

**BOOSTED PROJECTIONS AND LOW COST
TRANSFER LEARNING APPLIED TO SMART
SURVEILLANCE**

RICARDO BARBOSA KLOSS

**BOOSTED PROJECTIONS AND LOW COST
TRANSFER LEARNING APPLIED TO SMART
SURVEILLANCE**

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

ORIENTADOR: WILLIAM ROBSON SCHWARTZ

Belo Horizonte
Fevereiro de 2018

RICARDO BARBOSA KLOSS

**BOOSTED PROJECTIONS AND LOW COST
TRANSFER LEARNING APPLIED TO SMART
SURVEILLANCE**

Dissertation presented to the Graduate Program in Ciência da Computação of the Universidade Federal de Minas Gerais in partial fulfillment of the requirements for the degree of Master in Ciência da Computação.

ADVISOR: WILLIAM ROBSON SCHWARTZ

Belo Horizonte

February 2018

© 2018, Ricardo Barbosa Kloss.
Todos os direitos reservados.

Kloss, Ricardo Barbosa

K66b Boosted Projections and Low Cost Transfer Learning
Applied to Smart Surveillance / Ricardo Barbosa Kloss.
— Belo Horizonte, 2018
xxii, 62 f. : il. ; 29cm

Dissertação (mestrado) — Universidade Federal de
Minas Gerais

Orientador: William Robson Schwartz

1. Computação – Teses. 2. Visão por Computador.
3. Aprendizado do computador. 4. Teoria da estimativa.
I. Orientador. II. Título.

CDU 519.6*82.10(043)



UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

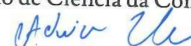
FOLHA DE APROVAÇÃO


Boosted projections and low cost transfer learning applied to smart surveillance

RICARDO BARBOSA KLOSS

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:


PROF. WILLIAM ROBSON SCHWARTZ - Orientador
Departamento de Ciência da Computação - UFMG


PROF. ADRIANO ALONSO VELOSO
Departamento de Ciência da Computação - UFMG


PROF. SILVIO JAMIL FERZOLI GUIMARÃES
Instituto de Ciências Exatas e Informática - PUC/MG

Belo Horizonte, 27 de Fevereiro de 2018.

Acknowledgments

I am most grateful for my parents and their help with which I would probably not even be able to have accomplished my bachelor degree, and whose care, motivation and understanding contributed greatly to the person I am today, be that good or bad, my curiosity and scientific hunger is only because of them.

I thank my advisor, professor William Robson Schwartz, which through patience, interesting discussions and ideas and constructive criticism helped me get here. Your dedication and care for your students is something rare and I am very grateful for having the chance to take part in it.

I thank my friends, all of them, namely, Artur Jordão, my loved Camilla Savicius, Cássio Elias dos Santos, Jessica Sena, Leandro Elias Canaan Mageste, Luiz Eduardo Mendes Matheus, Ricardo Boccoli Gallego, Rodrigo Kloss, Tales Rodrigues de Oliveira, Victor Hugo, Yago Guimarães. We are all and foremost made of memories and experiences, and I am proud to have chosen you to share and enjoy life.

I would also like to thank the Coordination for the Brazilian National Research Council – CNPq (Grant #311053/2016-5), the Minas Gerais Research Foundation – FAPEMIG (Grants APQ-00567-14, RED-00042-16, PPM-00540-17) and the Coordination for the Improvement of Higher Education Personnel – CAPES (DeepEyes Project).

*“ The road to wisdom?
-Well, it’s plain and simple to express:
Err and err and err again,
but less and less and less.”*
(Piet Hein)

Abstract

Computer vision is an important area related to understanding the world through images. It can be used as biometry, by verifying if a given face is of a certain identity, used to look for crime perpetrators in an airport blacklist, used in human-machine interactions and other goals. Since 2012, deep learning methods have become ubiquitous in computer vision achieving breakthroughs, and making possible for machines, for instance, to perform face verification with human-level skill. This work tackles two computer vision problems and is divided in two parts. In one we explore deep learning methods in the task of face verification and in the other the task of dimensionality reduction. Both tasks have large importance in the fields of machine learning and computer vision. We focus on their application in smart surveillance. Dimensionality reduction helps alleviate problems which usually suffer from a very high dimensionality, which can make it hard to learn classifiers. This work presents a novel method for tackling this problem, referred to as Boosted Projection. It relies on the use of several projection models based on Principal Component Analysis or Partial Least Squares to build a more compact and richer data representation. Our experimental results demonstrate that the proposed approach outperforms many baselines and provides better results when compared to the original dimensionality reduction techniques of partial least squares. In the second part of this work, regarding face verification, we explored a simple and cheap technique to extract deep features and reuse a pre-learned model. The technique is a transfer learning that involves no fine-tuning of the model to the new domain. Namely, we explore the correlation of depth and scale in deep models, and look for the layer/scale that yields the best results for the new domain, we also explore metrics for the verification task, using locally connected convolutions to learn distance metrics. Our face verification experiments use a model pre-trained in face identification and adapt it to the face verification task with different data, but still on the face domain. We achieve 96.65% mean accuracy on the Labeled Faces in the Wild dataset and 93.12% mean accuracy on the Youtube Faces dataset which are in the state-of-the-art.

Keywords: Computer Vision, Machine Learning, Smart Surveillance, Deep Learning, Dimensionality Reduction.

List of Figures

2.1	Illustration of a standard image classification pipeline exemplified with classes of images of dogs and cats.	7
2.2	Principal components shown in green and cyan. It illustrates that the components are roughly in the directions of maximum variance.	11
3.1	General structure of our proposed framework for improved projections. The node of a layer learns a projection using the hard samples found by the previous node, this continues until convergence. Then, it is possible to create a new layer, where the input features are the concatenation of the projections of each node in the previous layer. The number of nodes and layers are hyperparameters.	15
3.2	Detail of the Node of the Boosted Projection.	16
3.3	Explanation of the specialization of the nodes in the Boosted Projection.	16
4.1	Comparison of our boosted PLS projection with other approaches found in the literature. Results using the log-average miss-rate (lower values are better) of 10^{-2} to 10^0 (standard protocol).	21
4.2	Another comparison of our boosted PLS projection with other approaches found in the literature. Results using the area from 10^{-2} to 10^{-1}	22
6.1	The Original Perceptron with sigmoid activation. In matrix notation: $\hat{y} = \sigma(X \cdot \vec{w})$, where $\sigma(x) = \frac{1}{e^{-x}+1}$	29
6.2	The Multilayer Perceptron. Each circle is a Perceptron and each Level (Inputs, Hidden, Outputs) is called a Layer.	30
6.3	Boundary space of a Sigmoid Perceptron.	31

6.4	Combining the space partitioning of multiple linear sigmoid perceptrons, the multilayer perceptron is able to solve the XOR non-linear problem. Adapted from http://playground.tensorflow.org/ . The input consists of a 'x' and 'y' coordinate, and it is fed to each of five neurons in a sequential hidden layer where each of these neurons have learned a linear space separation that when combined leads to the non-linear boundary space of the output depicted in the far right cartesian space.	32
6.5	Illustration of Convolutional Layers. Taken from Stanford's CS231 Course	32
6.6	Example of the computation of a convolutional operation. (a) illustrated a convoution with a unit stride and (b) with a stride of 2, resulting in a subsampling operation. (c) is the applied convolutional filter. The output of the convolution operations are in the top row, depicted in yellow color. .	33
6.7	Fully Connected network that is equivalent to the convolutional one depicted in Figure 6.6(b). The weights outside the receptive field are zeroed as is seen in (b). The output of the convolution operations are in the top row, depicted in yellow color.	33
6.8	Locally Connected Convolution Compared with Regular Convolution. The input are the bottommost, gray, boxes. The output of the convolution operations are in the top row, depicted in yellow color. The filters are depicted in the middle between the input and the output. Note that the locally connected filters are different on each spatial position, depicted by different colors, green, red and blue.	34
6.9	Here we show a parallel between modern convolutional networks and what was previously done in computer vision to build mid-level features, e.g. bag of visual words.	35
6.10	AlexNet architecture. Taken from their original work paper [Krizhevsky et al., 2012].	36
6.11	Original Inception Module. The 1×1 convolution in the beginning of the module is a compression tricky in order to fit the model in the memory constraints of GPUs. This module explores patterns of different scales by using different sizes for the receptive fields of the convolutions The image is from the original paper [Szegedy et al., 2015].	36
6.12	Visualization of filters for the Two initial layers, first row, and last two layers of the model from [Zeiler and Fergus, 2014]. It is possible to see edge patterns (layer 1) that together compose texture patterns (layer 2), and by the final layers objects and parts of objects. Adapted from the original work.	37
6.13	Resnet module as seen in [He et al., 2016].	38

6.14	In this figure, taken from [Hu et al., 2014] it is illustrated the goal of Mahalanobis distance metric learning. It is desirable that not-same pairs that are close in the conventional euclidean space become distant in the transformed space and same pairs to be close.	40
6.15	Figure adapted from [Sun et al., 2013], illustrates their network architecture for face verification. The ensemble architecture is depicted in the top row and the individual ConvNet of the ensemble depicted in the bottom row. . .	41
6.16	Depiction of the Siamese Network. The boxes labeled <i>Feature Extraction Layers</i> share weights and are responsible for extracting features from each sample which are compared in the block labeled <i>Verify</i> . The features are optimized to separate pair of samples.	41
6.17	The Triplet Loss Architecture. The boxes labeled <i>feature extraction</i> are neural network layers with shared weight with one another. The goal is to learn features where samples from the same class are close and samples from different classes are far apart.	43
6.18	Taken from [Taigman et al., 2014]. Depicts their 3d alignment method. . .	43
7.1	Depiction of the steps of traditional transfer learning with fine-tuning. The Classifier in step one is initialized with random weights. Step 2 is optional and usually employed to improve performance by learning some feature patterns that are specific to the new domain,as a trade-off, it could also lead to overfit.	46
7.2	Pipeline of our Face Verification Approach.	47
7.3	Illustration of how we compute metrics with a convolutional (locally connected) operation. Each variable is weighted by the convolutional filter, meaning it learns, in theory, which variables are more or less important. . .	49

List of Tables

4.1	Results on the CIFAR-10 test dataset.	20
4.2	Elapsed time in different methods for pedestrian detection.	23
7.1	Handcrafted distance metrics used. Where x and y are two vectors and all operations applied are pointwise, that is they also return a vector.	47
8.1	Accuracy obtained in Labeled Faces in the Wild when using the feature map obtained from layer <i>pool5</i> with different metrics. The + sign indicates concatenation of the metric results.	51
8.2	Accuracy obtained in Labeled Faces in the Wild when using the feature map obtained from different layers. We also show the results of the handcrafted feature LBP in the last row. $L_1 + Sign$ metric used.	52
8.3	Accuracy obtained in Youtube Faces when using the feature map obtained from different layers. $L_1 + Sign$ metric used.	53
8.4	Comparison of the state-of-the-art Results in the Labeled Faces in the Wild. A unpaired T-Test of our result and the one reported in DeepFace [Taigman et al., 2014] shows these results to not be statistically different.	53
8.5	Comparison of the state-of-the-art results in the Youtube Faces. Results of the third row are their reported results without triplet loss embedding, also, they do not report standard error.	54

Contents

Acknowledgments	ix
Abstract	xiii
List of Figures	xv
List of Tables	xix
1 Introduction	1
1.1 Motivation	2
1.2 Objectives	3
1.3 Contributions	3
1.4 Work Organization	4
I Dimensionality Reduction	5
2 Related Works	7
2.1 Image Classification	7
2.2 Pedestrian Detection	8
2.3 Dimensionality Reduction	10
2.3.1 Principal Component Analysis	10
2.3.2 Partial Least Squares	10
3 Boosted Projections	13
3.1 Proposed Approach	14
4 Experimental Results	19
4.1 Image Classification	19
4.2 Pedestrian Detection	20

5	Conclusions	25
II Transfer Learning		27
6	Related Works	29
6.1	Neural Networks	29
6.1.1	Multilayer Perceptron	29
6.1.2	Convolutional Networks	32
6.2	Face Verification	38
7	Transfer Learning	45
7.1	Proposed Approach	47
8	Experimental Results	51
9	Conclusions	55
	Bibliography	57

Chapter 1

Introduction

Images are a two dimensional representation of a scene, and a mean to register this scene and extract information from it, by detecting objects, which could be associated with a tool used by a perpetrator of a crime, or by recognizing a face to open an automated door. Machine learning and computer vision are usually employed to tackle these tasks, where neural networks with deep architectures have gained much popularity and broke many state-of-the-art records.

This work tackles two problems related to the aforementioned applications: dimensionality reduction and transfer learning of deep models. The first is presented through a novel boosted projection approach applied to pedestrian detection and object classification. The latter is presented in a transfer learning method requiring little to no fine-tuning, applied to the face verification task.

Machine learning and computer vision problems usually suffer from very high dimensional data. Therefore, the employment of dimensionality reduction techniques is common to achieve reduction in training and testing cost and, most importantly, as a workaround to the curse of dimensionality, as can be seen in the work of Guyon and Elisseeff [2003].

Even with dimensionality reduction, deep models require large amount of data and learning such models would still be very costly. Transfer learning refers to reusing a model learned on a large corpus on a task with only a small corpus available. Usually, this process is conducted by means of fine-tuning, which involves model learning, which is costly.

In the first part of our work, we analyze the impact of the proposed Boosted Projection coupled with Partial Least Squares (PLS) [Wold, 1985], a method for dimensionality reduction that has been extensively explored in many tasks across the literature [Rosipal and Krämer, 2006; Uzair et al., 2015; Akata et al., 2014; Lowe,

2004]. To evaluate the proposed method, we test it in the pedestrian detection, using the INRIA pedestrian detection benchmark [Dalal and Triggs, 2005], and in the image classification task, using the CIFAR-10 [Krizhevsky and Hinton, 2009] benchmark. On the former task, our method achieves an improvement over the work of Jordao and Schwartz [2016], a PLS based random forest, which to the best of our knowledge is the most recent PLS-based approach for pedestrian detection. On the latter task, we achieve an average accuracy better than the single PLS approach with the same number of dimensions.

The second part of our work proposes a simple method of performing transfer learning by picking the most suited layer to extract features in the new task. This also investigate a hypothesis that sequential layers are correlated to complexity and specificity to the original dataset used. In this part we conduct experiments in the labeled faces in the wild dataset Huang et al. [2012] and in the Youtube Faces Database Wolf et al. [2011], which are popular datasets in the literature having been used by many works such as Taigman et al. [2014]; Parkhi et al. [2015]; Schroff et al. [2015]. Our results were comparable to the state of the art scenario on the Labeled Faces in the Wild dataset and on the Youtube Faces dataset.

1.1 Motivation

Some events, such as major terrorist attacks, violence in crowded places, sports events, lead to an increased demand for security in society. This has resulted in the deployment of large CCTV systems. For instance, London Underground and Heathrow Airport have more than 5000 cameras each [Valera and Velastin, 2005]. This increasing demand for security by society leads to a growing need for surveillance activities in many environments.

Due to this large growth in data availability, the amount of raw surveillance data from videos is humongous and human agents have a hard time to interpret relevant data from all those hours of video content. Two important tasks that depend on understanding of surveillance data are pedestrian detection and face verification.

Dimensionality reduction is important to circumvent the curse of dimensionality and to handle this large volume of data. We develop a method for dimensionality reduction that is based on the hypothesis that by combining multiple projections in an ensemble exploring discriminative information, classification error, we could reduce dimensionality with better discriminative results.

The second part of this work is motivated by the fact that in the sequential net-

works, such as VGG or AlexNet, each layer is built on top of the patterns learned by the previous. Implying that the layers follows a pattern of patterns structure. We believe that this means a correlation between depth of the layer and its complexity/specificity to the problem it was trained. This has an implication that to use a previously learned model on a different data than the original, there should be a layer that is most suitable, since the deeper layers would then have learned patterns more specific to the original problem.

1.2 Objectives

This work explores machine learning solutions for tackling surveillance and computer vision problems. We can divide the objectives into two main parts. First, we intend to demonstrate the viability of a novel framework inspired by the adaptive boosting [Freund and Schapire, 1995] to handle the problem of dimensionality reduction, this problem is still relevant since some approaches [Taigman et al., 2014; Schroff et al., 2015] in computer vision today consists of extracting features with deep learning model which have a high dimensionality and tends to suffer from the curse of dimensionality. The second objective is to verify our hypothesis of correlation between layer depth and specificity/complexity in sequential convolutional neural networks applied to the face verification task.

1.3 Contributions

Our first contribution is a novel dimensionality reduction framework, which we show its usefulness in the surveillance context of the pedestrian detection task. Our second contribution is a method that shows a simple transfer learning technique that achieves results comparable to the state-of-the-art.

The publications achieved with this work are listed as follows.

1. Kloss, R. B., Jordao, A., and Schwartz, W. R. (2017). Boosted Projections: An Ensemble of Transformation models. In 22nd Iberoamerican Congress on Pattern Recognition.
2. Kloss, R. B., Jordao, A., and Schwartz, W. R. (2018). Face Verification: Strategies for Employing Deep Models. The 13th IEEE Conference on Automatic Face and Gesture Recognition. (accepted).

1.4 Work Organization

We divide this work in two parts. One regarding our dimensionality reduction approach, Part I, and the other part regarding our transfer learning approach, Part II.

The first part is organized in related works, Chapter2, where we discuss some of the most famous dimensionality reduction methods and also some of the literature in the two tasks we performed our experiments, image classification and pedestrian detection. Then, we present our method in Chapter 3 and the results and discussions of this method in Chapter 4. Finally, Chapter 5, concludes this part of our work.

The second part is organized in related works, Chapter6, where we discuss the foundations of neural networks, and convolutional networks, and also the literature regarding face verification and some of the recent and important works with neural networks in general. We present our method in Chapter 7 with its results and discussions in Chapter 8. Chapter 9, concludes the final part of our work.

Part I

Dimensionality Reduction

Chapter 2

Related Works

In this chapter we discuss works regarding ensemble of models and dimensionality reduction. Last, we review the literature on the tasks of our validation experiments.

2.1 Image Classification

Image Classification consists of deciding to which, of a number of given types, the image belongs. These types are denominated classes, hence, the name, classification. This task can be used, for instance, to extract semantic information from images, which can aid in information retrieval or used as a feature in the classification of a document with images in it, and can also be used in robotics to help in the control of an intelligent agent. With this said, the importance of studying and researching this task becomes clear.

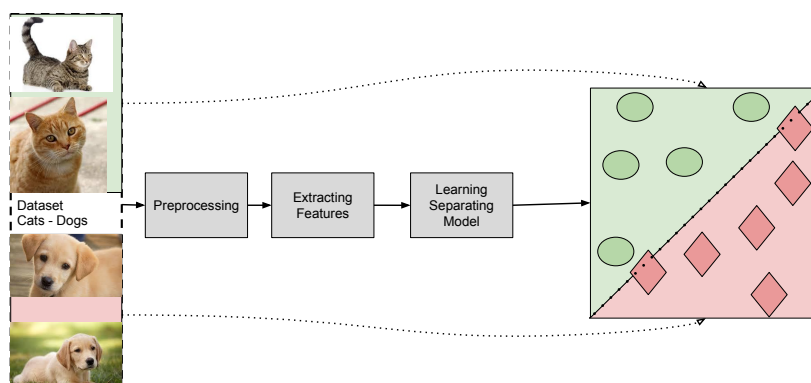


Figure 2.1. Illustration of a standard image classification pipeline exemplified with classes of images of dogs and cats.

Traditionally, image classification is performed by extracting features from the

images and using these features in a pattern recognition algorithm to learn a model that could separate images from one class to images from another class, which is illustrated in Figure 2.1.

In early works, the features extracted from images consisted of algorithms that exploited structure in the image to obtain information. These algorithms would extract, for instance, shape information, HOG [Dalal and Triggs, 2005], texture information, LBP [Ojala et al., 1996], or color information, through a simple color histogram, for example. Extracting these features is important to be able to map images to points in the space that can be separated, the shape information, for instance, could map the images in such a way that balls are distant from boxes. These features acts as an alternative to dealing with raw pixels which are noisy, non-sparse and high dimensional, and it is often times very hard for a classifier algorithm to learn a discriminating hyperplane that can separate image samples through their pixel input alone.

Before the breakthrough of deep learning models [Krizhevsky et al., 2012], a popular strategy for feature extraction was to use the bag-of-visual-words model, or alternatives [Lazebnik et al., 2006], which consist of finding parts, or visual words, of samples with representative features, extracted before-hand. Then, a sample is represented by a signature that consists of the similarity comparison of its parts against the representative parts. The main idea is that same objects have, roughly, the same parts. Interesting to note is that deep learning architectures work similarly, since the nodes in the first layer find discriminative parts and the feature map used as input in successive layers contains pretty much this similarity regarding the image parts.

With the advent of deep learning, the neural networks were able to extract rich and powerful features, and low-level features, HOG and LBP, or mid-level features were made, in general, obsolete. The deep features are capable of conveying high level information, and capturing complex parts of objects. As mentioned before they have contributed to a breakthrough in this task and have the best results in one of the most competitive benchmark for the task, the ImageNet competition.

2.2 Pedestrian Detection

The work of Schwartz et al. [2009] is one of the most successful examples of dimensionality reduction applied to pedestrian detection [Dollár et al., 2012]. In their work [Schwartz et al., 2009], the authors describe the human body by using the Histogram of Oriented Gradient (HOG) [Dalal and Triggs, 2005] combined with extra information provided by co-occurrence and color frequency features. This combination

of features generated a high dimensional feature space, rendering many traditional machine learning techniques intractable. To address this problem, the authors employed PLS to project the high dimensional feature space onto a low dimensional latent space before performing classification. We made use of this projection to a low dimensional feature space in order to improve classification in our work. Moreover, our proposed method use around 30 times fewer features, compared to the work of Schwartz et al. [2009], and achieves a more accurate detection.

In a similar way, Qiu and Sapiro [2013] proposed a method to project the original feature space into a new feature space. The goal of this projection, referred to as linear discriminative transformation, is to improve the class separation at each node in a decision tree. For this purpose, the authors employed the nuclear norm [Grothendieck, 1996], where each node composing an orthogonal decision tree, uses the arriving samples to learn this transformation. Furthermore, Nam et al. [2014] used the Linear Discriminant Analysis to transform a local decorrelation of the features, where the inter-class variation is minimized while the intra-class is maximized. In their work, they showed that by applying this transformation on the data it is possible to achieve a better data separation, which enables the use of simple classifiers, e.g., orthogonal decision trees. The idea of performing the dimensionality reduction is to find the most discriminative feature among the available features

Recent studies [Marín et al., 2013; Jordao and Schwartz, 2016] showed that by using strong classifiers as members of the ensemble, it is possible to achieve better results than the traditional approach that use weak classifiers [Criminisi and Shotton, 2013]. Marín et al. [2013] proposed an ensemble of decision trees, where each node that composes the tree consists of a linear SVM. The intuition behind building this type of decision tree, called oblique decision trees [Criminisi and Shotton, 2013], is that each node at the depth d is responsible for classifying the samples misclassified by the nodes at depth $d-1$. Jordao and Schwartz [2016] proposed an oblique decision tree where for each tree node a PLS regression is learned, which differs from the SVM chosen in the work of Marín et al. [2013]. Their work showed that PLS is more appropriate than SVM to generate oblique decision trees, since each tree node learns a richer representation of the features due to the projection provided by PLS. In this work, we show that our proposed method is statistically faster and obtains a more accurate detection regarding the work of Jordao and Schwartz [2016].

2.3 Dimensionality Reduction

Computer vision features can have large number of variables and this can lead to a poor performance of some classifiers. This is a well-known problem often called the curse of dimensionality. Dimensionality reduction is a workaround to the problem of the curse of dimensionality and can be used together with a classifier in order to improve cost effectiveness and classification performance.

2.3.1 Principal Component Analysis

Principal Component Analysis (PCA) [Wold et al., 1987] is a method that learns a rotation of the original data space, where, in the rotated space the covariance matrix of the data is approximated to a diagonal matrix. In this rotated space, its axes are pointing in the direction of maximum variance between the variables (features), as seen in Figure 2.2.

Mathematically, PCA uses singular value decomposition (SVD) to find a transformation that makes the covariance matrix of the transformed samples a diagonal matrix, as illustrated in Equation 2.1, where Z is the centered input. The goal is to find the rotation matrix R which rotates the space in such a way that the covariance matrix Σ_{Rot} is diagonal. This rotation matrix, R , is equivalent to a matrix with the eigenvectors of the covariance matrix of the input as columns.

$$\begin{aligned} Z^T Z &= \Sigma \\ (Z * R)^T * (Z * R) &= \Sigma_{Rot}. \end{aligned} \tag{2.1}$$

The SVD of the Z matrix gives a matrix that is equivalent to the rotation matrix R .

2.3.2 Partial Least Squares

The Partial Least Squares (PLS) is a dimensionality reduction technique employed to model the relationship between dependent and independent variables [Rosipal and Krämer, 2006]. A brief definition of the PLS is shown below.

Let $X \subset R^m$ be an n by m matrix representing n data samples in a m – *dimensional* space of features, and $Y \subset R$ represents the label (classes matrix). The method decomposes X and Y as

$$X = TP^T + E, \quad Y = UQ^T + F, \tag{2.2}$$

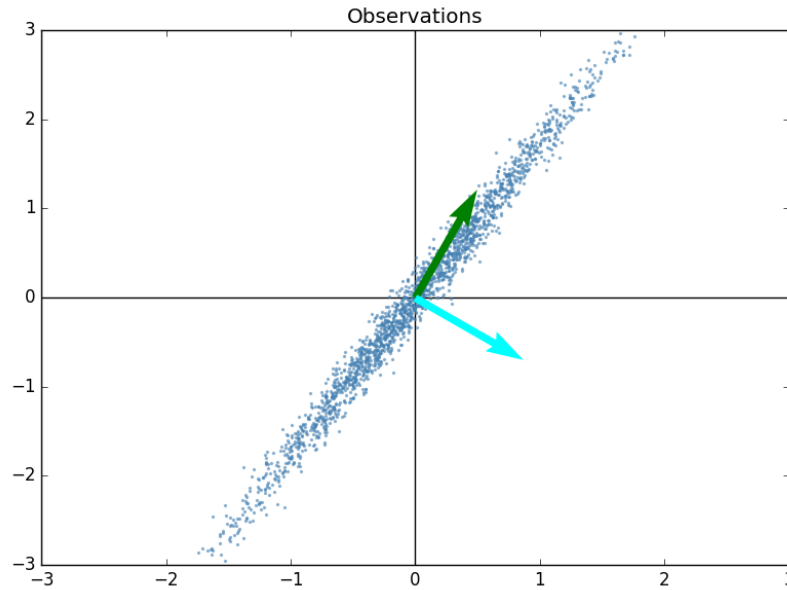


Figure 2.2. Principal components shown in green and cyan. It illustrates that the components are roughly in the directions of maximum variance.

where T and U are $n \times p$ matrices of variables in latent space, p is a parameter of algorithm. P and Q correspond to matrix $m \times p$ and vector $l \times p$ of loadings, in this order, where l is the number of dimensions in Y . The residuals are represented by E and F , matrices of size $n \times m$ and $n \times l$, respectively. The PLS method, constructs a matrix of weights, W , which rotates the original feature space in such a way that the axis are in the direction of maximum covariance between the features and the label matrix. The columns of W is computed by using the nonlinear iterative partial least squares algorithm (NIPALS) [Wold, 1985]. A more in-depth mathematical definition can be found in [Wold, 1985; Rosipal and Krämer, 2006].

PLS, as with PCA, is also related to an eigen problem, in the case of PLS the problem consists of finding the eigenvectors of $(X^T Y)^T (X^T Y)$ and $(Y^T X)^T (Y^T X)$ or analogously solving the singular value decomposition (SVD) of $X^T Y$ and $Y^T X$, more explanation of the SVD problem modelling in [Bookstein, 1994; McIntosh et al., 1996].

Chapter 3

Boosted Projections

It has been shown that performing dimensionality reduction on the raw features improves its representation while also reducing computational cost [Schwartz et al., 2009; Sánchez et al., 2013; Martis et al., 2013; Jordao and Schwartz, 2016]. The idea is to use a projection on the raw data to find a new space that uses a smaller number of features and improve classification performance. For instance, the Partial Least Squares (PLS) technique [Rosipal and Krämer, 2006] was employed to project to a low dimensionality space, achieving better results than the state-of-the-art on the task of hyperspectral face recognition with spatiospectral information by Uzair et al. [2015]. In another case, Akata et al. [2014] employed Principal Component Analysis (PCA) to reduce the dimensionality of their SIFT [Lowe, 2004] descriptors and circumvent the curse of dimensionality. It is important to note that such approaches could also be used together with deep learning in order to potentially compact and improve deep features

Besides the use of projection methods, another way of improving the classification is the employment of ensemble classifiers. These classifiers have been widely employed in several machine learning tasks [Avidan, 2007; Takemura et al., 2010; Cogranne and Fridrich, 2015; Jordao and Schwartz, 2016] due to its simplicity and good results. In general, ensemble classifiers are composed of weak classifiers (e.g., decision stumps) that are able to achieve a powerful classification when combined.

Inspired by the aforementioned studies, we propose a novel method that combines dimensionality reduction and ensemble techniques to find richer variables than those estimated by a single projection model. The units of our ensemble are modeled as nodes built iteratively. Each node is composed of a projection model and a weak classifier learned with a distinct subset of samples hard to classify (referred to as *hard samples*) by the previous node, i.e., samples that were incorrectly classified, according

to a threshold. The idea behind this procedure is to create projection models yielding rich and discriminative features for different categories of samples (from samples easier to harder to classify). It is also possible to cascade the projection obtained by a set of nodes to a new set of nodes, this is represented by multiple layers. The goal of using multiple layers is to reduce possible redundancy between nodes.

Our proposed method is similar to the Adaptive Boosting [Freund and Schapire, 1995] in the sense that it reweighs samples according to their classification score. However, while the adaptive boosting focuses on grouping classifiers, we use the classifier as a tool to find hard samples. This is also similar to the hard mining employed by Dalal and Triggs [2005], except we do not search for these samples on a validation set but on the training set itself. Another difference is that we iteratively search for hard samples to construct multiple models, whereas Dalal and Triggs only employ this operation to construct a single model. Since we use the boosting-based idea of reweighing hard samples applied to a projection context, we refer to our method as *Boosted Projection*. The proposed method takes advantage of the characteristics of the underlying projection method, for instance reducing redundancy, and it may also presents richer features due to the employment of the ensemble scheme that explores hard samples.

The contribution of this part of the work is the proposal of a novel method for feature projection, which demonstrates the possibility to enhance data representation exploiting the advantages provided by dimensionality reduction techniques and ensemble methods. In addition, the main advantage compared to other dimensionality reduction methods is that the nodes in our method explores different subsets of the data, where the deeper the node, the harder it is to classify its subset of samples, which can better convey discriminative information.

3.1 Proposed Approach

Our goal is to create an ensemble of projection methods that can yield richer and more discriminative features than a single projection method. To achieve such goal, we propose a framework for dimensionality reduction that explores discriminative information by employing a classifier to find samples that are hard to classify (*hard samples*) and using this information to learn new projections that prioritize the fitting of such samples. The method is shown in Figure 3.1.

The first unit of our framework is a *node*, represented in detail on Figure 3.2, which receives features of samples as input and learns a projection model that generates latent variables, used as new features. This projection is performed using a

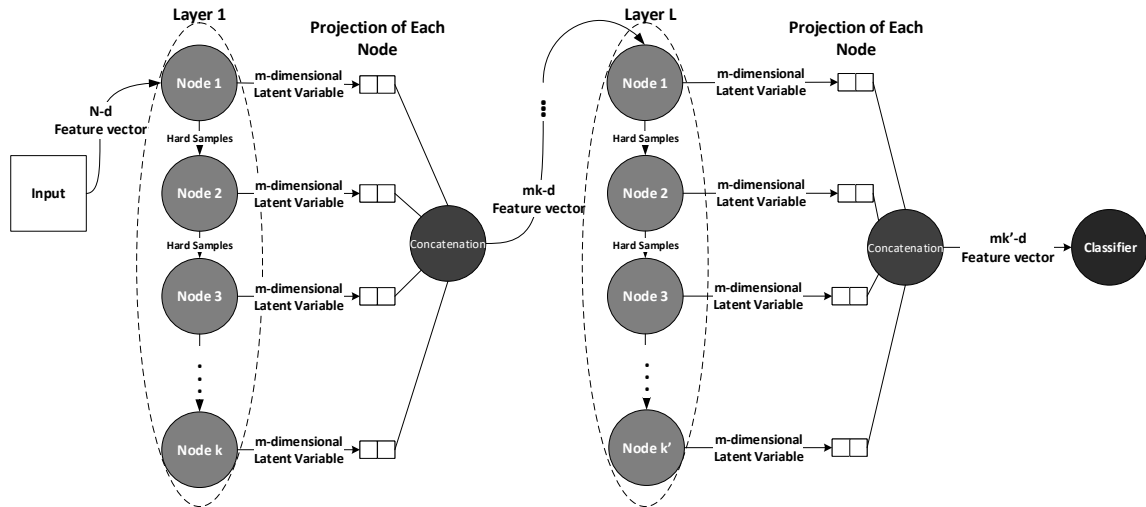


Figure 3.1. General structure of our proposed framework for improved projections. The node of a layer learns a projection using the hard samples found by the previous node, this continues until convergence. Then, it is possible to create a new layer, where the input features are the concatenation of the projections of each node in the previous layer. The number of nodes and layers are hyperparameters.

traditional projection method such as PLS [Wold, 1985]. The latent variables are used by a classifier to learn a discriminative model. By using the confidence score of this classifier and a threshold, we find samples that have been misclassified. That is, a model is trained on the projected features and we find the subset of samples that have been misclassified given a threshold, i.e., whose classifier score is below this threshold (*hard samples*). The confidence threshold is initialized with an arbitrary value and is decreased for each new node in order to make samples easier to classify and guarantee that the nodes have more diversity. This is important since in ensemble methods, a high diversity is desirable [Brown and Kuncheva, 2010; Woźniak et al., 2014], because you want incremental coverage.

After the hard samples have been obtained for a given node, a subsequent node is created by learning a new projection and a new classifier considering such samples. This new classifier learns a different separating hyperplane which will score samples differently leading to a new subset of hard samples. This procedure continues until convergence, which can be met either by reaching a maximum number of nodes or by having no more misclassified samples. Thus, each node learns a model based on different subsets of data, where the deeper the node, the harder it is to classify its subset of samples. Each node also specializes on the subset of samples it correctly

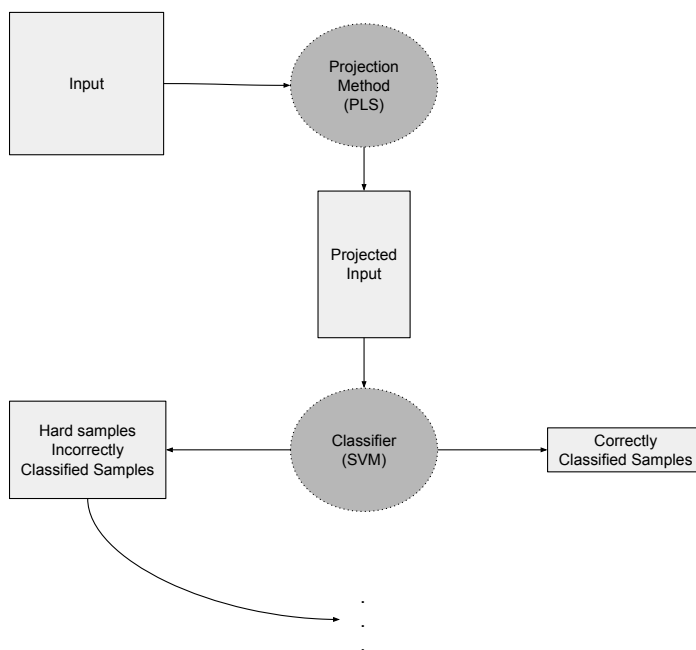


Figure 3.2. Detail of the Node of the Boosted Projection.

classified, as shown in Figure 3.3 along with this cascading process.

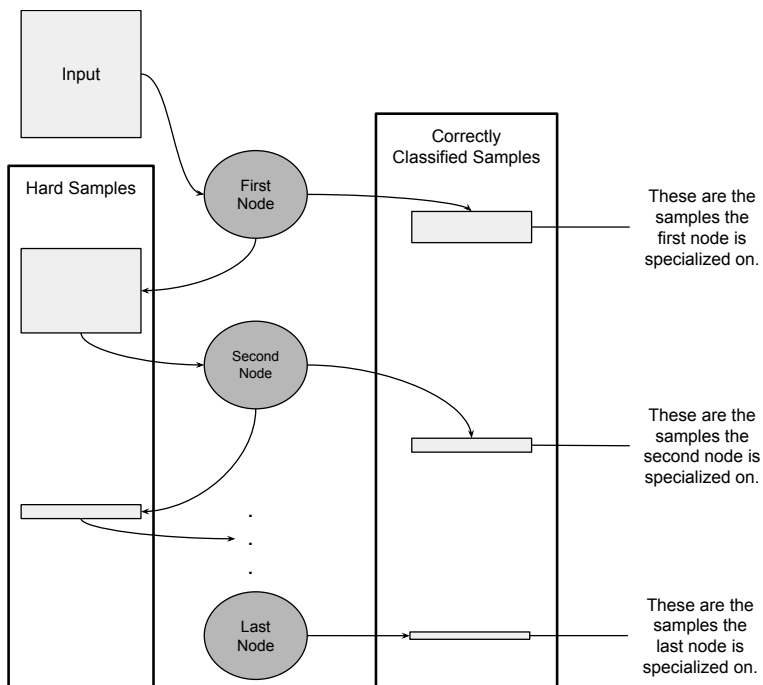


Figure 3.3. Explanation of the specialization of the nodes in the Boosted Projection.

We refer to the set of learned nodes built until the convergence as a *layer*, as

illustrated in Figure 3.1. The output of a layer is the concatenation of the latent variables of each node of that layer. After all the nodes of a layer have been built, its output can be used by a new node of a new sequential layer to start the whole process again. The benefit of using additional layers is that if there is redundancy in some nodes, probably because of a lack of diversity, the nodes of the next layer could be able to filter out this redundancy. The number of layers is a parameter of the framework.

The output of the last layer, which is the concatenation of all projections, is used to learn a final classifier, which might be different than the one used to find hard samples. It is important to emphasize that the output of the last layer is composed of projections learned for different subsets of samples and based on different features, causing each node's output to be an expert on that subset. This is an advantage of the proposed approach over a single projection method (such as PLS). A singleton approach would find the projection that is best for the average of the whole data and not suitable projections for parts of the data with different characteristics, which would not represent well the true data distribution.

Chapter 4

Experimental Results

In this section, we present our experiments and results achieved. First, we evaluate it on the task of *image classification* with the **CIFAR-10** benchmark, using the average accuracy, the dataset’s standard protocol, to measure our performance against a single projection method. Then, we evaluate our boosted projection method on the **INRIA pedestrian detection** benchmark, using log-average miss rate, which is its standard protocol, and compare it with other baseline approaches.

4.1 Image Classification

To evaluate our method on the image classification task, we use the CIFAR-10 [Krizhevsky and Hinton, 2009], a dataset with 32×32 pixels RGB colored images of ten classes. We extract deep features without data augmentation using the Keras Framework [Chollet, 2015] with a VGG16 [Simonyan and Zisserman, 2014b] network. We tune the C parameter of the Support Vector Machine (SVM), with linear kernel, by employing stratified cross validation on the training set. The parameter was found to be optimal as $C = 0.1$. Finally, we use the average of the accuracy of each class as the evaluation metric, which is the number of true positives over the sum of the number of true positives and false negatives (higher values are better).

The first column of Table 4.1 specifies the method used, which is either a SVM, a PLS transformation followed by a SVM, or the boosted version of PLS (Boosted PLS). The boosted version also uses a SVM to mine for hard samples and as the final classifier. The second and third columns show the results and the number of dimensions for each method, respectively. Regarding the dimensionality (third column of Table 4.1), in the case of the SVM, the dimensionality is the original dimensionality of the features (4096). However, with the dimensionality reduction method (PLS), the

Table 4.1. Results on the CIFAR-10 test dataset.

Method	Average Accuracy	Dimensionality
SVM	71.62%	4096
PLS + SVM	69.10%	10
PLS + SVM	74.52%	30
PLS + SVM	75.24%	200
PLS + SVM	75.39%	500
Boosted PLS (1 layer)	75.94%	$20 \times 10 = 200$
Boosted PLS (2 layers)	76.25%	$20 \times 10 = 200$

reported dimensionality is the number of components used. Furthermore, with the boosted methods, we report the dimensionality as the product of the number of nodes and the number of components of the underlying projection method.

According to the results in Table 4.1, the more components the models have, the better its representation. We can also see a correlation between the accuracy and the number of components. Employing the Boosted PLS transformation, with 10 components, and classifying with SVM (Boosted PLS) also achieved better results than with just the PLS transformation of the same model (PLS + SVM). In this case, the improvement is likely due to the fact that while PLS transformations minimize an average error, but due to the specialization of each node, our Boosted PLS can preserve information of small distributions in the data that deviates from the majority, which were represented in the first nodes.

Regarding the PLS transformation (rows 2 to 5 in Table 4.1), we can see that from 10 components to 30 components there is an improvement of 5.42 percentage points, between 200 components and 500 there is only an improvement of a 0.15, illustrating the saturation of the components learned. Our method, however, with 200 dimensions, was able to have an improvement over the 500 components PLS model of 0.86 percentage points.

4.2 Pedestrian Detection

The studies of Schwartz et al. [2009] and Jordao and Schwartz [2016] are, to the best of our knowledge, the most recent and withworks concerning PLS based dimensionality reduction applied in the context of pedestrian detection. Therefore, to show that our method achieves better results than traditional projection-based approaches, we employed it to the pedestrian detection problem and compare with other methods that employ comparable features, HOG and LBP.

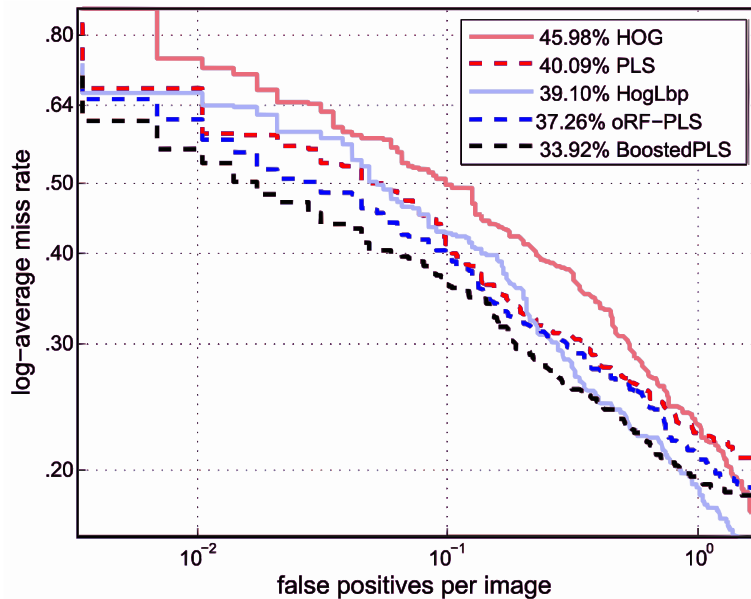


Figure 4.1. Comparison of our boosted PLS projection with other approaches found in the literature. Results using the log-average miss-rate (lower values are better) of 10^{-2} to 10^0 (standard protocol).

To validate our results, we adopt the evaluation protocol used by state-of-the-art works, which is called *reasonable set* [Dollár et al., 2012] (a detailed discussion regarding this protocol can be found in [Dollár et al., 2012; Benenson et al., 2014]), where the results are reported on the log-average miss rate. In addition, similarly to Jordao and Schwartz [2016], we used the TUD pedestrian dataset as validation set, to calibrate some parameters of our method.

Our first experiment evaluates the impact of the number of nodes and layers to compose the boosted projection. On the validation set, the best values for these parameters were 20 and 3, respectively. The remaining parameter to be defined in our method is the number of components (latent variables), for each PLS that will compose a node of a layer (see Chapter 3). Similarly to [Jordao and Schwartz, 2016] and according to the results achieved on the validation set, we note that this parameter is the most important parameter for PLS-based methods since varying this parameter from 3 to 6, the log-average miss rate decreases from 53.20 to 50.07, respectively. Based on the results obtained in the validation, our final PLS based boosted projection was set to 3 layers, where each layer is composed of 20 nodes that project the data to a 6-dimensional feature space using the PLS.

In our next experiment, we compare the results of the proposed Boosted Projection with other baseline methods. To provide a fair comparison, we considered the

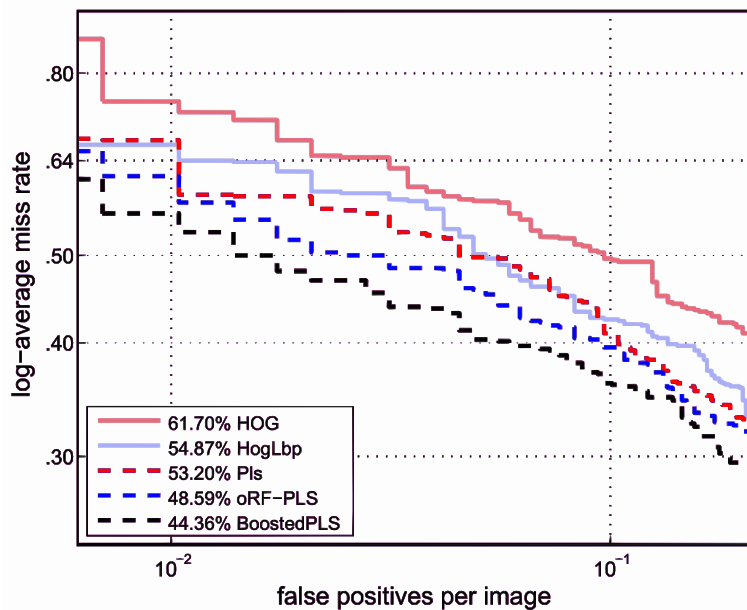


Figure 4.2. Another comparison of our boosted PLS projection with other approaches found in the literature. Results using the area from 10^{-2} to 10^{-1} .

results reported by the authors in their works. Figure 4.1 shows the log-average miss rate (on the standard protocol) achieved by methods on the INRIA person dataset. Our method achieved a log-average miss rate of 33.92, outperforming the works of [Schwartz et al., 2009] and [Jordao and Schwartz, 2016] in 6.17 and 3.34 percentage points, respectively. Regarding the area under the curve from 10^{-2} to 10^{-1} , showed in Figure 4.2, (which represents a very low false positive rate), our Boosted version of PLS (Boosted PLS) still outperforms the other methods, a evidence of being robust to hard detections. Moreover, our proposed method uses around 30 times fewer features, compared to the work of [Schwartz et al., 2009], and achieves more accurate detections. It is also important to reinforce that our method is able to outperform detectors that employ more complex features, for instance, HOG-LBP [Wang et al., 2009], while using simple HOG-features (the same setup as proposed by [Dalal and Triggs, 2005]).

Our last experiment, illustrated in Table 4.2 regarding pedestrian detection evaluates the computational cost compared to the work of Jordao and Schwartz [2016], which, to the best of our knowledge, is the most recent PLS-based method. We conduct this experiment to measure and compare the computational time of our method, since pedestrian detection is often a initial step in the surveillance pipeline, and therefore, needs to be fast. We execute, on the same hardware configuration of [Jordao and Schwartz, 2016], the detection on a 640×480 image for 10 times, as is done

by Jordao and Schwartz [2016], and we compute its confidence interval using 90% of confidence. Our method obtained a mean of 67.395 against a mean of 271.32 of [Jordao and Schwartz, 2016], The confidence interval of the means with 90% of confidence is depicted in Table 4.2, and since there is no overlap between them it shows that the methods present statistical difference of their execution time. Therefore, according to the results, our method is four times faster than [Jordao and Schwartz, 2016]. This reduction in computational cost is possible because our method uses fewer PLS projections than [Jordao and Schwartz, 2016] (even if we consider the optimal case, where each oblique decision tree consists of only one PLS projection, see [Jordao and Schwartz, 2016] for details).

Method	Confidence Interval of Elapsed Time
Boosted PLS	[67.14s, 67.64s]
oRF-PLS [Jordao and Schwartz, 2016]	[270.91s, 272.73s]

Table 4.2. Elapsed time in different methods for pedestrian detection.

Chapter 5

Conclusions

In this part of our work, a novel approach for dimensionality reduction, the *Boosted Projection*. The method focuses on the idea of using several projection models (e.g. Partial Least Squares), to build an ensemble with more compact and richer representation of the raw features. We conducted experiments on two important computer vision tasks: pedestrian detection and image classification. In the first, we demonstrate that the proposed method outperforms many pedestrian detectors, using simple features. In addition, our method is more accurate and faster than [Jordao and Schwartz, 2016], one of the most recent detector based on PLS. In the second task, we demonstrate that the proposed method is able to compute features richer than a single dimensionality reduction method.

For future works, we intend to test our proposed approach with different input features, with more tasks and with different transformation models.

Part II

Transfer Learning

Chapter 6

Related Works

In this chapter, we present a theoretic foundation for neural networks and an overview of state-of-the-art methods based on neural networks.

6.1 Neural Networks

In this section we introduce the basics of artificial neural networks, and then explain and explore the literature on the most used networks in computer vision, the Convolutional Networks.

6.1.1 Multilayer Perceptron

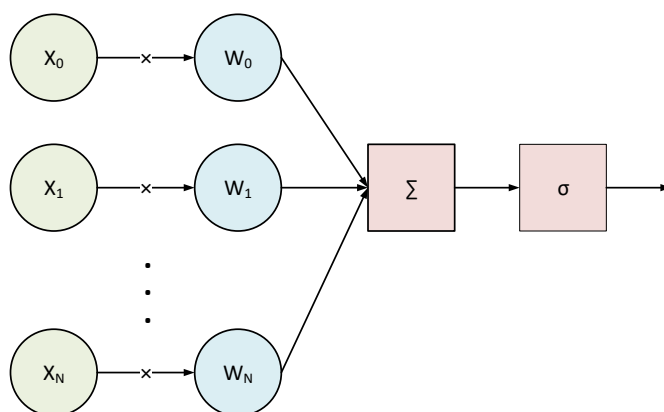


Figure 6.1. The Original Perceptron with sigmoid activation. In matrix notation: $\hat{y} = \sigma(X \cdot \vec{w})$, where $\sigma(x) = \frac{1}{e^{-x} + 1}$

Artificial Neural Networks are ensembles of Perceptrons [Rosenblatt, 1958] which is a simple model for a Neuron as envisioned by Rosenblatt. What a Perceptron actually does is a linear regression coupled with a non-linear activation. The first Perceptron model used a sigmoid function for this non-linearity, which limited the output to the range $[0, 1]$, this is analogous to the all-or-none law, where the response to a stimulus does not vary with the intensity of the latter [Cannon, 1922]. Figure 6.1 illustrates this Perceptron model, the variables in the input denoted by X_i are multiplied by weights and summed to generate an output, which is a multivariate linear regression.

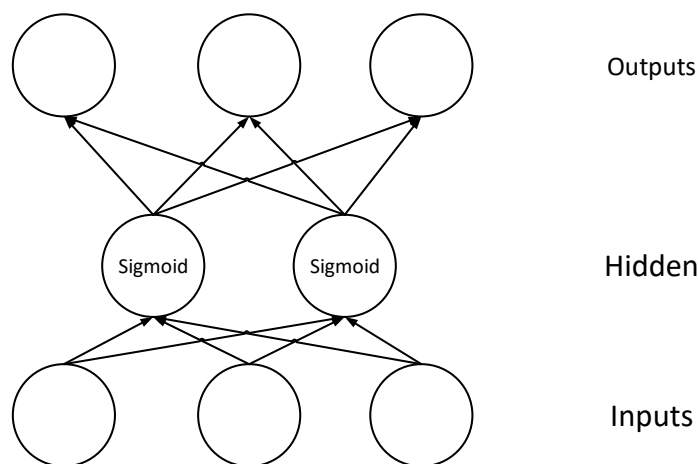


Figure 6.2. The Multilayer Perceptron. Each circle is a Perceptron and each Level (Inputs, Hidden, Outputs) is called a Layer.

The Multilayer Perceptron (MLP), as depicted in Figure 6.2, is a combination, or ensemble, of different linear regressors, the Perceptrons. The MLP model, with one hidden layer, can be represented in matrix notation as the following equation, $\hat{y} = \sigma(\sigma(X \cdot W^{(0)}) \cdot W^{(1)})$, where $W^{(i)}$ is the weights matrix of the i -th layer. The hidden layer functionality is similar to that of a kernel [Hofmann et al., 2008]. Whereas the kernel uses a handcrafted function that is equivalent to transforming the inputs to a higher dimensional space and computing a distance metric, the hidden layer learns this transformation from the data. The goal in both methods is to make non-linear problems solvable. Another point of notice in MLP models, is that the aforementioned non-linearity applied to a Perceptron is very important in the ensemble, and this benefit is far from the biological motivation. If linear functions are used the resulting transformation can actually be represented through only one layer, because successive matrix products can be represented by only one matrix product, e.g., $X \cdot W^a \cdot W^b = X \cdot W$, where $W = W^a \cdot W^b$. By using non-linearities, the output of the Perceptrons can also

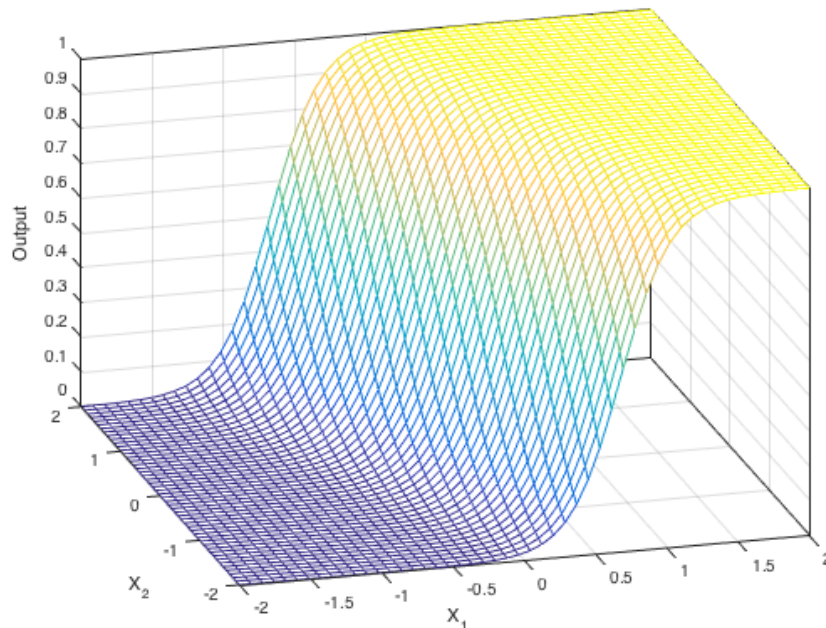


Figure 6.3. Boundary space of a Sigmoid Perceptron.

be combined to approximate any arbitrary function [Hornik, 1991].

The Perceptron is a linear classifier, which means it divides the space linearly in the shape shown in Figure 6.3. However, by combining partitions of different Perceptrons leads to a space partition that is non-linear, this is the idea behind the MLP, the simplest artificial neural network kind of model. Figure 6.4 shows a MLP, trained to solve the non-linear exclusive-or problem. In the particular case depicted in the figure, there is only one hidden layer with five neural nodes, or neurons, their partition in the space can be seen in the figure and its combination generates the final partition illustrated by the far right region of the figure. This characteristic of using several linear nodes to compose a stronger non-linear classifier is also exploited in Decision Trees and Random Forests [Ho, 1995].

We can see that each node in the hidden layer of the model showed in Figure 6.4 detects a pattern, which is a region of the 2D space. By stacking layers together, each layer is detecting patterns in the space of the patterns learned by the previous layers. This is able to capture more complex patterns which helps in solving harder, more complex, problems, such as the ones in computer vision, where the input are raw pixels which are noisy and by themselves alone lack semantic meaning.

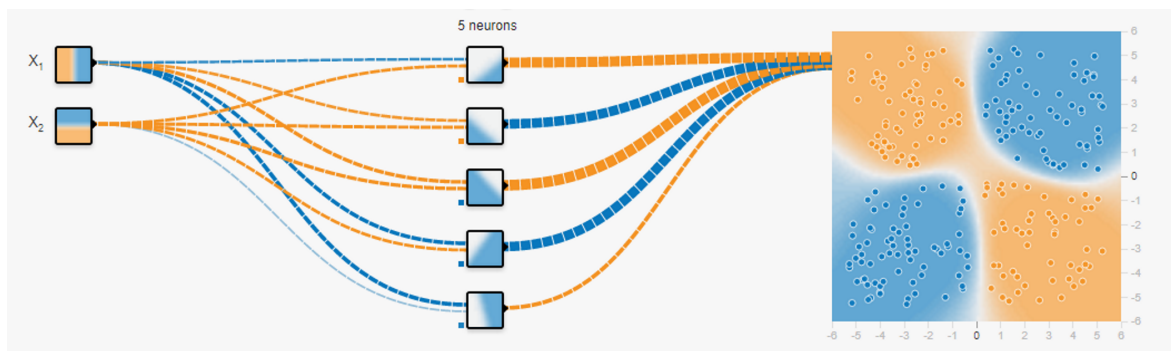


Figure 6.4. Combining the space partitioning of multiple linear sigmoid perceptrons, the multilayer perceptron is able to solve the XOR non-linear problem. Adapted from <http://playground.tensorflow.org/>. The input consists of a 'x' and 'y' coordinate, and it is fed to each of five neurons in a sequential hidden layer where each of these neurons have learned a linear space separation that when combined leads to the non-linear boundary space of the output depicted in the far right cartesian space.

6.1.2 Convolutional Networks

A convolutional layer, as depicted in Figure 6.5, consists of a dot product of a certain region of the input against a "filter" of small spatial resolution, the span of this spatial resolution is known as the filter's receptive field. In Figure 6.5, a 32 pixels wide and 32 pixels tall colored image input is convolved with multiple filters generating a cuboid, in blue, which is commonly referred as activation maps. This convolution operation, for the 1D case, is illustrated in Figure 6.6, the convolutional filter is Figure 6.6 (c) and Figures 6.6 (a) and (b) are convolutions with different strides, the green connections are where the filter connects with the input and the yellow squares (top) the result of

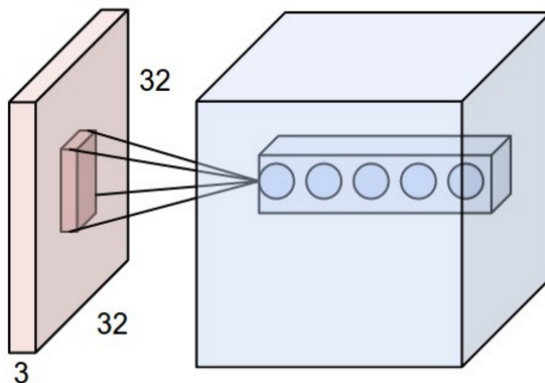


Figure 6.5. Illustration of Convolutional Layers. Taken from Stanford's CS231 Course

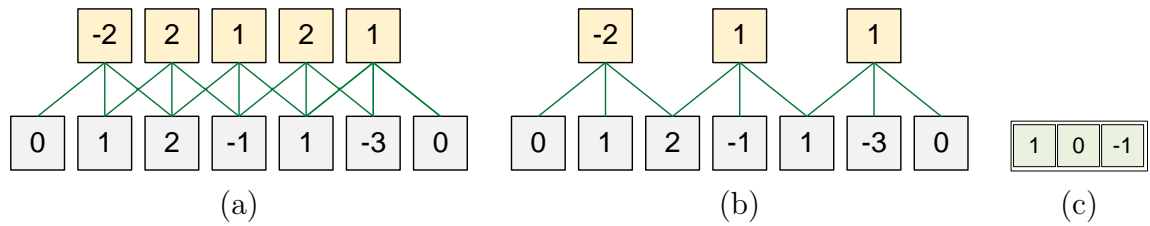


Figure 6.6. Example of the computation of a convolutional operation. (a) illustrated a convolution with a unit stride and (b) with a stride of 2, resulting in a subsampling operation. (c) is the applied convolutional filter. The output of the convolution operations are in the top row, depicted in yellow color.

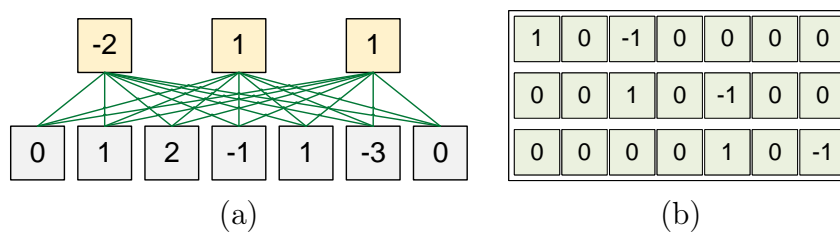


Figure 6.7. Fully Connected network that is equivalent to the convolutional one depicted in Figure 6.6(b). The weights outside the receptive field are zeroed as is seen in (b). The output of the convolution operations are in the top row, depicted in yellow color.

a dot product between them.

In fully connected layers, such as the one in a multilayer perceptron, a neural node has a weighted connection with all elements of the input. In the case of convolutional neural networks, for each position the filter is applied, it is equivalent to having a fully connected node whose weights outside the receptive field are zero. This implies that the equivalent fully connected network to a convolutional network has a number of nodes equal to the sum of elements of all activation maps. This parallel of fully connected layers and convolutional ones is illustrated in Figure 6.7.

Convolutional networks are neural networks with convolutional layers, whose weights are locally connected as opposed to fully connected. To the example of LeCun et al. [1998], convolutional networks can exploit a strong 2D local structure of images and consume less memory than fully connected layers. In addition, local features can be classified into a small number of categories (e.g., edges and corners), and convolutional networks force the extraction of these local features since they restrict the receptive field of hidden units to be shared and locally connected. The key assumptions behind these networks is that the patterns of interest are composed by smaller patterns which have a spatial independence and have a hierarchical complexity, regarding scale.

To illustrate to the reader with a computer vision background, but new to con-

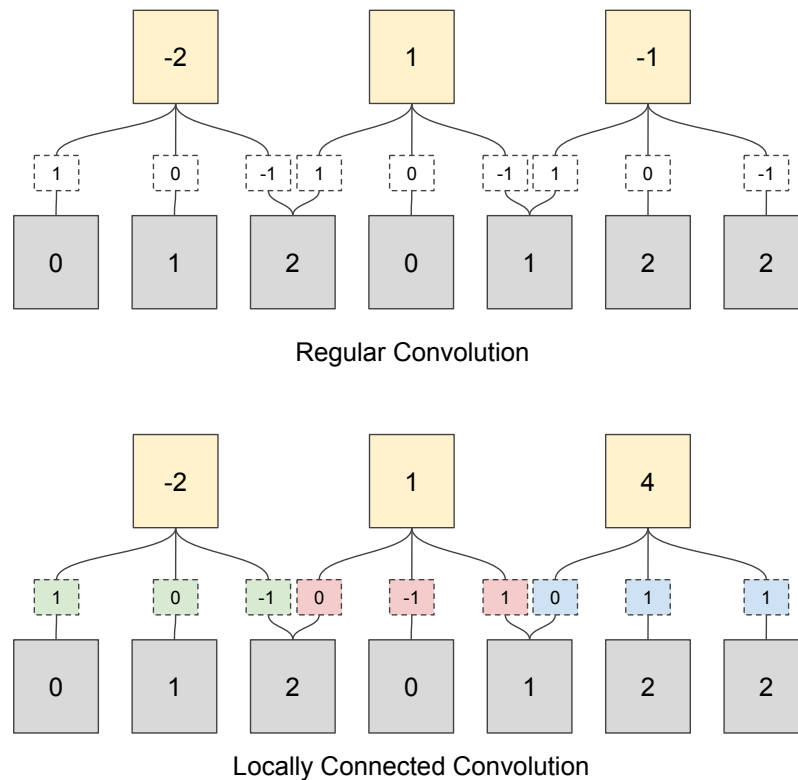


Figure 6.8. Locally Connected Convolution Compared with Regular Convolution. The input are the bottommost, gray, boxes. The output of the convolution operations are in the top row, depicted in yellow color. The filters are depicted in the middle between the input and the output. Note that the locally connected filters are different on each spatial position, depicted by different colors, green, red and blue.

volutional networks, a parallel between how the convolutions work and how mid-level features were extracted before convolutional networks is shown in Figure 6.9. Both methods learn patterns of smaller portion of the image, while in mid-level features a method such as K-means is employed to learn representative patches, with convolutional networks, gradient descent learns patterns analogous to these patches. Then, both methods compare the regions of the image to the learned patterns/patches to build a map of activations (feature map) that indicates how similar to the given pattern that region is. These activations can be used to learn more complex patterns, since, they convey information of which kind of patches are in a region, and complex patterns are composed of the patches. In the case of mid-level features, these activations were usually used to extract statistical measures, such as in a bag of words model, while in convolutional networks they can be fed to another pattern learning layer in a cascaded fashion, or fed to a classification layer.

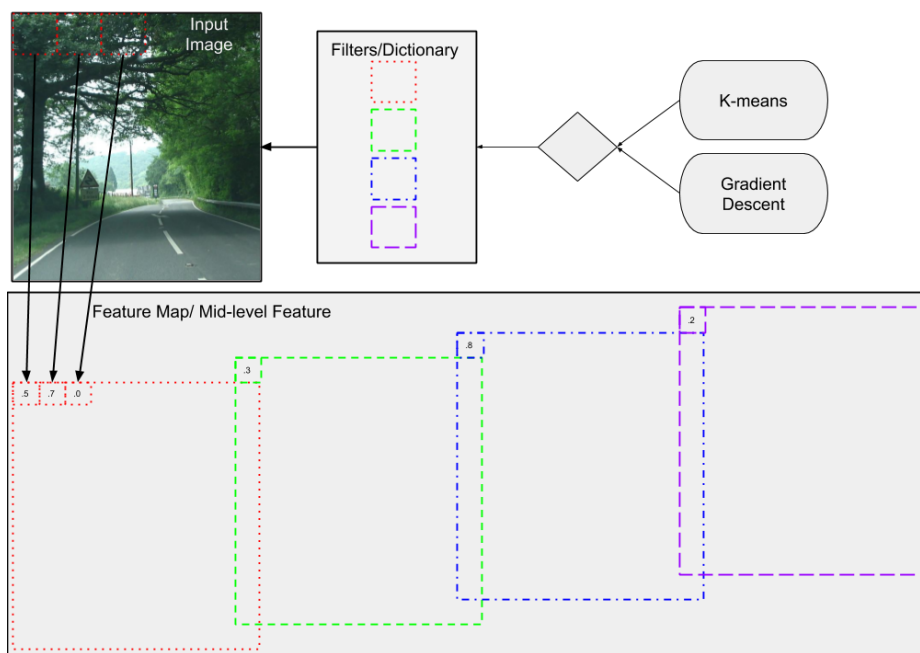


Figure 6.9. Here we show a parallel between modern convolutional networks and what was previously done in computer vision to build mid-level features, e.g. bag of visual words.

The first great breakthrough for these convolutional neural networks was the work of Krizhevsky et al. [2012] and their architecture called AlexNet, which is illustrated in Figure 6.10. By using a network with five convolutional layers, max-pooling, ReLu activation and dropout for regularization they were able to win the ImageNet competition of the year 2012. Their work is very important because they introduced ReLu, which is faster than the then used hyperbolic tangent. They also popularized dropout which is a simple solution for avoiding overfitting and they were able to show the power of these networks by winning the complex and challenging competition of ImageNet, which consists of natural images that must be classified as belonging to one of a thousand possible classes.

Later in 2014, Simonyan and Zisserman [2014b] showed the very simplistic neural network architecture, the VGG (Visual Geometry Group). It consists of only 3×3 convolutional filters and 2×2 with a 2 stride max-pooling. They also showed that three consecutive convolutional filters of size 3×3 , have an effective receptive field of size 7×7 , but with fewer weight parameters to adjust.

Also in 2014, Taigman et al. [2014] made a breakthrough in face recognition with convolutional networks. They achieved human results on the challenging Labelled Faces in the Wild dataset. Their results were mainly due to a powerful alignment

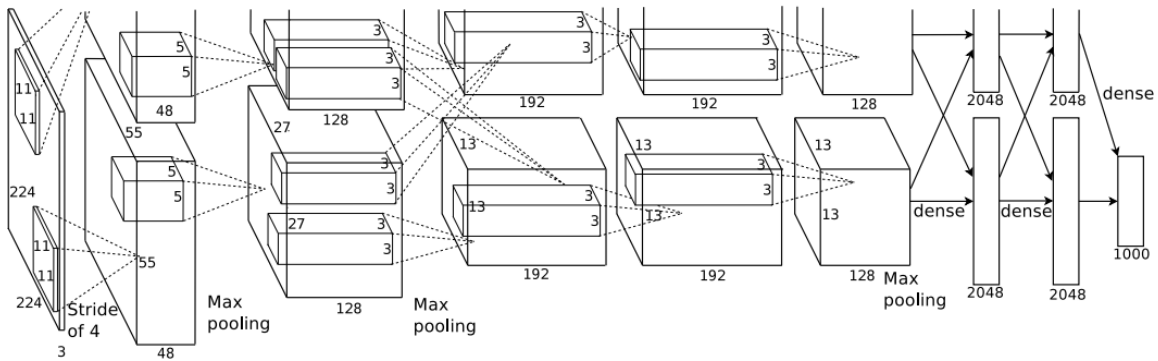


Figure 6.10. AlexNet architecture. Taken from their original work paper [Krizhevsky et al., 2012].

technique, frontalization, and the features learned by a deep model with lots of data. An interesting fact is that they used locally connected convolution, see Figure 6.8, since they postulated that, thanks to their alignment, there is no spatial independence of patterns and they are roughly bounded to a position in space. This convolution is different because its filters are not shared spatially.

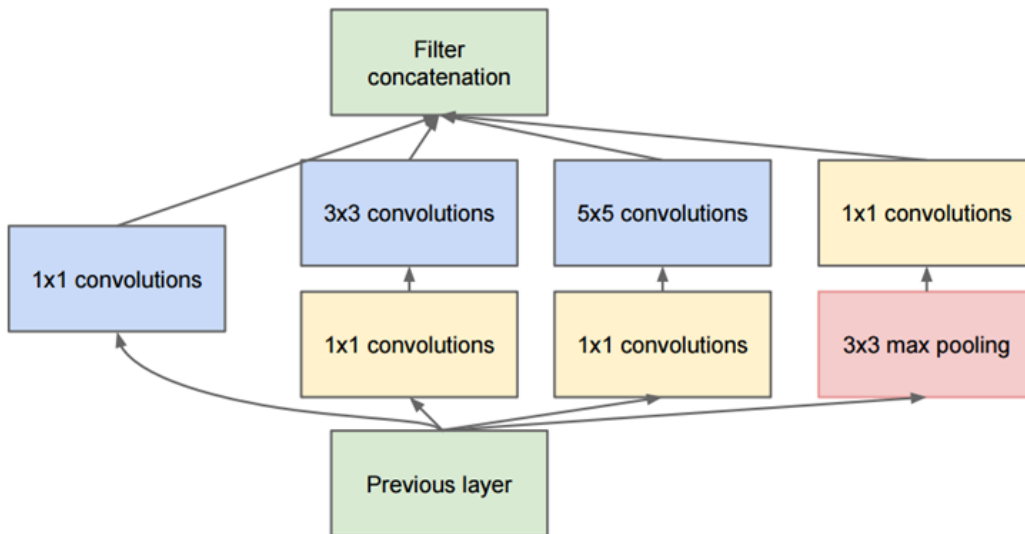


Figure 6.11. Original Inception Module. The 1×1 convolution in the beginning of the module is a compression trick in order to fit the model in the memory constraints of GPUs. This module explores patterns of different scales by using different sizes for the receptive fields of the convolutions. The image is from the original paper [Szegedy et al., 2015].

In 2015, Szegedy et al. [2015] used a novel inception module, which instead of having to choose at each layer which of a 1×1 , 3×3 or 5×5 filters or max-pooling would be employed, the inception module used all of them at once, Figure 6.11. This

inception module enabled deeper architectures, in their work they used 22 weighted layers. Because the number of channels of the input being feed to a inception module is usually large, they applied a 1×1 convolution prior to doing any other transformation in the data. This convolution works as a compression trick that lessens the cost of applying other operations. Later versions of this inception architecture also used new techniques, such as Separable Convolution [Howard et al., 2017], and global average pooling replacing the first fully connected layer, which spares lots of parameters.

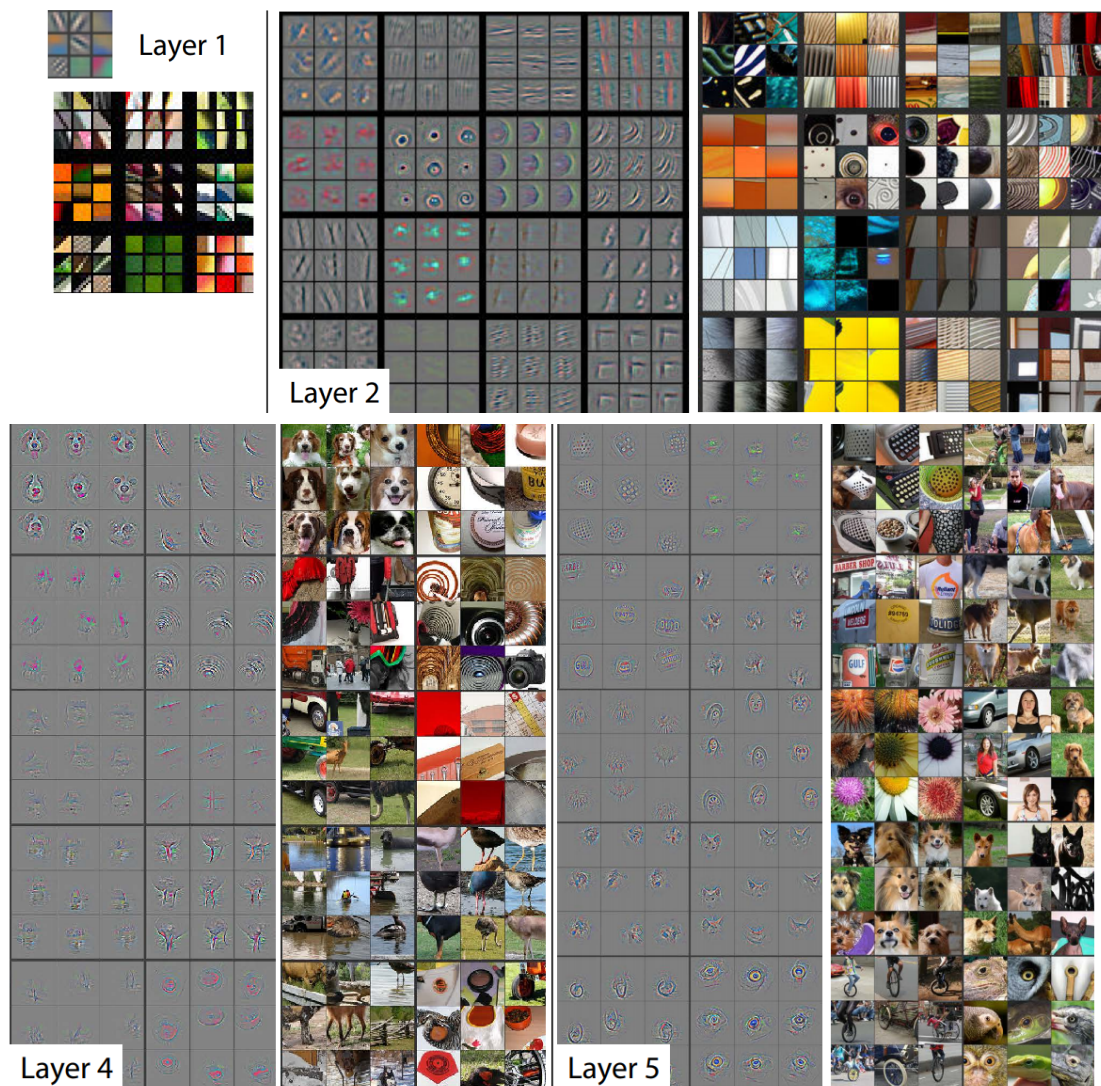


Figure 6.12. Visualization of filters for the Two initial layers, first row, and last two layers of the model from [Zeiler and Fergus, 2014]. It is possible to see edge patterns (layer 1) that together compose texture patterns (layer 2), and by the final layers objects and parts of objects. Adapted from the original work.

Understanding of a method is really important. Some methods may demonstrate great results but be really sensitive to small perturbations, or have an overfitting behav-

ior. The work of Zeiler and Fergus [2014] used deconvolutional networks to understand what each filter of each layer was learning. Their results can be seen on Figure 6.12, where a pair of features and patches that had the highest activation to the correspondent node. In the first layer the actual filters are shown, in the subsequent it is shown the pixels which had the most importance in the activation of the neural node. We can notice that the deeper layers, second row in Figure 6.12, works as detectors of parts of objects and that early layers detects edges, texture patterns and colors. It is the relation of these edges, textures, and colors that are used to compose the part detection of later layers, in turn, the relation between locally connected parts is used to find complete objects, hence the power of convolutional networks.

In 2016, He et al. [2016] proposed a model, Resnet, that had what they called skip connections which models the layers to compute the gradient of $f(X) + X$ instead of $f(X)$ only, as seen in Figure 6.13. This helps alleviate the problem of the vanishing gradient, the diminishing of its magnitude across the layers as it backpropagates, since the gradient of a sum is the sum of the gradients, this reinforces the gradient of early layers in the model. With these skip connections its models were able to have over 50 layers. By using an ensemble of these models their results on the top-5 error of imagenet was 3.57%, which was the best result on the dataset.

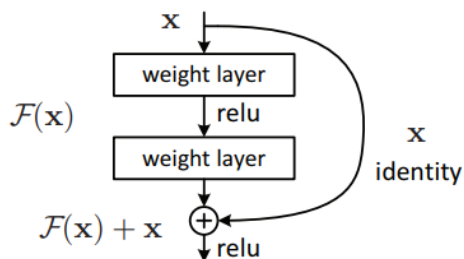


Figure 6.13. Resnet module as seen in [He et al., 2016].

6.2 Face Verification

In general, face recognition follows a four stage pipeline, detection, alignment, representation and classification, as can be seen in Taigman et al. [2014]. Many works in face recognition focus either on the representation step, the discrimination step, or both. In the face verification task, the discrimination might be employed by learning a classifier or a metric that can be used to separate same/not-same pairs. The works that tackle the discriminative problem usually focus on metric learning, which consists

of finding a transformation to another feature-space in which the euclidean distance is more discriminative than in the regular space, it is also known as Mahalanobis distance metric learning (MDML), Figure 6.14 illustrates the goal of MDML.

On problems with small available data, the feature representation problem was tackled by employing low level-feature extractor, such as Histograms of Oriented Gradients (HOG) [Dalal and Triggs, 2005] and Local Binary Patterns (LBP) [Ojala et al., 1996]. The first uses image derivatives to capture shape information while the second describes texture of pixel regions. With the popularization of deep learning and its breakthroughs in many tasks, it is now more common to learn features instead of engineering them. Although as previously said, in cases where a small amount of data is available learned features might prove unrepresentative due to overfitting.

In the next paragraphs we discuss methods that have been evaluated in the following two datasets for face verification, Labeled Faces in the Wild (**LFW**) [Huang et al., 2012] and Youtube Faces Database (**YTF**) [Wolf et al., 2011]. LFW consists of a 10 fold cross validation protocol where each fold has 600 samples with balanced labels, in *restricted protocol*, no outside data can be used, in *unrestricted protocol*, outside data can be used for training. Youtube Faces Database (**YTF**) [Wolf et al., 2011] is a dataset that has a similar setup to LFW, but instead of verifying pairs of images, pairs of videos are used. It is also a 10-fold protocol, but each fold has 500 label balanced samples.

One very successful work with respect to feature learning for face understanding is the work of Kumar et al. [2009]. They learned one single model for each one of 65 attributes (e.g., big nose, oval face), and also learned models for face parts of famous peoples which they called simile classifier. The idea behind the simile classifier is that humans can use information of is alike to person ' x ' and is distinct of person ' y ' to identify a subject. These models were learned by using amazon mechanical turk for labeling and low level features of fiducial points for representation. They used these two features as input to learn discriminative models for face verification. They achieved 85.29% accuracy on LFW (image **restricted** configuration).

Wolf et al. [2011], proposed a MDML method inspired by One Shot Similarity (OSS) [Wolf et al., 2009], which uses a collection of negatives and samples from two other classes and learn two models one for each class and predicts a confidence of one signature belonging to the other class and not to the negative one and the same to is applied to the other class. In their work, they also use a negative set which they call background. Given two sets of signatures to compare, X_1 and X_2 , they find nearest neighbors of X_1 in the background set and learn a model that separates X_1 from its nearest neighbors, it then computes the signatures of X_2 on this model. They also do

the opposite, train a model X_2 vs its nearest neighbors, and compute the signature of X_1 on this model. The combination of the response of the two models is combined to form their metric of similarity. They achieved 76.4 ± 1.8 average recognition rate on the YTF.

Regarding deep learning methods, Sun et al. [2013] proposed a deep architecture that relied on an ensemble of crop specialized convolutional networks. In their work, a pair of face images is fed as input to an ensemble of networks. Their network is depicted in Figure 6.15, M1 to M8 are pairs of images in different configuration of ordering and horizontal mirroring. To ensure diversity, each ConvNet model in the ensemble is presented with different cropped regions of the face which they found to contain discriminative information, these regions can be in color or in grayscale in order to add more diversity to the model. The L0 layer combines each group of deep ConvNet output by average pooling. The L1 and L2 layers are in charge of combining each subgroup in the lower levels and in the highest level there is a restricted boltzmann machine (RBM) that learns to separate between same and not-same pairs. Their method achieved 91.75 mean accuracy on the LFW dataset using the **unrestricted** protocol. In conclusion, they employed an end to end architecture for face-verification, but with a very complex and with very high capacity model, due to the ensemble, which makes the model prone to overfitting.

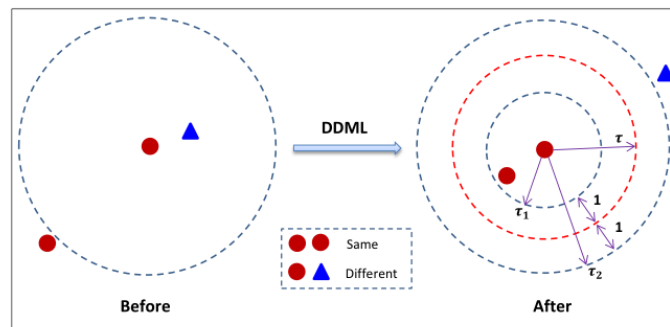


Figure 6.14. In this figure, taken from [Hu et al., 2014] it is illustrated the goal of Mahalanobis distance metric learning. It is desirable that not-same pairs that are close in the conventional euclidean space become distant in the transformed space and same pairs to be close.

Hu et al. [2014] also used deep learning models on face recognition. They employed a *siamese network* with shared weights to learn a MDML. A siamese network is composed of two shared blocks that extract features from pair of images, specially useful in verification task. Their network model, then, works as a space transformation

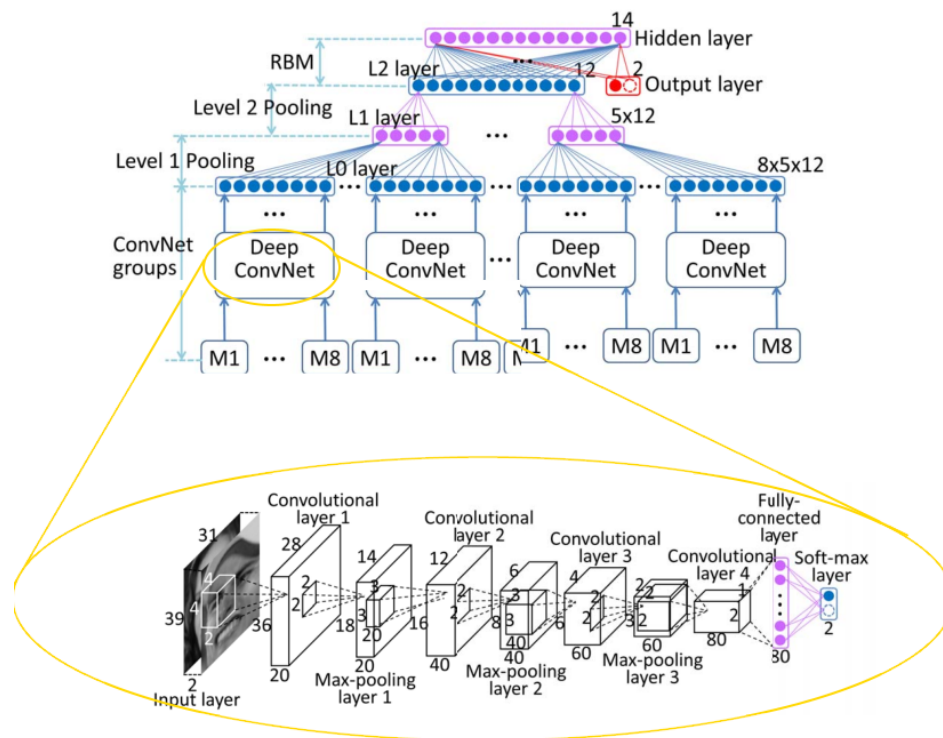


Figure 6.15. Figure adapted from [Sun et al., 2013], illustrates their network architecture for face verification. The ensemble architecture is depicted in the top row and the individual ConvNet of the ensemble depicted in the bottom row.

where the transformed features are more discriminative. Their network is composed of only fully connected layers and their input is a combination of handcrafted features. They achieved 90.68 ± 1.41 on the LFW image **restricted** protocol. Since they did not use convolutional networks in the raw images, instead used only fully connected layers on handcrafted features, their network does not learn a representation, only a transformation of the space of the handcrafted features, a MDML.

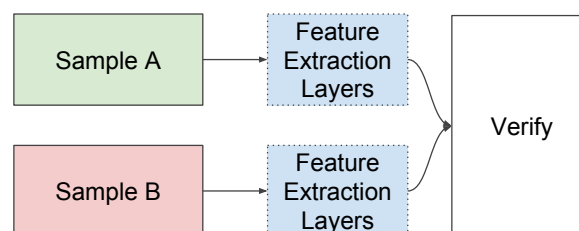


Figure 6.16. Depiction of the Siamese Network. The boxes labeled *Feature Extraction Layers* share weights and are responsible for extracting features from each sample which are compared in the block labeled *Verify*. The features are optimized to separate pair of samples.

In the same year, researchers from Facebook [Taigman et al., 2014] outperformed human results on the LFW dataset by employing a fully convolutional network that learns feature representation and is used in an ensemble of similar networks trained with different random seeds. They also learned a *siamese network*, see Figure ???. This network achieved 96.17 ± 0.38 as compared to their ensemble with a result of 97.35 ± 0.25 of mean accuracy on the **unrestricted** protocol. Another major contribution of their work and one of the pillar which led them to such great results is a face alignment technique based on 3D transformation, Figure 6.18. Although their network architecture is somewhat atypical, employing large convolutional kernels, it was a breakthrough beating human performance and was important to reinforce the strength of deep learning models. It is also important to note that they implemented a CPU-version of their model for doing inference and, using a single core Intel 2.2GHz CPU, the operator took 0.33 seconds to extract features and align an image.

Schroff et al. [2015] presented a follow-up for the *siamese network* used by the previous work. Their triplet network, as seen in Figure 6.17, receives as input a reference image, an image from the same class as the reference and another image from a different class, it then tries to find a space where samples from the same label are closer and samples from different labels are far apart. To compose triplets they select a hard positive and hard negative for each reference sample. Hard positives are 'same' samples that are distant from the reference, and the hard negatives, are 'not-same' which are closer to the reference sample. They Achieved 99.63% on lfw **unrestricted** protocol. On YouTube Faces DB it achieves 95.12%. The difference to siamese networks is that it optimizes both same and not-same cases at each gradient update, so in each update the network is being enforced to place same pairs close and not-same pairs far apart.

In the restricted setting of LFW with no outside data, the work with best results is from Ouamane et al. [2015]. They used an adaptation of Discriminant Analysis for the weakly labeled case of same/not-same pairs coupled with exponential kernel, the input of their method is the BSIF features [Kannala and Rahtu, 2012], which is inspired by LBP. They compare their method, Side Information Exponential Discriminant Analysis (SIEDA), with the linear version Side Information Linear Discriminant Analysis (SILD). they achieved a 94.63 ± 0.95 on the LFW **restricted** protocol.

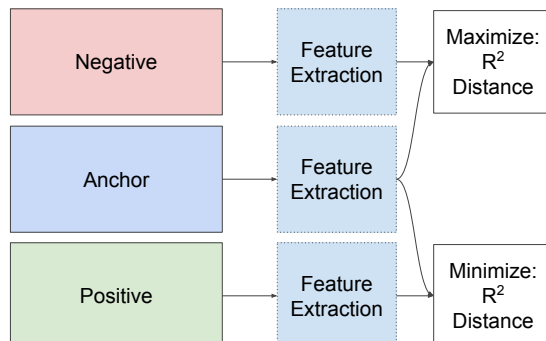


Figure 6.17. The Triplet Loss Architecture. The boxes labeled *feature extraction* are neural network layers with shared weight with one another. The goal is to learn features where samples from the same class are close and samples from different classes are far apart.

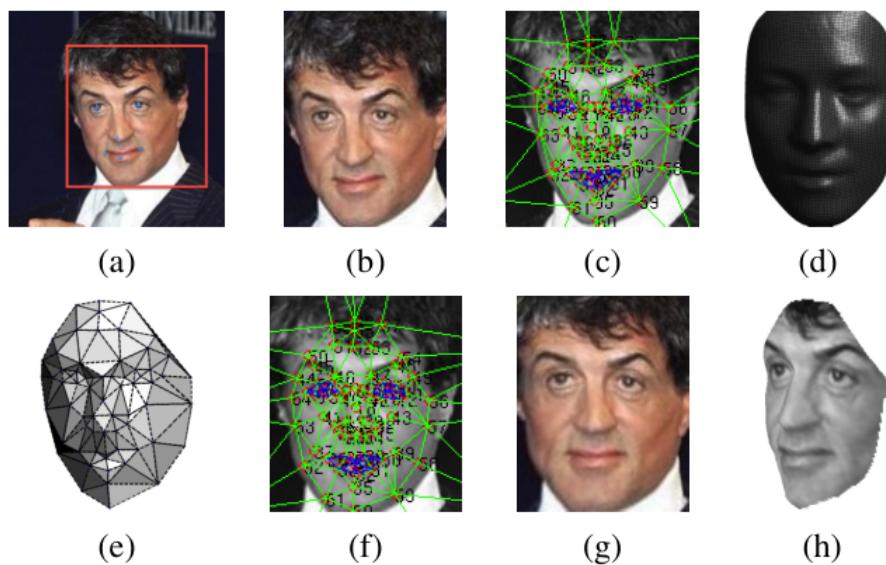


Figure 6.18. Taken from [Taigman et al., 2014]. Depicts their 3d alignment method.

Chapter 7

Transfer Learning

Since the AlexNet model [Krizhevsky et al., 2012] won the 2012 ImageNet challenge [Russakovsky et al., 2015], deep learning models have been popularized and made breakthroughs in many areas [LeCun et al., 2015], winning matches of a hard game such as Go against world champions [Silver et al., 2016], and achieving results comparable to humans in face verification [Taigman et al., 2014].

Many models which have achieved great results on ImageNet competitions have also been successfully used in datasets or tasks different than the ones they were originally trained on, circumventing part of the learning cost, or making possible to obtain good results in datasets with a small number of samples. One way this could be done is by employment of transfer learning, the act of adapting a model learned on one data to another, see Figure 7.1. Of important notice is the fact that although data augmentation could also alleviate the problem of having a small number of samples it would still require the learning of the whole model, for this reason, we believe our transfer learning approach to be less costly.

Transfer learning usually consists of replacing some of the final layers of a learned model and reconditioning the weights to new data [Simonyan and Zisserman, 2014a; Yosinski et al., 2014], as is shown in Figure 7.1. This is called fine-tuning, since the old model is being fine-tuned to the new data. This is usually done in order to suit a model to a small target domain/data avoiding overfitting of this small data by first training the model on a larger related data. The disadvantage of this is that it needs robust hardware to engage in the learning process and although lessened, the need of data for performing fine tuning is still present. Motivated by the aforementioned discussion, we explore, in this work, transfer learning approaches that would require little to no fine-tuning.

We conjecture that features associated with each layer of a sequential deep model

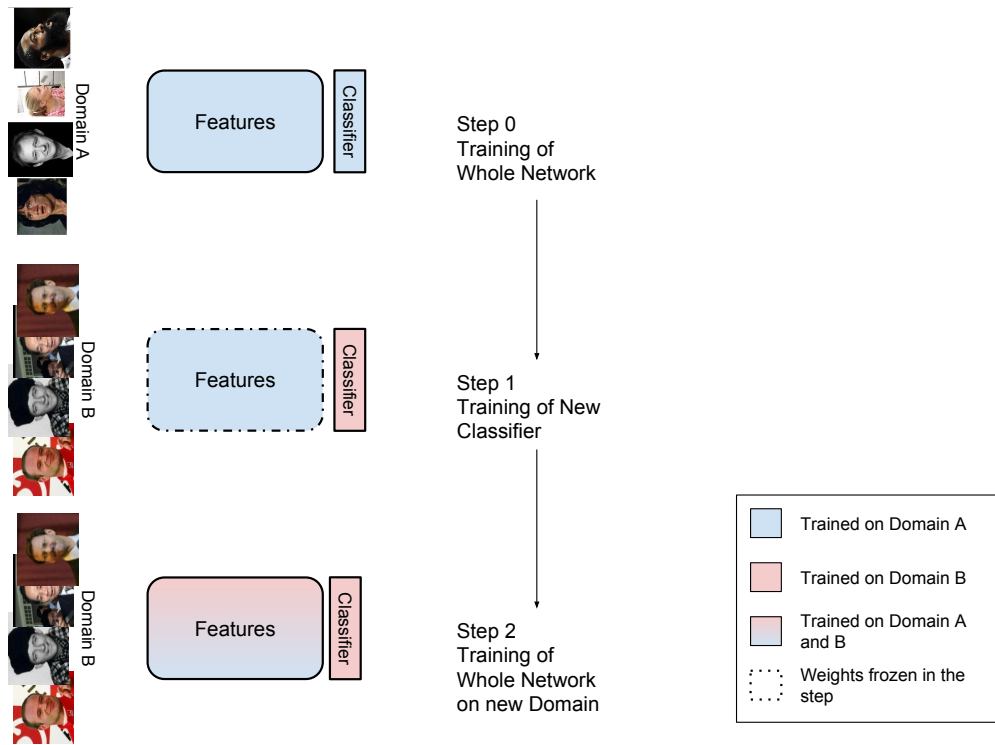


Figure 7.1. Depiction of the steps of traditional transfer learning with fine-tuning. The Classifier in step one is initialized with random weights. Step 2 is optional and usually employed to improve performance by learning some feature patterns that are specific to the new domain, as a trade-off, it could also lead to overfit.

tend to be more specific to the learned environment as its depth increases. This way, we believe that the penultimate layer, as usually done, might not be the ideal one to extract features if the new data is characteristically different than the data the model was learned on. One way that this has been approached is by using the average of features of the same image in multiple scales or crops, which adds to the computing cost, this makes the model more robust to inputs with different scales, this is done in the Inception architecture [Szegedy et al., 2015], for example.

We employ our approach to the task of face verification, which consists of given a pair of face images, determining whether they belong to the same identity or not. Verification differentiates from face identification in which given a face image it is necessary to predict the identity to which it belongs. We extract a feature vector of each face with a VGG16 model, learned on the VGGFaces dataset [Parkhi et al., 2015].

7.1 Proposed Approach

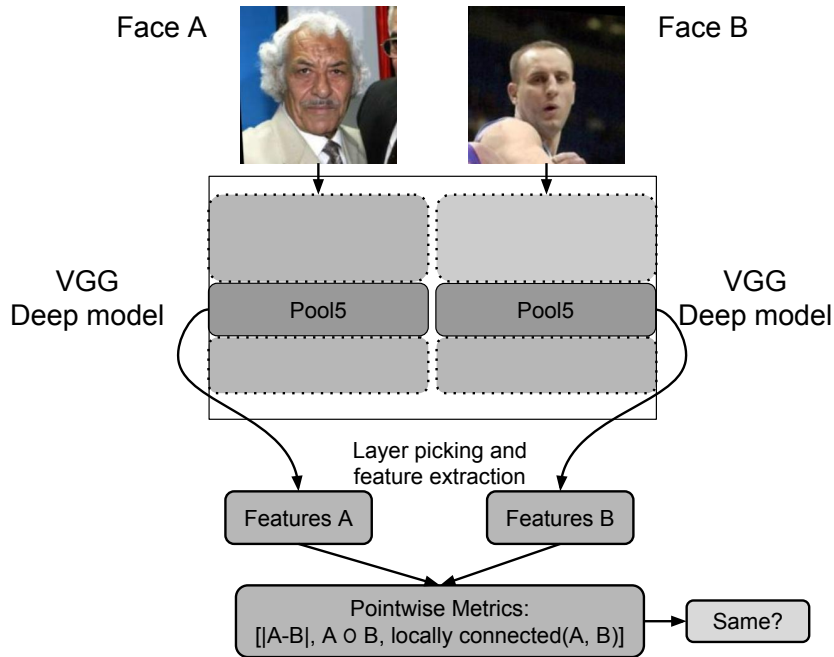


Figure 7.2. Pipeline of our Face Verification Approach.

Our method, illustrated in Figure 7.2, can be roughly described as follows. First, we carefully choose a layer from a neural network extracting its output as feature vectors for each face image, in the same way as a *siamese network*, detailed in Chapter 6. Then, the relationship between the two feature vectors is captured according to some distance metric, resulting in a new vector. The metrics employed are listed in Table 7.1, we call them handcrafted because they are not learned through the data. Finally, the resulting feature vector is presented to a classifier, which in our experiments we used the Multilayer Perceptron (MLP) Haykin [2001].

Name	Formula
χ^2	$\frac{(x-y)^2}{x+y}$
L_1	$ x - y $
<i>Sign</i>	$x \times y$
L_2	$(x - y)^2$

Table 7.1. Handcrafted distance metrics used. Where x and y are two vectors and all operations applied are pointwise, that is they also return a vector.

We use a VGG16 architecture Simonyan and Zisserman [2014b] learned on the Face Identification domain to extract features from the LFW and YTF datasets. This

architecture consists of 16 convolutional and fully connected ones layers. In general, the features are extracted from the penultimate fully connected layer Taigman et al. [2014]. However, we experiment with different layers (*pool5*, the last convolutional layer, *fc6*, the first fully connected layer and *fc7*, the penultimate layer), hypothesizing that there is an inverse correlation between generalization and layer depth.

Our hypothesis comes from two clues. The first is the fact that as max-pooling occurs it has the effect of enlarging the receptive field and then capture larger and more complex patterns, usually associated with a higher semantic level. The second is that a node of a neural network layer can be interpreted as a type of correlation to detect patterns. Therefore, with stacked layers, one layer output is in a pattern domain and the next layer will be in a pattern of patterns domain, which we believe increases complexity of the patterns and decreases generalization.

The feature vector of an image extracted by a deep model is composed of the response of the neurons of a layer of the model. Each of this response is the result of a correlation of signals after applying a nonlinear activation, the signals being the input to the layer and the pattern the neuron has learned to match. Each variable of the feature vector, then, represents the confidence that a given pattern was located by a neural node. In the image space, a two dimensional signal, there is spatial independence of patterns, i.e., an object such as a ball, can be located at different positions, however, in the feature space, the variables have more spatial dependency and it is reasonable that if a variable has large value on a sample and low value on the other, this is correlated with them being different from one another. This is a good motivation for using deep feature to compare two images, or face images, in this work.

Considering that some variables might be more important than others to discriminate a pair of samples, we experiment to use a weighted sum of the variables where the weights are learned from the data. This is done by means of a (2×1) locally connected convolutional (LCC) filter [Taigman et al., 2014], which we illustrate in Figure 7.3. These filters, different from standard convolutional filters, are not spatially shared, and for each variable of the pair of samples a different relationship in the form $x_i * w_{i,a} + y_i * w_{i,b}$ is learned, where x_i is the i -th variable from the first sample in the pair and y_i is the analogous for the second sample in the pair. We can note that the weights can therefore be learned to ignore a variable from either of the face of one of the pairs, suppose $w_{i,a} = 0$ or $w_{i,b} = 0$, to ignore the relationship altogether, $w_{i,a} = 0$ and $w_{i,b} = 0$, or to give it large importance, $w_{i,a} \gg 0$ and $w_{i,b} \gg 0$.

With the traditional convolution, the weights of a feature map are shared across different spatial positions. This implies that the weighting for one pair of variables is suitable to all the others. By using more convolutional filters, more weighting rela-

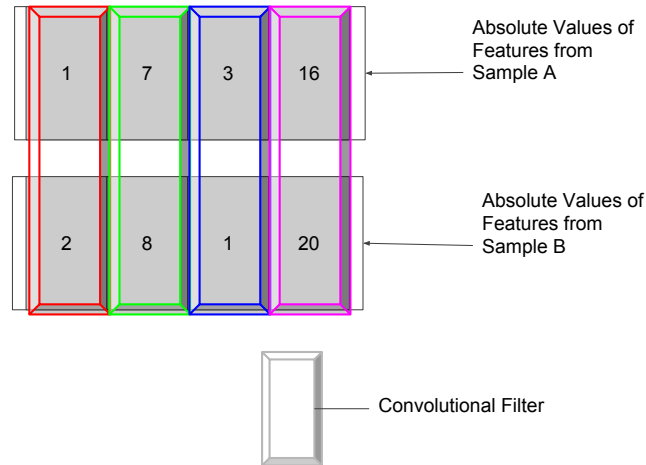


Figure 7.3. Illustration of how we compute metrics with a convolutional (locally connected) operation. Each variable is weighted by the convolutional filter, meaning it learns, in theory, which variables are more or less important.

tionships are learned. This will generate an output with size $N \times F$, where N is the length of the feature vectors and F is the number of convolutional filters. In general, this process results in memory problems which can be circumvented with a trick also used in the Inception Module [Szegedy et al., 2015]. It consists of employing another convolution with a (1×1) sized filter to compress the channels dimension after performing the layer convolution, resulting in an output of size N , the original length of the input vector.

One problem of simply applying the weighted difference between any two variables, as discussed previously, is that the weighted sum of the convolutions would have different value depending on the order they are executed, since $a - b \neq b - a$. Thus, to avoid this, the difference is followed by an absolute value layer, making the relationship indifferent to the order that a pair of variables, and consequently, faces, is presented. However, by taking the absolute value the information of the sign of the variables is lost. In the product of variables, *sign* in Table 7.1, the result of the operation will be positive if the operands have the same sign or negative if they have different signs. This apparent complementarity was the motivation that led us to test the sign metric and combine it with other metrics.

Searching for complementary information, we conducted experiments with different metrics. The χ^2 , defined by the square of differences over the sum, which is suited for histogram comparison and was used in the work of Taigman et al. [2014] to compare different feature vectors extracted from deep models. The square of differences (L_2), which is an alternative to taking the absolute value to make it indifferent to the or-

der of the variables, which also accentuates very small and very large values, this last characteristic could convey complementary information to that of the simple absolute value of differences. An overview of the metrics is listed in Table 7.1.

When comparing two faces in the verification task, this comparison yields a similarity metric, which is a numeric value. It is then necessary to choose a threshold to classify a pair of samples as belonging to the same identity or not. In our case, the similarity is constrained in the $[0, 1]$ domain, due to the output of a sigmoid neural node and we simply choose the median of the domain as the threshold, 0.5.

Chapter 8

Experimental Results

In this section, we present our experiments and results achieved. Finally, we present results on the face verification task in the Labelled Faces in the Wild (**LFW**) dataset and the Youtube Faces dataset (**YTF**) with our transfer learning method. The two datasets used and their protocols are described in detail in Chapter 6.

This section presents the results of our transfer learning approach on face verification and discusses findings and insights they led us to. We extract features of different layers and test them with different distance metrics to be employed as features of a metric learning model. We also experiment using locally connected convolutional layers to learn a weighted distance metric. In summary, we used the learned model to perform a transfer learning requiring a small amount of additional training and data.

Metric Name	Mean Accuracy	Confidence Interval _{90%}
LCC	95.15	[94.51, 95.79]
L_1	90.83	[90.34, 91.32]
<i>Sign</i>	90.75	[89.92, 91.58]
$L_1 + \textit{Sign}$	96.33	[95.83, 96.80]
χ^2	96.39	[95.70, 97.08]
$L_1 + \textit{Sign} + \text{LCC}$	96.48	[95.91, 97.05]

Table 8.1. Accuracy obtained in Labeled Faces in the Wild when using the feature map obtained from layer *pool5* with different metrics. The + sign indicates concatenation of the metric results.

Table 8.1 shows the performance of the different metrics we used. The χ^2 , which is commonly employed [Taigman et al., 2014] is able to achieve good results by itself, as in the 5th row of the Table. However, combining it with other metrics did not have a positive effect. In comparison, the L_1 and *Sign* metrics, although showing

a lower accuracy, individually, when combined showed an reasonable improvement, indicating that the information of each metric is indeed, complementary to the another. The Locally Connected (LCC) metric also obtained a good result alone (first row of Table 8.1) which we take as an example that it can be used as a learned approach for a variable measurement that does not need careful domain knowledge. Finally, combining the $L1 + Sign + LCC$ trough concatenation, yielded the best result of the metrics with 96.48% mean accuracy.

Layer Name	Mean Accuracy	Confidence Interval _{90%}	Rank
<i>pool5</i>	96.32	[95.78, 96.85]	1
<i>fc6</i>	94.58	[93.79, 95.37]	2
<i>fc7</i>	93.08	[92.11, 94.06]	3
LBP Ojala et al. [1994]	66.42	[64.98, 67.85]	4

Table 8.2. Accuracy obtained in Labeled Faces in the Wild when using the feature map obtained from different layers. We also shoe the results of the hand-crafted feature LBP in the last row. $L_1 + Sign$ metric used.

According to Tables 8.2 and 8.3, we believe that in sequential models such as the VGG16 [Simonyan and Zisserman, 2014b], the complexity and specialization of the model increases with the depth, and that is the reason we observe a worse result in the final layer *fc7* on both datasets. For the LFW, we note that the best result was with the *pool5* layer and the result was gradually deteriorating. This layer is likely to be the one with most suitable patterns for the LFW and the successive layers are to specific to the VGGFaces domain. The correlation between depth and generalizability/specification is probably a characteristic of purely sequential models such as VGG16 and we believe this corroborates with the hypothesis of Huang et al. [2017] that connecting a layer to multiple successive layers is also a form of regularization. We hypothesizes it as being a distribution of complexity across layers of all depths.

Our best result in LFW is presented in Table 8.4. It was obtained by using our best metric $L1 + Sign + LCC$ together with the concatenation of two feature vectors, one obtained from LBP [Ojala et al., 1994] and the other from *pool5* as defined previously. Our result achieved a mean accuracy of 96.65 and although this number does not outperform the one from the work of Taigman et al. [2014], a unpaired T-Test [Jain, 1990] with 90% confidence shows that the confidence interval, [-1.44, 0.4], of the difference of these two means of accuracies actually contains the zero, and therefore, they are not statistically different within this confidence.

It is important to emphasize that our best result was achieved with a simple

pipeline, as we do not use of their 3D alignment, called frontalization, and we also use a model for deep features that was trained on a smaller amount of samples. Regarding Schroff et al. [2015], our result is statistically inferior to them, they employ a Triplet Loss network that learns a projection in which same pairs are closer regarding the second order minkowski distance and not-same pairs are far apart. Our deep feature models use much less samples than their and also do not require a careful selection of the triplet to be presented to the model. Thus, with much less complexity and computation, careful selection of a layer feature map and simple metric of variables yielded statistically equivalent results.

Layer Name	Mean Accuracy	Confidence Interval _{90%}	Rank
<i>pool5</i>	91.12	[90.25, 91.99]	3
<i>fc6</i>	93.08	[92.33, 93.83]	1
<i>fc7</i>	91.32	[90.61, 92.03]	2

Table 8.3. Accuracy obtained in Youtube Faces when using the feature map obtained from different layers. $L_1 + Sign$ metric used.

In the Youtube Faces, our best result achieved a mean accuracy of 93.12, which is presented in Table 8.5. Since the Confidence Interval of the mean of the accuracies have no overlap, our result is statistically different from that of Taigman et al. [2014]. This implies that our method is statistically superior to it, since we have the better number, even though we use less samples and a simpler face alignment technique. This is a valid example that careful consideration of which layer to use in another dataset can be a valid transfer learning technique. It is important to notice, in the third row of Table 8.5 third row, that this careful selection was able to outperform the result, without the triplet-loss embedding of the model originally trained on the VGGFaces dataset.

Method Name	Mean Accuracy	Confidence Interval _{90%}
FaceNet [Schroff et al., 2015]	99.63 ± 0.09	[99.58, 99.68]
DeepFace [Taigman et al., 2014]	97.35 ± 0.25	[96.90, 97.81]
Ours	96.65 ± 0.34	[96.03, 97.27]

Table 8.4. Comparison of the state-of-the-art Results in the Labeled Faces in the Wild. A unpaired T-Test of our result and the one reported in DeepFace [Taigman et al., 2014] shows these results to not be statistically different.

Method Name	Mean Accuracy	100% - EER	Confidence Interval _{90%}
FaceNet [Schroff et al., 2015]	95.12 ± 0.39	-	[94.89, 95.35]
DeepFace [Taigman et al., 2014]	91.40 ± 1.1	91.4	[90.76, 92.04]
VGG [Parkhi et al., 2015]	91.6	92.8	-
Wolf [Wolf et al., 2011]	76.7 ± 1.8	74.7	[75.36, 77.44]
Ours	93.08 ± 0.41	92.58	[92.33, 93.83]

Table 8.5. Comparison of the state-of-the-art results in the Youtube Faces. Results of the third row are their reported results without triplet loss embedding, also, they do not report standard error.

Chapter 9

Conclusions

In this part of our work, we presented state-of-the-art results on the task of face verification with a simple method of transfer learning, careful choice of a neural network feature map. Our result outperformed Taigman et al. [2014] in one benchmark (YTF) and was statistically equivalent in another (LFW). Their work was the first work to have human comparable results on this task, and they relied on a 3D alignment, frontalization, which our simpler model does not and we also use less samples to learn the deep feature model. Finally, we believe that model fine-tuning although able to yield improved results is not a hard requirement for transfer learning and have showed experimental evidence that it is possible through, simple and computationally cheap, layer/scale selection adapt a model trained on one dataset to another.

We believe that, for future works, the transfer learning approach could be verified on different task and different networks.

Bibliography

- Akata, Z., Perronnin, F., Harchaoui, Z., and Schmid, C. (2014). Good practice in large-scale learning for image classification. *IEEE transactions on pattern analysis and machine intelligence*, 36(3):507--520.
- Avidan, S. (2007). Ensemble tracking. *IEEE transactions on pattern analysis and machine intelligence (PAMI)*, 29(2).
- Benenson, R., Omran, M., Hosang, J., , and Schiele, B. (2014). Ten years of pedestrian detection, what have we learned? In *European Conference on Computer Vision*.
- Bookstein, F. L. (1994). Partial least squares: a dose-response model for measurement in the behavioral and brain sciences. *Psychology*, 5(23):1.
- Brown, G. and Kuncheva, L. I. (2010). Good and bad diversity in majority vote ensembles. In *International Workshop on Multiple Classifier Systems*, pages 124--133. Springer.
- Cannon, W. (1922). Biographical memoir: Henry pickering bowdich. *National Academy of Sciences*, 17:181--196.
- Chollet, F. (2015). Keras. <https://github.com/fchollet/keras>.
- Cogranne, R. and Fridrich, J. (2015). Modeling and extending the ensemble classifier for steganalysis of digital images using hypothesis testing theory. *IEEE Transactions on Information Forensics and Security*, 10(12):2627--2642.
- Criminisi, A. and Shotton, J. (2013). *Decision Forests for Computer Vision and Medical Image Analysis*. Springer Publishing Company, Incorporated. ISBN 1447149289, 9781447149286.
- Dalal, N. and Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

- Dollár, P., Wojek, C., Schiele, B., and Perona, P. (2012). Pedestrian detection: An evaluation of the state of the art. *IEEE transactions on pattern analysis and machine intelligence (PAMI)*, 34.
- Freund, Y. and Schapire, R. E. (1995). A decision-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pages 23--37. Springer.
- Grothendieck, A. (1996). Résumé de la théorie métrique des produits tensoriels topologiques. *Resenhas do Instituto de Matemática e Estatística da Universidade de São Paulo*, 2(4):401--481.
- Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157--1182.
- Haykin, S. S. (2001). *Neural networks: a comprehensive foundation*. Tsinghua University Press.
- He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770--778.
- Ho, T. K. (1995). Random decision forests. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 1, pages 278--282. IEEE.
- Hofmann, T., Schölkopf, B., and Smola, A. J. (2008). Kernel methods in machine learning. *The annals of statistics*, pages 1171--1220.
- Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251--257.
- Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861.
- Hu, J., Lu, J., and Tan, Y.-P. (2014). Discriminative deep metric learning for face verification in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1875--1882.
- Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

- Huang, G. B., Mattar, M., Lee, H., and Learned-Miller, E. (2012). Learning to align from scratch. In *Conference on Neural Information Processing Systems (NIPS)*.
- Jain, R. (1990). *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. John Wiley & Sons.
- Jordao, A. and Schwartz, W. R. (2016). Oblique random forest based on partial least squares applied to pedestrian detection. In *IEEE International Conference on Image Processing (ICIP)*.
- Kannala, J. and Rahtu, E. (2012). Bsif: Binarized statistical image features. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 1363--1366. IEEE.
- Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097--1105.
- Kumar, N., Berg, A. C., Belhumeur, P. N., and Nayar, S. K. (2009). Attribute and simile classifiers for face verification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 365--372. IEEE.
- Lazebnik, S., Schmid, C., and Ponce, J. (2006). 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.
- LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436-444.
- LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278--2324.
- Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91--110.
- Marín, J., Vázquez, D., López, A. M., Amores, J., and Leibe, B. (2013). Random forests of local experts for pedestrian detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

- Martis, R. J., Acharya, U. R., and Min, L. C. (2013). Ecg beat classification using pca, lda, ica and discrete wavelet transform. *Biomedical Signal Processing and Control*, 8(5):437--448.
- McIntosh, A., Bookstein, F., Haxby, J. V., and Grady, C. (1996). Spatial pattern analysis of functional brain images using partial least squares. *Neuroimage*, 3(3):143-157.
- Nam, W., Dollár, P., and Han, J. H. (2014). Local decorrelation for improved pedestrian detection. In *Conference on Neural Information Processing Systems (NIPS)*.
- Ojala, T., Pietikainen, M., and Harwood, D. (1994). Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In *IAPR Asian Conference on Pattern Recognition (IAPR)*, volume 1, pages 582--585. IEEE.
- Ojala, T., Pietikäinen, M., and Harwood, D. (1996). A comparative study of texture measures with classification based on featured distributions. *Pattern recognition*, 29(1):51--59.
- Ouamane, A., Bengherabi, M., Hadid, A., and Cheriet, M. (2015). Side-information based exponential discriminant analysis for face verification in the wild. In *Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on*, volume 2, pages 1--6. IEEE.
- Parkhi, O. M., Vedaldi, A., and Zisserman, A. (2015). Deep face recognition. In *British Machine Vision Conference*.
- Qiu, Q. and Sapiro, G. (2013). Learning transformations for classification forests. volume abs/1312.5604.
- Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.
- Rosipal, R. and Krämer, N. (2006). Overview and recent advances in partial least squares. In *in Subspace, Latent Structure and Feature Selection Techniques, Lecture Notes in Computer Science*, pages 34--51. Springer.
- Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211--252.

- Sánchez, J., Perronnin, F., Mensink, T., and Verbeek, J. (2013). Image classification with the fisher vector: Theory and practice. *International Journal of Computer Vision*, 105(3):222--245.
- Schroff, F., Kalenichenko, D., and Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815--823.
- Schwartz, W., Kembhavi, A., Harwood, D., and Davis, L. (2009). Human Detection Using Partial Least Squares Analysis. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.
- Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484--489.
- Simonyan, K. and Zisserman, A. (2014a). Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568--576.
- Simonyan, K. and Zisserman, A. (2014b). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- Sun, Y., Wang, X., and Tang, X. (2013). Hybrid deep learning for face verification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1489--1496.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1--9.
- Taigman, Y., Yang, M., Ranzato, M., and Wolf, L. (2014). Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701--1708.
- Takemura, A., Shimizu, A., and Hamamoto, K. (2010). Discrimination of breast tumors in ultrasonic images using an ensemble classifier based on the adaboost algorithm with feature selection. *IEEE transactions on medical imaging*, 29(3):598--609.

- Uzair, M., Mahmood, A., and Mian, A. (2015). Hyperspectral face recognition with spatio-spectral information fusion and pls regression. *IEEE Transactions on Image Processing (TIP)*, 24(3):1127--1137.
- Valera, M. and Velastin, S. A. (2005). Intelligent distributed surveillance systems: a review. *IEE Proceedings-Vision, Image and Signal Processing*, 152(2):192--204.
- Wang, X., Han, T. X., and Yan, S. (2009). An HOG-LBP human detector with partial occlusion handling. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.
- Wold, H. (1985). Partial least squares. *Encyclopedia of statistical sciences*.
- Wold, S., Esbensen, K., and Geladi, P. (1987). Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37--52.
- Wolf, L., Hassner, T., and Maoz, I. (2011). Face recognition in unconstrained videos with matched background similarity. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 529--534. IEEE.
- Wolf, L., Hassner, T., and Taigman, Y. (2009). The one-shot similarity kernel. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 897--902. IEEE.
- Woźniak, M., Graña, M., and Corchado, E. (2014). A survey of multiple classifier systems as hybrid systems. *Information Fusion*, 16:3--17.
- Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320--3328.
- Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *Proceedings of the IEEE European Conference on Computer Vision (ECCV)*, pages 818--833. Springer.