, Kui Jia, Dit-Yan Yeung, and Bertram E Shi. Human action recognition using factorized spatio-temporal convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4597–4605, 2015.

[98] Rich Caruana. Multitask learning. In *Learning to learn*, pages 95–133. Springer, 1998.

[99] Rajeev Ranjan, Swami Sankaranarayanan, Carlos D Castillo, and Rama Chellappa. An all-in-one convolutional neural network for face analysis. In *Automatic Face & Gesture Recognition (FG 2017), 2017 12th IEEE International Conference on*, pages 17–24. IEEE, 2017.

[100] Tianzhu Zhang, Bernard Ghanem, Si Liu, and Narendra Ahuja. Robust visual tracking via multi-task sparse learning. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2042–2049. IEEE, 2012.

[101] Xiao-Tong Yuan, Xiaobai Liu, and Shuicheng Yan. Visual classification with multitask joint sparse representation. *IEEE Transactions on Image Processing*, 21(10):4349–4360, 2012.

[102] Qian Xu and Qiang Yang. A survey of transfer and multitask learning in bioinformatics. *Journal of Computing Science and Engineering*, 5(3):257–268, 2011.

[103] Haonan Yu, Jiang Wang, Zhiheng Huang, Yi Yang, and Wei Xu. Video paragraph captioning using hierarchical recurrent neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4584–4593, 2016.

[104] Pingbo Pan, Zhongwen Xu, Yi Yang, Fei Wu, and Yueting Zhuang. Hierarchical recurrent neural encoder for video representation with application to captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1029–1038, 2016.

[105] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2425–2433, 2015.

[106] Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. Grounding of textual phrases in images by reconstruction. In *European Conference on Computer Vision*, pages 817–834. Springer, 2016.

[107] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[108] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[109] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[110] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1933–1941, 2016.