Micael Cabrera Carvalho

Transfer Schemes for Deep Learning in Image Classification

Esquemas de Transferência para Aprendizado Profundo em Classificação de Imagens

Campinas

2015

UNICAMP

UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

Micael Cabrera Carvalho

Transfer Schemes for Deep Learning in Image Classification

Esquemas de Transferência para Aprendizado Profundo em Classificação de Imagens

Supervisor: Prof. Dr. Eduardo Alves do Valle Junior

Co-supervisor: Dr. Sandra Eliza Fontes de Avila

Master's dissertation presented to the Post Graduate Program of the School of Electrical and Computer Engineering of the University of Campinas to obtain a Master's degree in Electrical Engineering, in the area of Computer Engineering.

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas para obtenção do título de Mestre em Engenharia Elétrica, na área de concentração Engenharia de Computação.

This volume corresponds to the version of the dissertation submitted to the examining board by Micael Cabrera Carvalho, under the supervision of Prof. Dr. Eduardo Alves do Valle Junior and Dr. Sandra Eliza Fontes de Avila

Este exemplar corresponde à versão da dissertação apresentada à banca examinadora pelo aluno Micael Cabrera Carvalho, sob orientação de Prof. Dr. Eduardo Alves do Valle Junior e Dr^{a}. Sandra Eliza Fontes de Avila

Campinas

2015

Agência de fomento: Não se aplica
Nº processo: Não se aplica

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Elizangela Aparecida dos Santos Souza - CRB 8/8098

C253t

Carvalho, Micael Cabrera, 1991-
Transfer schemes for deep learning in image classification / Micael Cabrera Carvalho. – Campinas, SP : [s.n.], 2015.

Orientador: Eduardo Alves do Valle Junior.
Coorientador: Sandra Eliza Fontes de Avila.
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Aprendizado de máquina. 2. Redes neurais artificiais. 3. Melanoma. 4. Imagem. I. Valle Junior, Eduardo Alves do,1978-. II. Avila, Sandra Eliza Fontes de. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Esquemas de transferência para aprendizado profundo em classificação de imagens
Palavras-chave em inglês:
Machine learning
Artificial neural networks
Melanoma
Image
Área de concentração: Engenharia de Computação
Titulação: Mestre em Engenharia Elétrica
Banca examinadora:
Eduardo Alves do Valle Junior [Orientador]
William Robson Schwartz
Fernando José Von Zuben
Data de defesa: 01-07-2015
Programa de Pós-Graduação: Engenharia Elétrica

COMISSÃO JULGADORA - TESE DE MESTRADO

Candidato: Micael Cabrera Carvalho

Data da Defesa: 1 de julho de 2015

Título da Tese: "Transfer Schemes for Deep Learning in Image Classification (Esquemas de Transferência para Aprendizado Profundo em Classificação de Imagens)"

Prof. Dr. Eduardo Alves do Valle Junior (Presidente): Eduardo Valle

Prof. Dr. William Robson Schwartz: [illegible]

Prof. Dr. Fernando José Von Zuben: Fernando Jose Von Zuben

Abstract

In Computer Vision, the task of classification is complex, as it aims to identify the presence of high-level categories in images, depending critically upon learning general models from a set of training samples. Deep Learning (DL) for visual tasks usually involves seamlessly learning every step of this process, from feature extraction to label assignment. This pervasive learning improves DL generalization abilities, but brings its own challenges: a DL model will have a huge number of parameters to estimate, thus requiring large amounts of annotated data and computational resources. In this context, transfer learning emerges as a promising solution, allowing one to recycle parameters learned among different models. Motivated by the growing amount of evidence for the potential of such techniques, we study transfer learning for deep architectures applied to image recognition. Our experiments are designed to explore the internal representations of DL architectures, testing their robustness, redundancy and precision, with applications to the problems of automated melanoma screening, scene recognition (MIT Indoors) and object detection (Pascal VOC). We also take transfer learning to extremes, introducing Complete Transfer Learning, which preserves most of the original model, showing that aggressive transfer schemes can reach competitive results.

Keywords: transfer learning, deep learning, feature redundancy, deep features, automated melanoma screening.

Resumo

Em Visão Computacional, a tarefa de classificação é complexa, pois visa a detecção da presença de categorias em imagens, dependendo criticamente da habilidade de aprender modelos computacionais generalistas a partir de amostras de treinamento. Aprendizado Profundo (AP) para tarefas visuais geralmente envolve o aprendizado de todos os passos deste processo, da extração de características até a atribuição de rótulos. Este tipo pervasivo de aprendizado garante aos modelos de AP maior capacidade de generalização, mas também traz novos desafios: um modelo de AP deverá estimar um grande número de parâmetros, exigindo um imenso conjunto de dados anotados e grandes quantidades de recursos computacionais. Neste contexto, a Transferência de Aprendizado emerge como uma solução promissora, permitindo a reciclagem de parâmetros aprendidos por modelos diferentes. Motivados pela crescente quantidade de evidências para o potencial de tais técnicas, estudamos de maneira abrangente a transferência de conhecimento de arquiteturas profundas aplicada ao reconhecimento de imagens. Nossos experimentos foram desenvolvidos para explorar representações internas de uma arquitetura profunda, testando sua robustez, redundância e precisão, com aplicações nos problemas de rastreio automático de melanoma, reconhecimento de cenas (MIT Indoors) e detecção de objetos (Pascal VOC). Também levamos a transferência a extremos, introduzindo a Transferência de Aprendizado Completa, que preserva a maior parte do modelo original, mostrando que esquemas agressivos de transferência podem atingir resultados competitivos.

Palavras-chave: transferência de aprendizado, aprendizado profundo, representações redundantes, descrições profundas, rastreio automático de melanoma.

Contents

Acknowledgements

First and foremost, I would like to thank my parents, Jairo and Elza, for their unconditional support and patience. They have been the ones to help and motivate me when I needed the most, sacrificing their own time and resources to prioritize my education. My brother, Faister, also had an important part in this project, spending many hours helping me. I will be eternally grateful for you three.

I would also like to express my deep gratitude (pun intended) to my supervisors Prof. Eduardo Valle and Sandra Avila, possessors of brilliant minds, whose precious lessons I will most certainly remember. Our meetings always had a tad of fun, making everything easier and uncomplicated. My scientific trajectory was greatly influenced by their invaluable advices, which were often able to change my perspective about difficult decisions, making them look less difficult.

Part of this work was developed at the *Laboratoire d'Informatique de Paris 6* (LIP6), where I had the opportunity to work with great researchers. I am specially thankful for the help of Marion Chevalier, Thibaut Durand, Prof. Nicolas Thome and Prof. Matthieu Cord. Professors Cord and Thome made fundamental contributions to the experiments described in Chapter 4, offering their inestimable insights and guiding the progress of many experiments. Marion saved me a lot of trouble during my staying in France. She and Thibaut also gave me excellent french lessons, teaching me things that I can't imagine learning from a book and providing content for hours of laugh.

During my years at UNICAMP, I had the opportunity to get to know wonderful people. I would like to thank David Kurka for the hours we spent discussing fascinating philosophical questions, Suelen Mapa for many interesting challenges, Saullo Haniel for his advices, Alan Godoy for his political insights, Clarissa Loureiro for her never ending positiveness and strength and Raul Rosa for his example, standing up for others. Most of my supervisor-siblings also had important roles during the development of this work, such as Eliezer Silva, Paul Hidalgo, Michel Fornaciali, Jomara Bindá, Pedro Tabacof and Eduardo Seiti. I am specially thankful for Michel, who saved me lots of time and also contributed during the development of this work with many suggestions. I am also grateful for my friends Tabata Barbam, Larissa Viana, Renato Tonieta, João Zanini and Marcos Campos, who supported me since the beginning.

Finally, I would like to thank RECOD, LMCAD, Amazon Web Services and Microsoft Azure for providing computational resources during part of this work. I am very grateful for the scholarship provided by Samsung Eletrônica da Amazônia, which was undoubtedly essential for the development of this project, and for the Santander International Mobility grant, which made possible our cooperation with LIP6.

List of Figures

List of Tables

List of Acronyms and Abbreviations

AI	Artificial Intelligence
ANN	Artificial Neural Network
AUC	Area Under the {ROC} Curve
BoVW	Bag-of-Visual-Words
BoW	Bag-of-Words
CNN	Convolutional Neural Network
CTL	Complete Transfer Learning
DCNN	Deep Convolutional Neural Network
DL	Deep Learning
DNN	Deep Neural Network
GPGPU	General-Purpose Graphics Processing Unit
GPU	Graphics Processing Unit
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
mAP	Mean Average Precision
MLP	Multilayer Perceptron
RBF	Radial Basis Function
ReLU	Rectified Linear Unit
ROC	Receiver Operating Characteristic
SPM	Spatial Pyramid Matching
VOC	Visual Object Classes

List of Algorithms

1 Introduction

Artificial Intelligence (AI) aims at studying and designing intelligent entities. With strong relations to Computer Vision and other fields, AI's object of study is the intelligent behavior present in humans and animals [Russel and Norvig, 2010]. As a subject, it poses many exciting and difficult questions, not only technical but also philosophical (e.g., would finding a logical explanation for our most human feature make us "less human"?). Whitby [2008] explains that the intellectual rewards are even more exciting, as AI is pursuing scientific comprehension for one of the most tough questions we could pose about ourselves or the world; guiding us at the beginning of a journey for the inner, in which fundamental questions of being a thinking entity are investigated.

In Computer Vision, the task of *image classification* addresses the identification of the presence of a category in a given image, a complex task, demanding the ability to learn a general model from a set of training samples. Traditionally, this process starts by extracting numerical descriptions (features) for each image, and then follows by training a classifier using an annotated set of images.

The performance of classification depends critically on the features used as input to the classifier, since it can only learn from the information preserved in such features. Because many classifiers (e.g. Support Vector Machines and Artificial Neural Networks) are known to converge in probability to the correct model [Hammer and Gersmann, 2003; Haykin, 2009], the quest for representations able to preserve discriminability while ignoring noise will be often the most challenging in a classification system.

The Bag-of-Visual-Words (BoVW) model [Sivic and Zisserman, 2003] was, until recently, the most successful scheme to describe images for classification. It draws inspiration from textual information retrieval, in which the Bag-of-Words (BoW) model describes textual documents by the frequency of the words they contain [Baeza-Yates and Ribeiro-Neto, 1999], while ignoring the structure of the text. In the same spirit, the standard BoVW model describes visual documents by the frequency of local features (e.g. SIFT [Lowe, 2004]) reinterpreted as visual "words", while ignoring their spatial configuration. The visual words come from a visual "vocabulary" or *codebook*, which represents a quantization of the local-feature space. The BoVW model creates, thus, features of higher level from the local features [Boureau et al., 2010]. These features are fed to a supervised classifier that learns and decides on the labels.

In contrast to that model, Deep Learning (DL) networks are artificial neural networks with many layers and a huge number of parameters. For visual tasks, DL usually involves learning every step from feature extraction to supervised classification in a unified scheme. This pervasive learning gives DL more generalizing capacity, but creates its own challenges: a DL model will have a huge number of parameters to estimate, thus requiring large amounts of annotated data.

The advent of gigantic annotated image datasets, such as ImageNet [Russakovsky et al., 2015], allowed for advocates of the DL models to overcome many problems, while small datasets remained largely insufficient to estimate the millions of parameters involved in DL architectures [Bengio, 2009; Bengio and LeCun, 2007]. This is easily perceived in the task of Computer-Aided Diagnosis, since annotated datasets for medical imagery are, due to the cost of acquiring and annotating the data, almost always small (only a few hundred to a few thousand examples). This poses a serious challenge for DL, that generally uses hundreds of thousands (up to millions) of annotated samples for training, mainly due to the impossibility of learning, in a deep schema, different representation levels from such small datasets.

When one does not have many data nor the computational resources needed for DL, *transfer learning* emerges as a promising solution, as it allows transferring knowledge between models in order to reduce the demand for several labeled examples.

In this work we take transfer learning to extremes, showing that feature maps can be extracted from the last layer of a Convolutional Neural Network (CNN) in a way that it is still possible to achieve good results on diverse tasks. We have reason to believe that the preservation of most of the original network, allied to its application in tasks that suffer from lack of annotated data, lead to competitive performance. We also offer an extensive analysis of the information redundancy in the internal network representations, suggesting that such models could be strongly simplified by removing unnecessary precision and costly internal procedures. To the best of our knowledge, this has not been attempted before.

1.1 Our Approach and Contributions

Our main goal is to further study deep architectures and their applicability to image recognition through transfer schemes, as the piling amount of evidence for the potential of such techniques, explored by the scientific community, is starting to grow. We're particularly interested in the problem of small datasets. For this purpose, we've started from a problem that suffers from the lack of annotated data, the automated melanoma screening, using transfer learning as our main tool.

The objectives of this work are as follows:

- Evaluate the applicability of Deep Convolutional Neural Networks (DCNNs) to small datasets by better understanding the network internal representations;
- Explore transfer learning as a means to alleviate the need of very large training sets for DCNNs;
- Extend the possibilities of classification through transfer learning by analyzing and proposing different approaches.

With that in mind, we have formulated three hypotheses: (1) Unusual transfer schemes, such as the use of the last layers of a DCNN, may achieve competing results, specially when applied to small datasets, due to their smaller dimensionality; (2) The internal representations of such networks may contain unnecessary precision or information redundancy (similar to neural redundancy in animals, for recovering from brain damage), possibly suggesting that they can be compressed; and (3) the application of high-level representations, inspired by techniques such as Object Bank [Li et al., 2010] and Spatial Pyramid Matching [Lazebnik et al., 2006] (SPM), may be used during the process of the transfer for increasing the robustness of the features.

Finally, the contributions of this project are:

- A novel approach for applying transfer learning to small datasets;
- State-of-the-art results for automated melanoma screening;
- An extensive analysis of the network internal representations.

1.2 Outline

Chapter 2 - Literature Review starts with a high-level introduction to topics in classification of visual data (Section 2.1), including the task of classification, the Bag-of-Visual-Words model and Support Vector Machines. Section 2.2 briefly introduces the contents of Artificial Neural Networks strongly related to this work, focusing on the artificial neuron and convolutional neural networks. In Section 2.3 we talk about some successful deep neural networks, including recent frameworks for implementing and running deep neural networks. Transfer Learning is explored and explained in Section 2.4, where we examine the basic concepts behind classical transfer learning and initiatives for improving the quality of the transferred features. In Section 2.5 we discuss the problem of melanoma screening, which motivated the outset of this work, showing the current state-of-the-art and methods used in the field. Finally, in Section 2.6, we offer a light discussion on some of these topics, arguing on problems in the BoVW model, how deep learning can offer better strategies and why our experiments are important for those fields.

Chapter 3 - Complete Transfer Learning describes our approach to the problem of automated melanoma screening (Section 3.1), using the network final layers for the transfer process. Our experimental setup, detailing the choices of datasets, architectures and configurations for the experiments, is discussed in Section 3.2. The results of our experiments, shown in Section 3.3, are very promising, but the comparison with the state-of-the-art is problematic due to differences in the datasets and methods. We also offer comparative results in two different datasets: MIT Indoors and Pascal VOC 2007, showing that the complete approach can be advantageous in adverse situations.

Chapter 4 - Internal Representations explores the representations constructed by deep models, and their use for transfer learning. In Section 4.1, we propose six experiments, which (1) reduce the dimensionality of the feature vectors (Section 4.3); (2) decrease the precision of the feature representations (Section 4.4); (3) quantize their values and analyze the impact of positiveness and negativeness (Section 4.5); (4) combine two of the previous experiments (Section 4.6); (5) apply fusion methods in an attempt to improve the quality of the representation (Section 4.7) and (6) adopt an approach inspired by high level methods for obtaining discriminative features (Section 4.8). Our results reveal strong redundancy in deep representations, suggesting that aggressive compression methods may be used with such architectures in different scenarios.

Chapter 5 - Conclusions synthesizes our contributions, showing the impacts of this work in the fields of Automated Melanoma Screening and Transfer Learning. We also provide open questions and guidelines for future work, since our results suggest that few changes in some experiments could improve the effects we have observed.

2 Literature Review

In the last few years, the machine learning community advanced the state-of-the-art for several classification and localization challenges. This was possible mainly due to the successful application of very deep architectures in classification challenges, such as the proposal of Krizhevsky et al. [2012].

Not surprisingly, the results obtained through such methods gained the attention of new researchers, since they suggested changing the standard classification paradigm for large scale problems, discussed in Section 2.1, to a more unified scheme.

Artificial Neural Networks (ANN), introduced in Section 2.2, are connexionist models inspired by the human brain, and were already under exploration since the beginning of the studies in neurocomputing. They are behind some of the most famous deep architectures, shortly presented in Section 2.3. We limit our scope, therefore, to deep architectures related to ANN and to image classification problems, which are most central to this work.

Because the models created by such huge architectures were expected to be very robust, the application of transfer learning (discussed in Section 2.4) was promising, as the knowledge acquired by them could be effectively used for problems without enough samples for training a deep network. Results obtained by the use of transfer learning, as the ones presented by Razavian et al. [2014] and Chatfield et al. [2014], inspired some of the approaches of this work, and are among our recommended readings.

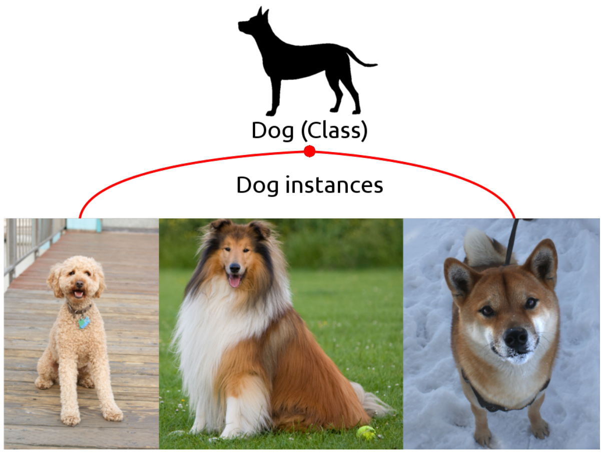
Seeking further reduction in the amount of parameters to be learned, we have adopted the task of automating the melanoma screening, briefly introduced in Section 2.5, which suffers from the lack of enough annotated data for a deep approach. The experiments proposed in Chapter 3 are strongly related to this subject.

Finally, in Section 2.6, we offer a discussion connecting some of these topics, and how they contributed to the decisions made for our experiments.

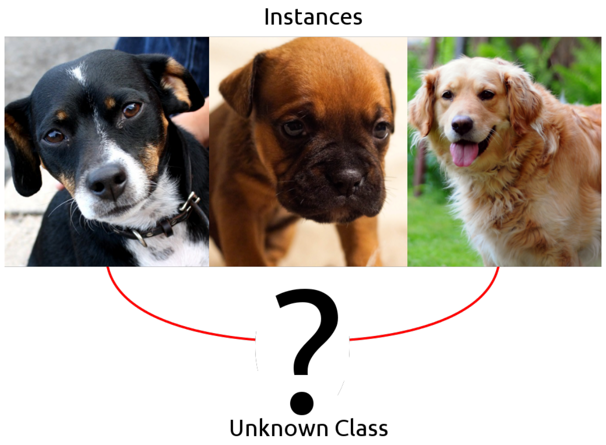
2.1 Classification of Visual Data

In Computer Vision, the task of *image classification* addresses the identification of the presence of a category in a given image, a complex task, demanding the ability to learn a general model from a set of training samples.

When trying to understand the real world, computers usually cannot gather all of the



(a) The dog class and some of its instances



(b) Instances from an unknown class

Figure 1 – The construction of a general model for identifying a given class is a complex task, demanding the ability to identify objects of interest while ignoring background and noise in the image. Figure 1a shows the dog class, and some of its instances. Figure 1b exemplifies the task of learning *what is a dog*, from a set of images limited by camera perspective and other factors.

characteristics of an individual due to many reasons (e.g. occlusion and perspective). Because of this lack of information, learning algorithms have to be able to estimate the class of a given set of instances by trying to find common characteristics in their limited description[1]. An example of such task is shown in Figure 1, where Figure 1a shows the ground-truth, with the originating class (dog) and some of its instances, and Figure 1b exemplifies the task of learning *what is a dog*.

Until very recently, the typical classification procedure for visual data was composed of 4 steps: (1) feature extraction for the training set; (2) supervised/unsupervised learning using the training data; (3) feature extraction for the test set; and (4) classification of the test data [Avila, 2013]. This procedure was used for both *online* (with a stream of data as test set) and *offline* (with a pre-defined test set) classification, and is exemplified in Figure 2.

The steps of feature extraction (1) and (3) could be performed with the use of hand-crafted descriptors, which can be divided into two groups: *low-level* descriptors focus on capturing local invariances and correlations directly from the image; and *mid-level* descriptors usually work on combining low-level features in an information-preserving manner. This hierarchical organization allows for features to become increasingly more meaningful and specialized, but each step also forces part of the information to be strongly compressed or lost.

[1] In the scope of this work, each instance is described by a picture, containing noise (e.g. background and other objects) and limited by the camera perspective and the pose of the subject.

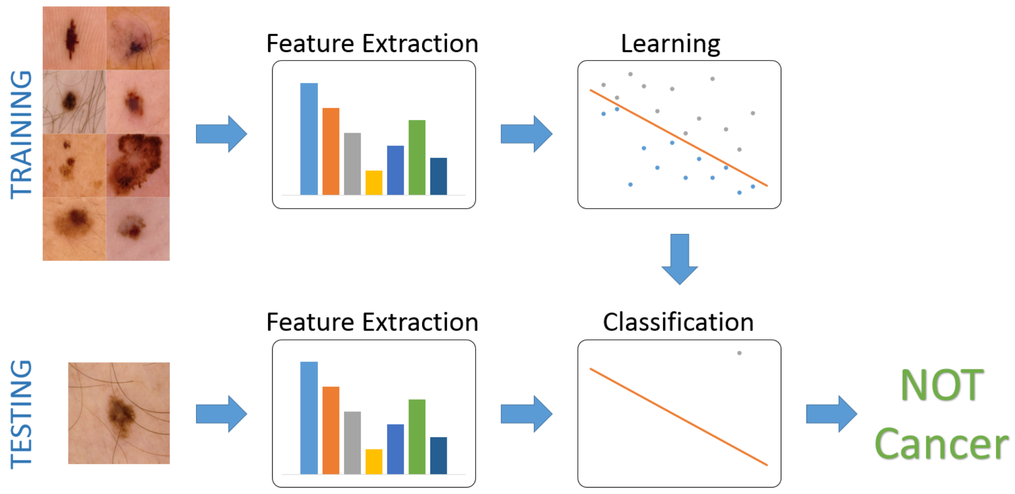


Figure 2 – The typical classification procedure was, until recently, composed of 4 steps: (1) feature extraction for the training set; (2) supervised/unsupervised learning using the training data; (3) feature extraction for the test set; and (4) classification of the test data.

The process of describing an image with a low-level descriptor usually involves the use of an *interest point detector* to select the regions we want to represent (exemplified in Figure 3b) or dense sampling, in which a grid of regions (possibly overlapping) is selected to be described (exemplified in Figure 3c). The adoption of such strategies is usually associated with the improvement of the robustness of the representations, but they also cause an increase in the size of the final representation, since many regions may be selected for a single image.

After extracting the low-level description of the image, a mid-level representation can be constructed. A common approach for such is to resort to the information retrieval's Bag-of-Words, in which a set of words is chosen to compose a dictionary, and the description of a text is the occurrence of such words in the dictionary. In the Bag-of-Visual-Words (BoVW), however, we have *visual words* (in most cases, the low-level description) and a *visual dictionary*. There are many ways to choose the words that will compose the dictionary, like random sampling and k-means clustering[2], and many ways to re-describe the image with respect to words contained in the dictionary (coding stage)[3].

The final step in the learning process is to use the mid-level description to train a classifier. Although a wide variety of classifiers have been proposed by the scientific community, we are going to focus on two types: Artificial Neural Networks (discussed in the next section) and Support Vector Machines (SVM).

2 For more information on k-means clustering, we recommend the reader to refer to the book of Duda et al. [2000], subsection 10.4.3.

3 We recommend the Ph.D. thesis of Avila [2013], subsection 2.2.3, for extended details on the BoVW coding and its pipeline.

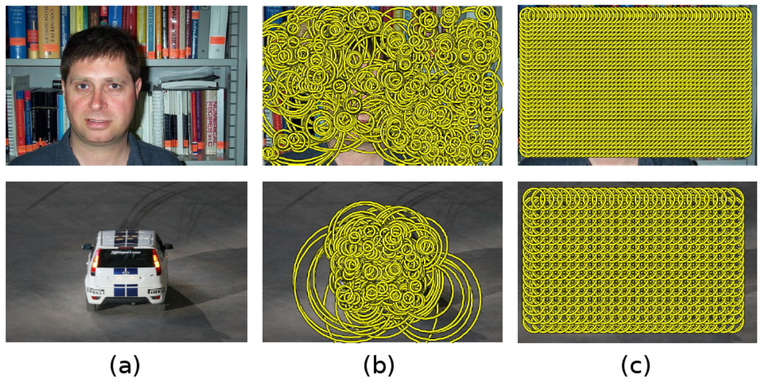


Figure 3 – The low-level description of an image (a) can be extracted using many sampling strategies. Each yellow circle represents a region to be described and two popular strategies for defining such regions are by detecting interest points (b), which involves the use of a detector (e.g. a simple edge detector), and by dense sampling (c), which uses a dense grid to define the regions to be described. Reproduced from Tuytelaars [2010] apud Avila [2013].

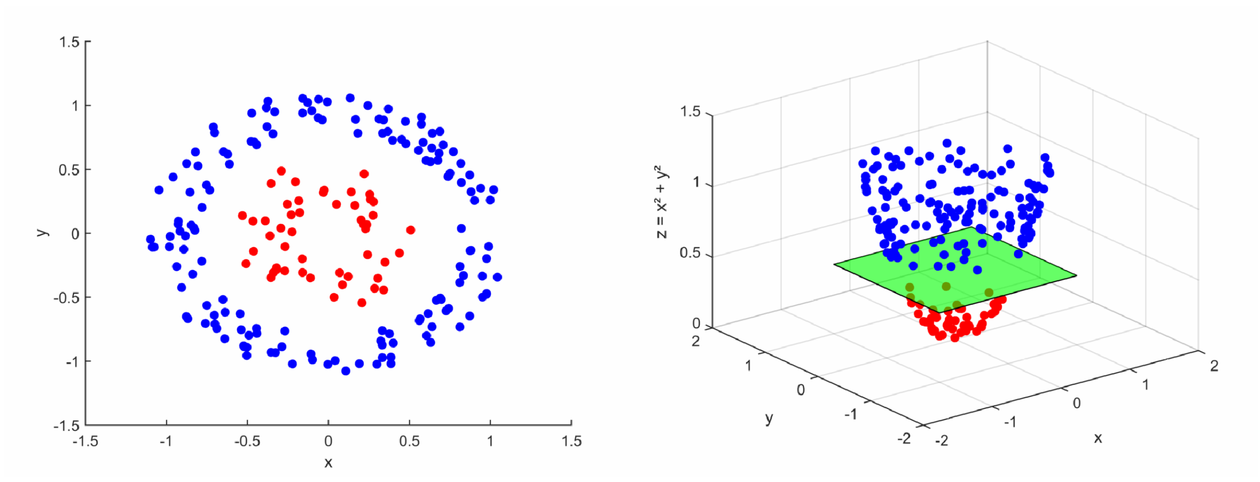
The emergence of kernel functions [Aizerman et al., 1964], capable of operating in high dimensional spaces without previously transforming every data point, contributed to the creation of SVMs [Vapnik, 1998]. Duda et al. [2000] explain that using a proper nonlinear mapping, data from two different categories can always be separated by a hyperplane in a sufficiently high dimensionality space, as long as the two categories do not have points in common. The idea behind SVM, represented in Figure 4, is to find such hyperplane, while maximizing its distance from its neighbor points. Figure 4a shows the input space (original data); in this space, it is clear that the categories are not linearly separable. Figure 4b, however, shows the input space after being transformed to a space with higher dimensionality (the *feature space*), where it is possible to find a hyperplane separating the categories[4].

For classifying the test data, each image has to be represented by the same descriptor used during the training phase. The mapping $\varphi()$ and the hyperplane (obtained by means of the training of the SVM classifier), can then be used to decide which category best represents the given sample, completing the classification procedure.

2.2 Artificial Neural Networks

The first mathematical model inspired by the biological neuron was introduced in 1943 by McCulloch and Pitts [1943]. This publication not only started the studies of neurocomputing, but also created the first artificial neuron (exemplified in Figure 5). In 1949, in *The Organization of Behavior*, Hebb [1949] described a system capable of learning using neu-

[4] More details about SVM can be found in the book of Duda et al. [2000], section 5.11.



(a) Data in $\mathbb{R}^2$ (b) Data transformed to $\mathbb{R}^3$

Figure 4 – Support Vector Machines rely on kernel functions to implicitly find a space of higher dimensionality where the categories can be easily separated. To exemplify how a space of higher dimensionality can help in separating classes, we show in Figure 4a data in $\mathbb{R}^2$, with two categories: red and blue. Clearly, there is no way to linearly separate such categories in this space. In Figure 4b, the original data is transformed to $\mathbb{R}^3$, with $z = x^2 + y^2$, where it can be easily separated by a hyperplane (green).

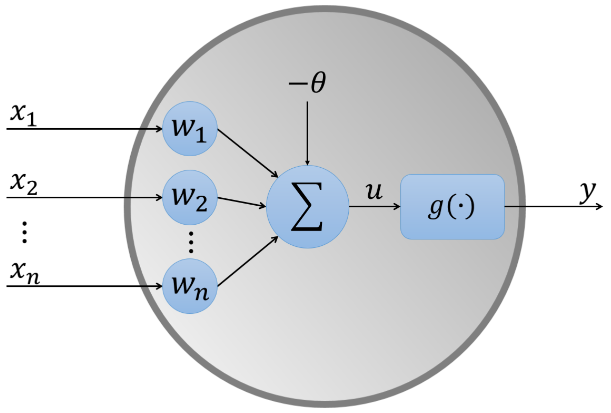


Figure 5 – An artificial neuron. The internal value u is obtained by subtracting an activation threshold θ from the sum of all of the input values x_i multiplied by their respective trainable weights w_i. The output of the artificial neuron is obtained by $g(u)$, where $g(\cdot)$ is the activation function. Figure adapted from da Silva et al. [2010].

ron correlations, introducing the Hebb's rule, the first training method for Artificial Neural Networks (ANN).

In 1959, Rosenblatt [1959] developed the first neurocomputer, named *Mark I – Perceptron*, creating the first model of a *Perceptron*, a linear classifier inspired by the artificial neuron. The design of a Perceptron was further debated by Minsky and Papert [1969], in their book *Perceptrons*, also known for proving that a Perceptron could not represent the operation XOR (eXclusive-OR). In the same book, a conjecture (as they termed: "intuitive" judgement) marked the history of ANN, arguing that such limitations would extend to any possible configuration of Perceptrons, discouraging many researchers to continue working in these models.

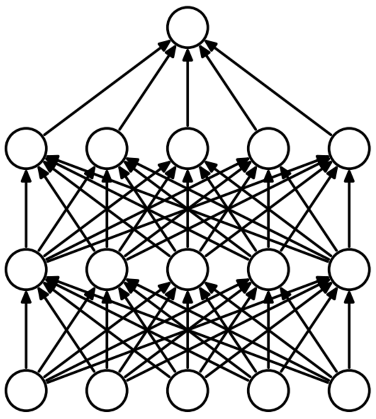
Four years later, Werbos [1974] introduced a procedure based on the idea of reverse gradients that could be directly used for training ANNs, but its potential was not fully understood by then, and eight years later Hopfield [1982] proposed the *Hopfield Network*, capable of mimicking associative memory. It was only by mid 80s that ANNs started regaining the interest of the scientific community, when a procedure very similar to the one proposed by Werbos [1974] was reintroduced under the name of *backpropagation* [Rumelhart et al., 1986], showing how multilayered networks could be efficiently trained and proving wrong the XOR conjecture.

A little earlier, Fukushima [1980] proposed the first Convolutional Neural Network (CNN). Although many authors have also presented models with such capabilities, it was only in 1995 that the first popular CNN was proposed, by Lecun and Bengio [1995]. A CNN is a variant of a Multilayer Perceptron (MLP), also inspired by biological processes, strongly founded on the mathematical operation of convolution.

Modern CNNs for state-of-the-art classification challenges have to deal with massive amounts of data and create robust representations. For this reason, their architecture is usually very complex, demanding efficient ways to be trained. Many authors have addressed this issue, creating simple techniques for improving the quality of the model, like the Dropout regularization [Hinton et al., 2012], exemplified in Figure 6, and for reducing computational cost of training the classifier, like the Rectified Linear Unit (ReLU), which simplifies the activation function to $g(z) = max(z, 0)$ [LeCun et al., 2015].

In Figure 7, we show an example of a CNN; each layer (represented by a horizontal block) is the result of a convolution operation applied to the previous layer, and modern CNN architectures may contain several convolutional and fully connected layers. Fully connected layers have each of their artificial neurons connected to all of the outputs of the previous layer, forming a *complete bipartite graph*. In Figure 6a, all of the layers are fully connected.

For an extended review of the ANN literature, we recommend the reader to refer to the book of Haykin [2009].


(a) Standard multilayer feedforward neural network

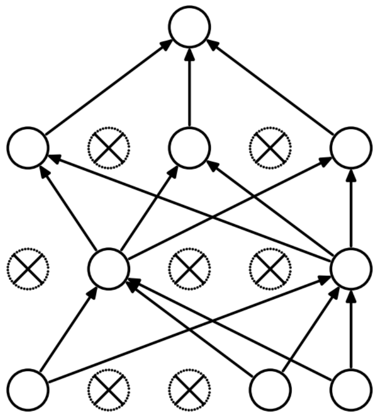

(b) After applying dropout

Figure 6 – Example of the Dropout regularization. Figure 6a shows a neural network with two hidden layers, Figure 6b shows the same network after applying dropout (crossed units have been dropped). This procedure is repeated for each training case, effectively creating different random architectures inside the same network. Reproduced from Srivastava et al. [2014].

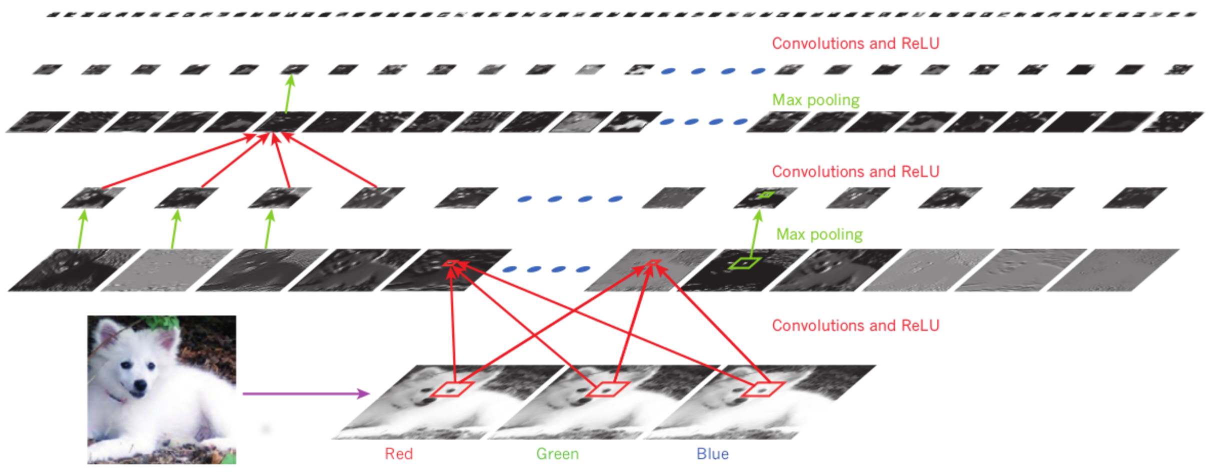
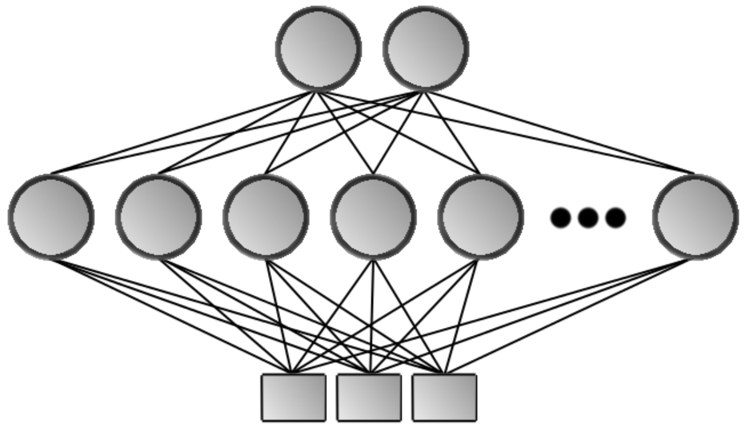


Figure 7 – Representation of a Convolutional Neural Network. Each layer has a bank of filters and its output is the application of such filters to the input of the layer. The input of the first layer is usually a raw image, with Red, Green and Blue color channels. Pooling strategies (like max pooling) and activation functions (like ReLU) may be applied between convolutional layers. Reproduced from LeCun et al. [2015].

2.3 Deep Learning

Deep architectures have been around for a long time, as the very model presented by Fukushima [1980] may be considered a Deep Neural Network. The term *Deep Learning* (DL), however, was broadly adopted only after a publication of Hinton et al. [2006]. By then, timing was just right for the technique to gain the attention of the scientific community: computers were much faster than before and graphics boards were widely accessible, allowing bold optimizations with the use of General-Purpose Graphics Processing Units (GPGPUs), as described by Oh and Jung [2004].

Recently, Le et al. [2011] reported a huge architecture trained in an unsupervised protocol that achieved great results. The size of the model, however, is so big that a modest-sized cluster would be required just for using it in test mode, making the technique accessible for big companies needing powerful classifiers, but not for standard laboratory experiments. This work marks the transition of ANN from the laboratories to a massively parallel scheme, distributed in many computers, greatly increasing their scalability. At the time, there was no architecture learning at such magnitudes (one billion trainable parameters), specially in a strongly distributed environment (1,000 machines).


(a) Shallow Network

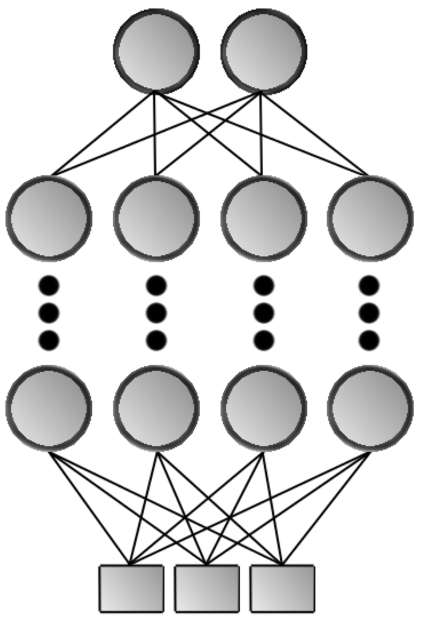

(b) Deep Network

Figure 8 – A comparison between shallow (a) and deep (b) architectures. Each layer from a deep architecture operates on the outputs of its previous layer, creating an architecture that can be progressively more complex. Because of the additional depth, (b) may also demand less parameters for the same number of nodes.

A variety of different definitions for deep architectures have circulated the scientific community. In this work, we adopt the one proposed by Bengio and LeCun [2007]: "In general terms, deep architectures are composed of multiple layers of parameterized non-linear modules. The parameters of every module are subject to learning". In Figure 8 we represent a shallow (Figure 8a) and a deep (Figure 8b) architecture. Bengio [2009] argues that a shallow architecture, in contrast with a deep architecture, may need exponentially more computational elements (in this case, artificial neurons) to complete the same task. This perspective encouraged scientists to try DL in very complex tasks, like the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [Russakovsky et al., 2015] and the PASCAL Visual Object Classes (VOC) Challenge [Everingham et al., 2010].

It was only in 2012, when Krizhevsky et al. [2012] won the ILSVRC 2012 (classification challenge), that a DL network reached an established degree of scientific and public attention. They reported a large, deep convolutional neural network, with 60 million parameters and 650,000 neurons, trained to classify the 1.2 million images from the ILSVRC contest (a subset of the ImageNet dataset). Their results were considerably better than the previous state-of-the-art, but a detailed description on ImageNet classification and localization with CNNs was published only a few months later.

In 2013, Jia et al. [2014] made available the Caffe deep learning framework and, along with Donahue et al. [2014], the DeCAF convolutional neural networks framework. Sermanet et al. [2013] also released an open-source package for classification and off-the-shelf image description and, in 2014, Vedaldi and Lenc [2014] released the MatConvNet, making easier the task of creating and training deep models. These were to be the building blocks for a generation of models that redefined the state-of-the-art panorama for several classification tasks through Transfer Learning, discussed in the next section.

2.4 Transfer Learning

Transfer learning is used when one has a model, trained for a specific source task, and wants to reuse part of its knowledge to help solving another problem. There are many schemes that can be employed for making the transfer of knowledge, ranging from combining the source and target training sets to estimate the parameters, to completely reusing some of the parameters estimated in the source task into the target task. Several authors have addressed transfer learning, with or without ANN involved (e.g. Razavian et al. [2014]; Oquab et al. [2014]; Donahue et al. [2014]; Zeiler and Fergus [2014]; Gong et al. [2014]; Girshick et al. [2014]; Tommasi et al. [2010]; Aytar and Zisserman [2011]; Pan and Yang [2010]; Iandola et al. [2014]).

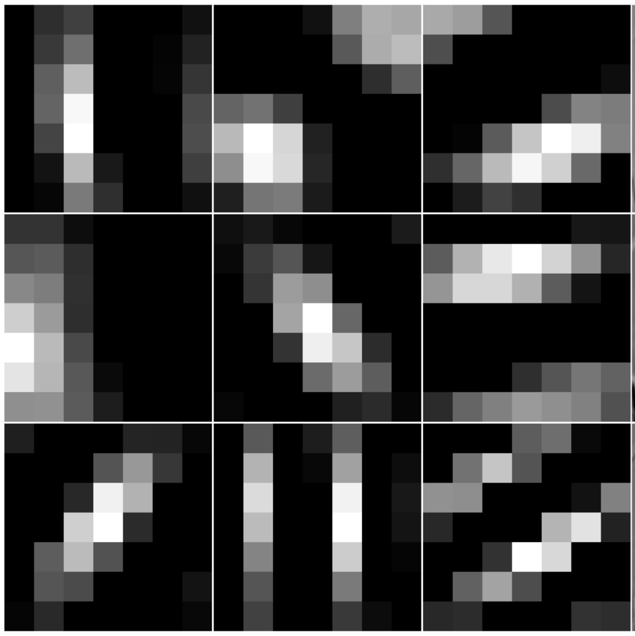
Classical use of CNNs for transfer learning takes into account that the lower levels of the network are essentially learning to mimic simple Gabor filters, producing very simplistic features that can be considered as a low-level description by the architecture, while subsequent layers learn to generate descriptors with increasingly higher abstraction by associating features of lower levels [Yosinski et al., 2014]. Zeiler and Fergus [2014] not only support this claim, but also show that deeper layers have better invariance to transformations on the input. The similarity between Gabor filters and the first layer of a CNN can be easily seen in Figure 9. Filters from the first convolutional layer of a CNN (Figure 9a) look like simplified versions of Gabor filters (Figure 9c), and it is possible to see that their application in an image can produce similar looking images, emphasizing borders and local characteristics (Figures 9b and 9d).

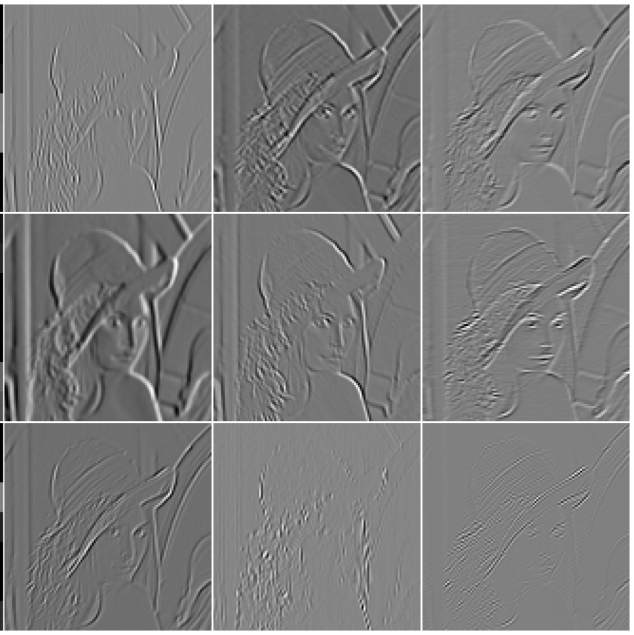
A straightforward strategy for transferring knowledge is to fix the weights up to a chosen layer of the network and then reshape and retrain the remaining layers for the new task (see Figure 10). This kind of setup aims to reduce the number of parameters to be learned by reusing knowledge, considered to be convenient to a new task, from the previous model. Although it is possible to end up admitting some knowledge that may no longer be useful, the computational effort needed is reduced by focusing only on learning the new task, instead of also learning how to represent the input data (see the work of Razavian et al. [2014] and Yosinski et al. [2014] for applications of this strategy).

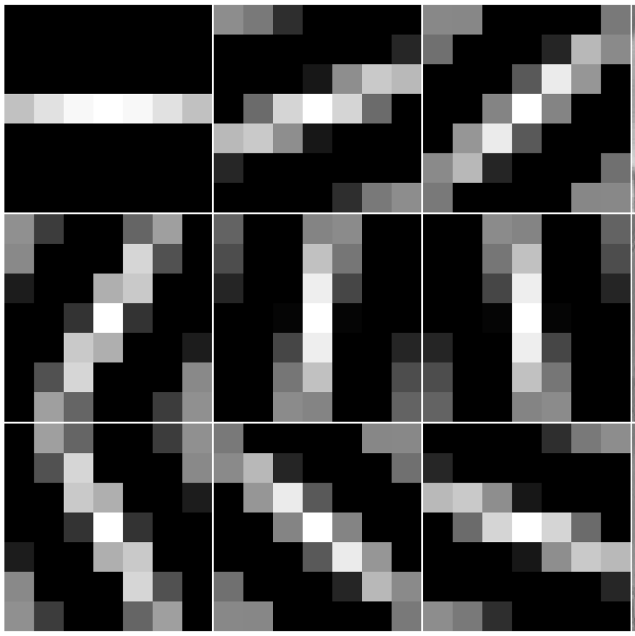
This procedure grants more control over the transfer process, since it is possible to simply pick different layers from the original network, resulting in distinct degrees of *knowledge* being transferred, and because no training is needed in the fixed layers, the output of the last fixed layer can be seen as a feature vector, enabling the use of a different classifier (e.g. an SVM) at the end of the network.

Choosing the layers to be fixed is a problem addressed by Razavian et al. [2014]. Although there are layers that usually yield best results, there is no layer known to always be the optimal choice. The basic assumption is that layers located close to the input are learning to act as low-level descriptors, while layers located near the output can detect high-level features, very specific to the task. It is reasonable to believe that the chosen layer should be selected according to the similarity between the original and the new tasks, as similar tasks should demand similar features, and tasks with less similarity should share only the basic filters of the network, having different representations in higher layers.

To improve the results obtained by the transference of knowledge between models, Gong et al. [2014] described a procedure for extracting features from different regions of the image, aiming to better geometry invariance. Such procedure vaguely resembles Spatial Pyramid Matching, proposed by Lazebnik et al. [2006] (Lazebnik is part of both teams).


(a) Normalized filters extracted from the first convolutional layer of MatConvNet M


(b) Correspondent normalized outputs of the first convolutional layer


(c) Normalized Gabor filters

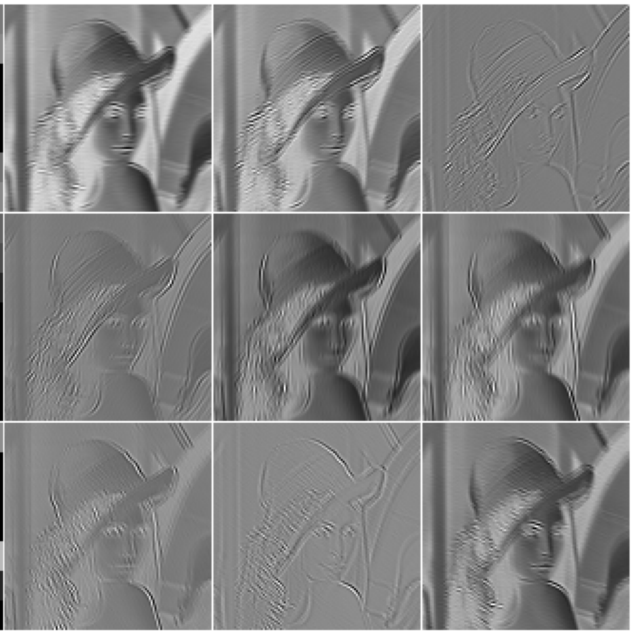
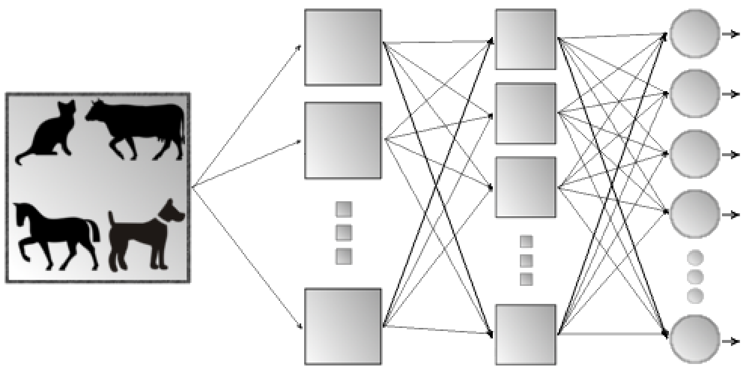
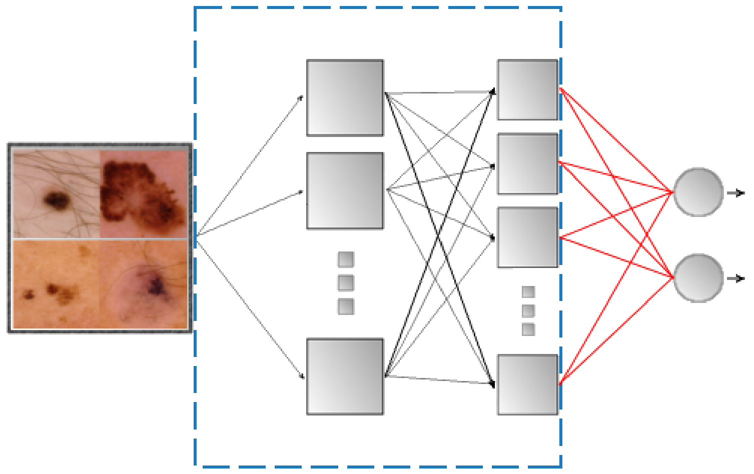

(d) Correspondent normalized output of convolution operations using Gabor filters

Figure 9 – The similar nature of the first convolutional layer of a CNN and Gabor filters. Figure 9a shows 9 filters extracted from the first convolutional layer of the MatConvNet M network [Vedaldi and Lenc, 2014] and Figure 9b shows the resulting images after the convolution operation with such filters. The same process is shown with 9 Gabor filters in Figures 9c and 9d, respectively.



(a) Original network



(b) New network, with fixed layers surrounded by a blue dashed rectangle

Figure 10 – Transferring knowledge from a deep neural network to a new task. Figure 10a shows the original network, trained in an animal dataset (fictional), and Figure 10b shows the resulting network, after fixing the first n layers and reshaping and retraining the remaining layer to classify a skin cancer dataset.

Similar approaches were proposed by Girshick et al. [2014], Zhang et al. [2014] and Iandola et al. [2014].

Another promising way to apply transfer learning to medium-sized datasets is by also fine-tuning the network. This means that the layers inside the blue dashed rectangle, in Figure 10b, are not completely fixed, but have smaller learning rates instead. Allowing these layers to slightly adapt during the process of transferring knowledge may have an important impact in the network's performance, as small problems caused by them may be corrected, without drastically changing the weights and the representation inside the network. Although fine-tuning is out of the scope of this project, we strongly recommend the reader to refer to the work of Yosinski et al. [2014] for a better understanding of the subject.

Osadchy et al. [2007] presented an architecture similar to the one proposed by LeCun et al. [1998], but trained for detecting faces and estimating pose. They postulated that such tasks may have correlation in lower layers of the network, and created a model that can benefit from this concept, achieving better scores than individual classifiers for both assignments. Their results indicate that models can benefit from task similarities to improve performance.

It is Oquab et al. [2014] who finally argued that deep architectures can be surpassed because of their data-hungry nature. Several low-level and mid-level descriptors have emerged due to the problem of small datasets, since learning such representations from the images is not trivial and may require large amounts of data. Examples of such descriptors are SIFT [Lowe, 2004], SURF [Bay et al., 2008], RootSIFT [Arandjelovic and Zisserman, 2012], Spatial Pyramid Matching [Lazebnik et al., 2006], VLAD [Jégou et al., 2010], Super-Vector [Zhou et al., 2010], LLC [Wang et al., 2010] and BossaNova [Avila et al., 2013].

Donahue et al. [2014] and Razavian et al. [2014] explored transfer learning across many tasks and reported good results. As Donahue et al. [2014] point, although training DL models is a time-consuming task, using them as feature descriptors is not necessarily slower than other robust descriptors available.

We also acknowledge that Razavian et al. [2014] and Zeiler and Fergus [2014] suggest experiments resembling the ones we will describe in Chapter 3, using network's final layers. These experiments, however, are performed up to the last but one layer, contrary to our experiments. Razavian et al. [2014] also present an analysis of the performance's impact of choosing different layers for the transferred features, complementary to our experiments presented in Chapter 4, and show that layers close to the end of the network lead to the best results, strongly supporting some of our findings.

2.5 Automated Melanoma Screening

According to Urteaga and Pack [1966], the first recorded case of melanoma probably dates back to the fifth century B.C.E., but the first report of the surgical removal of a melanoma points to the year of 1787 [Rebecca et al., 2012]. Formerly known as the "black cancer", melanoma is the most dangerous of all skin cancers [Gniadecka et al., 2004], and cutaneous melanoma is among the most aggressive types of human cancer [Monzani et al., 2007]. It has, however, high survival rates if detected and treated early, but in the absence of early treatment it can quickly become fatal [Gniadecka et al., 2004; Jerant et al., 2000].

Because of the infeasibility of having a dermatologist available for isolated and remote regions, automating the melanoma screening is an important task, already under exploration by the scientific community [Fornaciali et al., 2014]. Even though skin lesions are known to have visual patterns, the similarities between malign and benign cases make it a task of very high complexity, as shown in Figure 11.

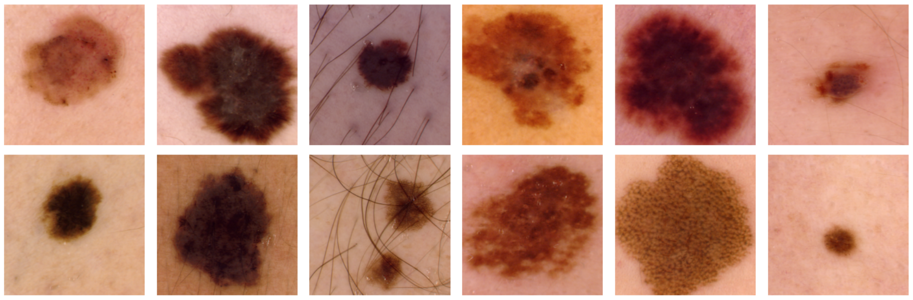


Figure 11 – Examples of skin lesions. Their classification is a challenging task, as melanomas (top row) may be very similar to benign skin lesions (bottom row). Reproduced from Fornaciali et al. [2014].

Most of the studies on automated melanoma screening, however, rely on computational implementations of the methods used by dermatologists, analyzing predefined visual characteristics, instead of a straightforward machine learning approach. Successful examples are the approaches based on the ABCD[5] rule, which is one of the most adopted techniques (e.g. Abbas et al. [2012]; Pellacani et al. [2006]; Gola Isasi et al. [2011]; Iyatomi et al. [2008]), and the ones based on the 7-point checklist[6], which allows less experienced observers to obtain high diagnostic accuracy [Argenziano et al., 1998] (e.g. Di Leo et al. [2010]). Both methods, however, perform poorly on atypical cases, as explained by Argenziano and Soyer [2001].

We highlight three approaches that differ from the ones previously mentioned: Baldi et al. [2009] proposed a content-based information retrieval strategy, searching in a set of labeled images for the more similar to the lesion under classification; Zouridakis et al. [2015] extracted features according to many clinical schemes, including the ABCD rule and the 7-point checklist, and other algorithms (for segmentation, iterative self-organizing data analysis, clustering, connected components and more), using many classifiers; finally, Fornaciali et al.

5 ABCD stands for the characteristics of melanomas: (A) asymmetrical, (B) irregular border, (C) multiple colors and (D) differential structure or diameter. The ABCD rule is often extended to ABCDE, with (E) for evolving, indicating changes in the lesion. Please refer to the work of Nachbar et al. [1994] for more information on the ABCD rule, from the medical point of view.

6 The 7-point checklist comprises 2 groups, the first with the major features of a melanoma: (1) atypical pigment network, (2) gray-blue areas and (3) atypical vascular pattern; and the second with the minor features: (1) streaks, (2) blotches, (3) irregular dots and globules and (4) regression pattern. Please refer to the original proposal, by Argenziano et al. [1998], for more information on the 7-point checklist.

[2014] applied a modern mid-level descriptor, based on the BoVW model, and show the impact of the different description levels for the classification.

Table 1 – State-of-the-art for automated melanoma classification.

Authors	**Method**	**Dataset #pos/#neg**	**AUC (%)**
Fornaciali et al. [2014]	BossaNova; SVM	187/560	93.7
Seidenari et al. [2005]	Color descriptor; Linear combination	95/364	93.3
Iyatomi et al. [2008]	Color and texture descriptors; Neural network	198/1060	92.8
Wadhawan et al. [2011]	Haar wavelet; SVM	388/912	91.4
Abbas et al. [2012]	ABCD rule-based features; SVM	60/60	88.0
Situ et al. [2008]	Color histogram; Gabor filter; BoVW; SVM	30/70	82.2

We show the state-of-the-art results for automated melanoma classification in Table 1. In the scope of this project, we have limited our search only to the authors who report their Area Under the ROC Curve (AUC) scores.

2.6 Discussion

In Section 2.1, we presented two sampling methods for extracting low-level features: by interest point detector and dense sampling (see Figure 3). Although both approaches are known to yield good results in a variety of tasks, the first clearly relies on a robust detector and the second may encode information not relevant to the problem, forcing the next steps to filter out the noise.

It is also easy to perceive that learning from any of the mentioned {low and mid} description levels have disadvantages: (1) A low-level description is usually in a space of very high-dimensionality and may have no distant spatial correlation, making it hard to learn a model directly from it, and (2) a mid-level description passes through destructive coding and pooling processes, removing potentially important information from the final description. The first problem can be properly addressed by correctly choosing the classifier to be used, but we would still suffer from poor description, as low-level features are very localized and would not be able to capture distant correlations in the image. For the second problem, alternative coding and pooling strategies have been formulated by the scientific community, like changes in the soft-assignment coding, proposed by Liu et al. [2011], but even the most carefully handcrafted descriptor will inevitably suffer from biases introduced by its authors.

The idea behind autoencoders, based on the concept of sparse coding [Olshausen and Field, 1996], is to learn the coding operation while maximizing the amount of information preserved. This approach has strong advantages over the traditional coding operations, but it introduces to the model an extra level that needs to be learned independently. The DL approach, however, usually involves learning every step in the process, from feature extraction to supervised classification, in a unified scheme that progresses from the pixels to the local features, from those to the higher level features, and from those to the final label assignment. Although DNNs may also have pooling layers, they are trained with these layers from the beginning, allowing the model to adapt itself in a way that minimizes the information lost by the pooling operation. This pervasive learning gives DL more generalizing capacity, because all of the operations necessary to the classification are learned by the same model at the same time, but creates new challenges: a DL model will have a huge number of parameters to estimate, thus requiring very large amounts of annotated data.

For small datasets, we have seen that transfer learning (Section 2.4) can be an interesting alternative to benefit from the power of DL models. The quality of the initial model, from which features are to be extracted, is, however, a pressing issue. It is arguable that as big datasets get even bigger, the features obtained by DL techniques will become more robust and general, improving the performance of transferred models in other datasets.

As for the transfer itself, one would expect the tasks from the original and the new models to be strongly different, because in scenarios where both tasks have class intersections, like ImageNet and Pascal VOC [Everingham et al., 2010], part of the transfer could be seen as a cross-dataset experiment. It is important, then, to evaluate the transferred features in tasks that are disconnected at semantic level, like the classes of ImageNet and the detection of skin cancer, further explored in Chapter 3. This reinforces the importance of this work, not only by the numerical results obtained, but also for better understanding how knowledge is represented inside DL models.

In Section 2.5 we briefly describe the techniques currently being used for the automated melanoma screening. Clearly, most of the art in this subject is still strongly related to methods used by dermatologists. We believe the attachment to such procedures, which analyze predefined characteristics, is mainly due to the transparency of the final model. By knowing the exact steps in the classification process, it is not only easier to validate the whole system, because of its behavior similar to the methods used by dermatologists, but also to justify possible misdiagnoses.

We defend that, even though machine learning approaches are not as explicit as the ones inspired by procedures from the medicine, they may find an unbiased perspective to the problem, allowing for high accuracy diagnosis. Our claim is strongly supported not only by the results in the melanoma problem (e.g. Fornaciali et al. [2014]), but also by the analysis of other tasks for visual classification, such as the ILSVRC and the Pascal VOC.

3 Complete Transfer Learning

As seen in Section 2.4, the classical use of Convolutional Neural Networks (CNNs) for transfer learning is based on the idea that the lower levels of the network are essentially learning to mimic common Gabor filters. Subsequent layers, then, learn to generate descriptors with increasingly higher abstraction by associating features of lower levels and, although there are layers that usually yield best results while being transferred, there is none known to *always* be the optimal choice.

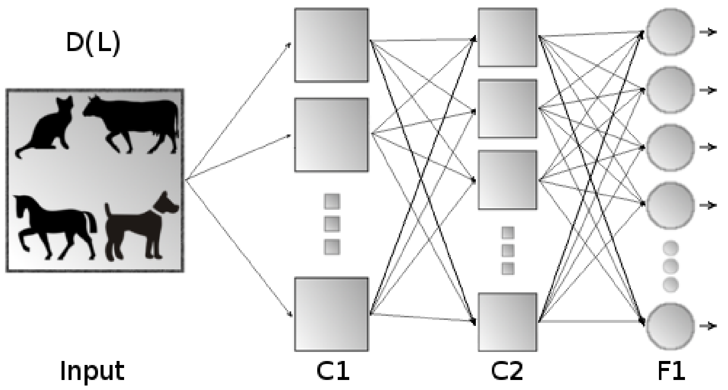
In this chapter we explore a novel perspective to the problem, showing that feature maps can be extracted from the last layer of a CNN in a way that makes it possible to achieve good results, even for tasks that are semantically different from the dataset for which the original model was trained. We believe that the preservation of every layer of the original network, allied to its application in tasks that suffer from the lack of annotated data, may lead to competitive performance.

The scheme that we call Complete Transfer Learning (CTL) keeps every layer in the source network in order to transfer its knowledge to different tasks. Since the high-level layers tend to be very task-specific, the idea to preserve all layers might seem preposterous at first, but we demonstrate experimentally that it works well, when compared to classical transfer-learning (see Section 3.3). Our results reveal interesting properties of the representations constructed by deep architectures, suggesting that the information loss in higher layers of the network is not as large as previously thought.

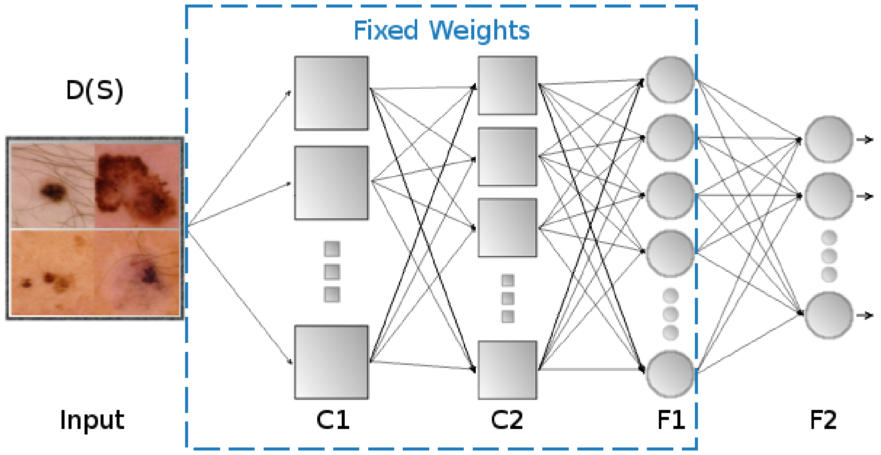
3.1 Proposed Approach

Because Deep Convolutional Neural Networks (DCNNs) are composed of multiple layers, each containing several parameters, training them is a matter of estimating such parameters. In order to avoid overfitting (when the model over-adapt to the training data, memorizing it and reducing generalization) and underfitting (when the model is undertrained, whether by lack of enough training data, training time or other factors), the required size for the dataset can grow as much as the increase in the sophistication level of the model [Oquab et al., 2014].

Many transfer schemes have been proposed for DCNNs, and their use is reported to have state-of-the-art results for many challenging tasks (e.g. [Chatfield et al., 2014; Razavian et al., 2014; Oquab et al., 2014; Donahue et al., 2014; Zeiler and Fergus, 2014; Gong et al.,



(a) $D(L)$: Classifier D, trained for classifying a large dataset L



(b) $D(S)$: Classifier D, trained for classifying a large dataset L and adapted for classifying the melanoma dataset S

Figure 12 – The Complete Transfer Learning proposal. $D(L)$: Large dataset classifier; $D(S)$: Melanoma classifier. Layers starting with C are Convolutional and layers starting with F are Fully Connected. On $D(S)$, part of the network corresponding to $D(L)$ (surrounded by a blue dashed rectangle) is fixed, so the last layer, added to the model, can learn how to classify the new dataset S based on the output of $D(L)$'s final layer.

2014; Girshick et al., 2014]). Such approaches, however, rely on feature vectors extracted from the outputs of middle-upper layers of the network, which usually bring extra dimensionality into the problem, potentially causing an increase in the number of parameters to be estimated and, consequentially, requiring bigger training sets.

Our approach to this problem is to take transfer learning to extremes, as illustrated in Figure 12. We assume to have two datasets: a small S, with few training examples, and a large L, containing several examples, many classes and high intraclass variety. A DCNN model D, represented in Figure 12a, was successfully trained on the large dataset L.

Due to the nature of DCNN models, we assume that D can detect and internally represent small nuances in the input. The intuition behind this assumption is that in order to correctly classify L, the model must be capable of representing small nuances in the input. This is specially reinforced by datasets with many examples and high intraclass variety (e.g. images taken from non-controlled environments), since the model would have to learn to locate the objects of interest by their many distinctive features. We postulate, then, that details about the input data may be preserved throughout the classification process and, given the state of one of the final layers of D, there may be important information coded into this state, allowing us to distinguish classes from S, even if they are not semantically related to the categories of L.

By assuming that the output layer $F1$ of $D(L)$ can also represent important information about the input, we can attach another classifier to *F1* as the new decision-making procedure; the parameters originated from $D(L)$ are then fixed and the new classifier $D(S)$ is trained in the melanoma dataset S (Figure 12b). In our example, the new classifier is a fully connected layer $F2$.

When an image is introduced to $D(S)$, it is processed by the fixed layers and, upon reaching the new classifier $F2$, the output of $F1$ is treated as input feature vector for training it. This new architecture is shown in Figure 12b. Our experimental results suggest that the removal of the Softmax operation in the last layer of the original network leads to better results and, because such operations do not characterize a visible layer, the cut is not visually represented in Figure 12. This claim will be further evident in Section 3.3 and Chapter 4.

An important remark is that, in our experiments, the new task (S) can be semantically different from the original task (L), in which the model may not be trained to detect any type of skin lesions or skin whatsoever. When we present the new dataset S to the new classifier $D(S)$ with the expected output ("melanoma" or "non-melanoma"), it will learn to recognize the state of $F1$ (with fixed parameters from $D(L)$) and to distinguish the new classes based upon any information preserved by it.

3.2 Experimental Setup

In order to validate the proposed approach, we address a very important problem that suffers intrinsically from small datasets: the automated melanoma screening. According to Gniadecka et al. [2004], melanoma is the most aggressive and dangerous of all skin cancers. It has, however, high survival rates if detected and treated early, but in the absence of early treatment it can quickly become fatal [Jerant et al., 2000] (see Section 2.5 for more details about automated melanoma screening).

In general, the protocols for creating annotated data for computer-aided diagnosis are very expensive and time-consuming, as the data must be manually annotated by medical specialists. It is, thus, impractical to obtain large datasets for training a whole DCNN for melanoma screening.

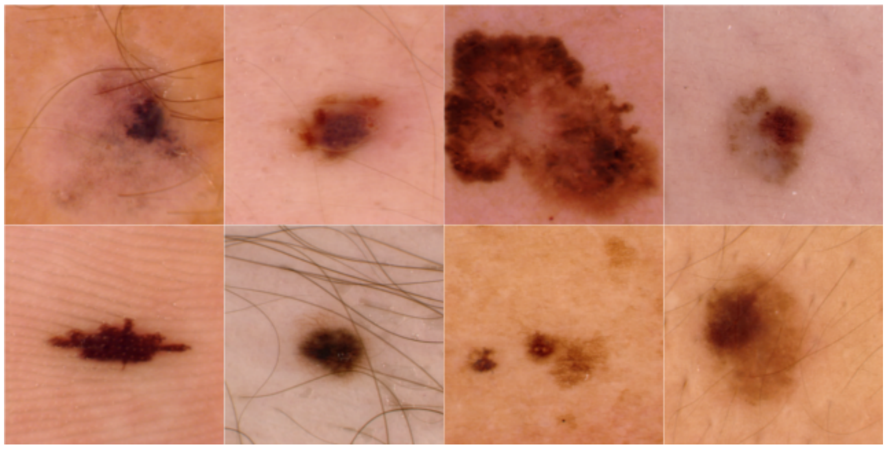


Figure 13 – Excerpts from the IRMA dataset. The melanoma images are shown at top row and the benign lesions are exhibited at bottom row. The similarity between positive and negative samples makes it a hard task.

Part of the experiments we describe in this chapter are performed on a third-party dataset composed of 747 dermoscopic images with resolution of 512×512 pixels each, developed by the Department of Medical Informatics from RWTH Aachen University. 187 of the 747 images in the IRMA dataset are considered melanomas and the other 560 are considered benign skin lesions. A few excerpts from it are shown in Figure 13.

With the purpose of strengthening our analysis of the differences between the CTL and the classical transfer approach, we also perform experiments on two extra datasets: MIT Indoors [Quattoni and Torralba, 2009] and Pascal VOC 2007 [Everingham et al., 2010]. Our goal in these experiments is not to reach state-of-the-art results for the mentioned tasks, leaving aside the tests using different classifiers, optimizations, and focusing only on the MLP approach. For this reason, we perform only the MLP experiments and we do not offer a comparison with results from the literature. Because the images from such datasets are not of fixed ratio, they were resized so their smaller side is of size 221 and square of size 221×221 was cut from their center.

Our initial model was chosen to be OverFeat [Sermanet et al., 2013], a publicly available CNN. OverFeat was trained on a 1000-category dataset, from the ILSVRC 2013[1], satisfying our conditions for size and variability of the training set in the initial task. Two models

[1] http://image-net.org/challenges/LSVRC/2013/

are provided with OverFeat: (1) fast, with reduced number of parameters and layers, increasing the speed of the network and (2) accurate, with smaller error on the ILSVRC dataset, but bigger and slower than the fast model. For the purpose of validating our approach, the accurate network was chosen, and the output of the last layer (Layer 9 in Table 2) was used as feature vector. For comparison, we have also performed the same experiments with the layer suggested by the OverFeat team for transfer learning (Layer 8 in Table 2, features extracted before its activation function).

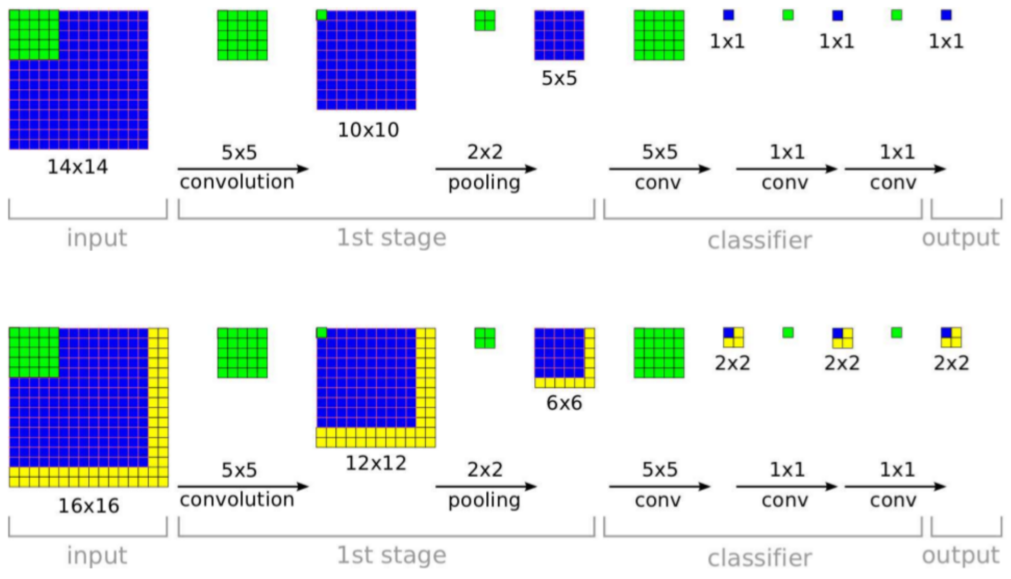


Figure 14 – Producing a bigger spacial output map by using convolutional layers, without the need for training another network with the desired image size. Figure reproduced from Sermanet et al. [2013].

Two setups were chosen for the experiments, the first with the original size of the images (512×512), and the second with the expected size for the accurate model (i.e. the size used for training it, 221×221). By using an image size bigger than the originally used during the training phrase, the output of each layer is changed due to the nature of the convolution operation. Such changes are represented in Figure 14, which shows how a spatial output map can be generated by CNNs using images of different sizes. Finally, a feature vector can be generated by applying the vectorization operation in the spatial output map.

After defining the layer to be transferred, we have described every image in our dataset by presenting it to the trained model, taking as feature vector the output of each of the chosen layers (8 pre-activation and 9). This procedure is repeated for the two different sizes we have defined.

For the next stage of our approach, two distinct classifiers were trained with the feature vectors previously extracted. The first, an SVM (see Section 2.1), and the second,

Table 2 – Details for the OverFeat accurate model. For both of our setups, the *spatial input size* is shown at the bottom of the table, with images in the expected size (221×221) and in their original size (512 × 512). Adapted from Sermanet et al. [2013].

	Layer 1	**Layer 2**	**Layer 3**	**Layer 4**	**Layer 5**	**Layer 6**	**Layer 7**	**Layer 8**	**Layer 9**
Stage	conv + max	conv + max	conv	conv	conv	conv + max	full	full	full
Num. channels	96	256	512	512	1024	1024	4096	4096	1000
Filter size	7×7	7×7	3×3	3×3	3×3	3×3	–	–	–
Conv. stride	2×2	1×1	1×1	1×1	1×1	1×1	–	–	–
Pooling size	3×3	2×2	–	–	–	3×3	–	–	–
Pooling stride	3×3	2×2	–	–	–	3×3	–	–	–
Zero-padding size	–	–	1×1×1×1	1×1×1×1	1×1×1×1	1×1×1×1	–	–	–
Spatial input size	221×221 512×512	36×36 84×84	15×15 39×39	15×15 39×39	15×15 39×39	15×15 39×39	5×5 13×13	1×1 9×9	1×1 9×9

an MLP (see Section 2.2). LibSVM [Chang and Lin, 2011] was chosen for running our SVM experiments, as it is a widely adopted library that supports all the desired features, and Theano [Bergstra et al., 2010], which is a compiler for mathematical expressions in Python[2], was chosen for the implementation of the MLP, as it provides easy ways to accelerate the experiments by using Graphics Processing Units (GPUs).

For the SVM experiment, preliminary tests have shown that the scores obtained by the linear kernel were not considerably different than those obtained by the Radial Basis Function (RBF) kernel, which presented higher training time. The linear kernel was then selected as our main SVM classifier.

As for the MLP experiment, we have used the hyperbolic tangent as activation function for the hidden layer, varying the number of hidden neurons in the architecture. The maximum number of training epochs was fixed at 1000, and the best epoch was chosen by its score on the validation set. This procedure encourages the network to be initially overtrained, evaluating its performance on the validation set for each epoch. After reaching the maximum number of epochs, we search for the epoch in which the training should have stopped (i.e. the one with the smallest validation error) and restore its state, effectively undoing its subsequent training.

With the purpose of maintaining the simplicity of the architecture, we have limited our search for the number of hidden neurons in the MLP to 7 distinct values: 5, 10, 20, 50, 100, 200, 500; as well as the number of hidden layers to 1.

The final architecture for the MLP experiment was composed of two layers: a hidden layer with variable number of neurons and a logistic regression layer with output of size 2, for positive and negative classification; and trained using the stochastic gradient descent.

Both of our SVM and MLP experiments followed a classical 5-fold cross-validation

[2] https://www.python.org/

protocol and, for optimizing the hyperparameters in the MLP experiment, we have further separated each training set in 4 balanced groups, using each combination of 3 to 1 as training and validation sets, respectively. The models with the biggest Area Under the Receiver Operating Characteristic (ROC) Curve (AUC) for their validation set were, then, chosen as representatives.

In order to promote a fair comparison between the results obtained by the different classifiers, the 5 folds were ensured to be the same for both experiments. We have also carefully altered the source code for both architectures in order to optimize the models by their AUC, which, in comparison to the accuracy, provides a better description of their performance.

3.3 Results and Discussion

A direct comparison between our results and most of the state-of-the-art for automated melanoma screening is troublesome, as each result is reached by different means. The contacted authors either refused to share their code or their datasets, therefore, reproducing their experiments was not feasible for the scope of this project. We do offer, however, an informative comparison, assuming equivalent noise and bias in all datasets.

Table 3 shows the results for our experiments with the IRMA dataset. Due to the elevated execution time, we chose to perform the RBF experiment only on the CTL approach for the 512×512 configuration. The scores are measured by the AUC, and the average score of all folds is considered in our analysis.

Table 3 – Results for our experiments with the IRMA dataset. The scores are given in AUC. Rows marked with **Complete** follow the proposed approach and rows marked with **Classical** use features extracted from the recommended OverFeat layer. **RBF** indicates an SVM with a radial basis function kernel, **LIN** an SVM with linear kernel and **MLP** a multilayer perceptron.

Approach	**Resolution**	**Fold 1**	**Fold 2**	**Fold 3**	**Fold 4**	**Fold 5**	**Average**
Classical + MLP	221×221	93.8	91.9	92.9	91.6	96.2	**93.3**
Complete + RBF	512×512	94.2	91.4	91.5	92.3	95.2	**92.9**
Classical + LIN	221×221	91.4	91.8	93.7	90.0	94.8	**92.3**
Classical + LIN	512×512	90.0	90.7	92.8	92.2	95.0	**92.1**
Complete + MLP	512×512	95.4	89.2	93.8	90.4	91.2	**92.0**
Classical + MLP	512×512	93.8	89.5	92.8	89.0	93.7	**91.8**
Complete + MLP	221×221	91.4	91.2	89.3	88.3	94.9	**91.0**
Complete + LIN	512×512	90.1	87.4	90.0	92.7	91.1	**90.3**
Complete + LIN	221×221	88.3	91.8	89.4	87.0	93.0	**89.9**

For the MLP experiment, we have calculated the ROC curve by displacements in the probabilities of each class, while the SVM experiment considered the distance to the hyperplane. We have adopted the composite trapezoidal rule to estimate the integral of the ROC curve, obtaining the desired AUC.

The results for the 512×512 experiment indicate that using an SVM with the RBF kernel as the new classifier, in the CTL approach, lead to slightly better results, with the average AUC of 92.9% versus 90.3% for the linear kernel and 92.0% for the MLP. In addition, the RBF kernel was more stable (best AUC = 95.2%, worst AUC = 91.4%, variation = 3.8 p.p.) than the linear kernel (best AUC = 92.7%, worst AUC = 87.4%, variation = 5.3 p.p.) and the MLP (best AUC = 95.4%, worst AUC = 89.2%, variation = 6.2 p.p.). However, the small sample of 5 folds was not enough to establish a statistical difference between the methods, as the paired Wilcoxon signed-rank test fails with p-value = 0.104 for the linear kernel and p-value = 0.625 for the MLP, suggesting that their difference is not significant.

As for the 221×221 experiments, it is clear that the classical approach with the MLP classifier was better, having the best average AUC for all of our experiments and the best overall scores in 2 out of 5 folds. We believe there are at least two factors behind the difference between the results of the 221×221 and of the 512×512 experiments: (1) The imposed constraints may cause underfitting in the latter (see Section 3.2 for details); and (2) Bigger-sized images may preserve less information during the feature extraction, as of any process composed of finite pre-defined convolutions and sliding-windows, in which resizing the input makes it harder to associate distant pixels. Although the classical approach with the MLP classifier had the best results, we point that such model was trained with feature vectors of 2^{12} dimensions, while the results for the CTL approach are reached having less than 2^{10} dimensions in the feature vectors. A statistical difference could not be established, since the paired Wilcoxon signed-rank test fails even in unfair comparisons, between the best classical and the worst CTL approaches, with p-values of 0.0625 for the 221×221 setup and 0.3125 for the 512×512 setup.

In Table 4, we show the approximated training time for our experiments. The MLPs were trained using General-Purpose Computing on Graphics Processing Units (GPGPU) in a GTX Titan Black video card, and the SVMs in a i5-3337U processor. We offer the SVM training times for reference, but we chose to exclude them from our analysis due to differences in the pipeline and implementation, causing the training time to spike up.

For the MLP training times, the overhead of the experiment, such as reading the input data and building the model, is clearly dominant in the 221×221 setup, in which no significant difference was noted. For the 512×512 setup, however, the feature vectors were much bigger, increasing the training time and suggesting that the MLP experiments

Table 4 – Approximated training time for our experiments. Rows marked with **Complete** follow the proposed approach and rows marked with **Classical** use features extracted from the recommended OverFeat layer.

Architecture	**Average Training Time (in sec.)**
Complete Transfer MLP (512×512)	6×10^1
Complete Transfer MLP (221×221)	3×10^1
Complete Transfer SVM (512×512)	3×10^3
Complete Transfer SVM (221×221)	6×10^1
Classical Transfer MLP (512×512)	2×10^2
Classical Transfer MLP (221×221)	3×10^1
Classical Transfer SVM (512×512)	6×10^5
Classical Transfer SVM (221×221)	3×10^1

with the CTL approach were faster than the classical experiment. An important highlight is the benefits of training these models on a GPU, greatly reducing their training time when compared to our SVM experiments, and still maintaining results close to the ones obtained by the SVM models, after an expensive (in terms of resources and time) grid search.

In order to facilitate our analysis, we reproduce the state-of-the-art for the automated melanoma screening in Table 5. We only consider the authors who have reported the AUC for their proposals.

Table 5 – State-of-the-art for automated melanoma classification.

Authors	**Method**	**Dataset #pos/#neg**	**AUC (%)**
Fornaciali et al. [2014]	BossaNova; SVM	187/560	93.7
Seidenari et al. [2005]	Color descriptor; Linear combination	95/364	93.3
Iyatomi et al. [2008]	Color and texture descriptors; Neural network	198/1060	92.8
Wadhawan et al. [2011]	Haar wavelet; SVM	388/912	91.4
Abbas et al. [2012]	ABCD rule-based features; SVM	60/60	88.0
Situ et al. [2008]	Color histogram; Gabor filter; BoVW; SVM	30/70	82.2

In terms of performance, our classical results are better than most of the state-of-the-art reported AUCs [Situ et al., 2008; Abbas et al., 2012; Wadhawan et al., 2011; Iyatomi et al., 2008], while our complete approaches are better than Situ et al. [2008]; Abbas et al. [2012]; Wadhawan et al. [2011] and slightly worst than Iyatomi et al. [2008]; Seidenari et al. [2005] and Fornaciali et al. [2014]. Our results are very similar to the ones obtained by Fornaciali et al. [2014], indicating that transferring knowledge from deep models, without prior information about the problem, can compete against very specialized architectures.

With the purpose of further validate our results, we show in Table 6 scores in two

Table 6 – Our results for the MIT Indoors and the Pascal VOC datasets. Two different evaluation methods were used, according to the standard evaluation for each dataset. The experiments followed our MLP setup and because the images from such datasets are not of fixed ratio, they were resized so their smaller side is of size 221 and square of size 221 × 221 was cut from their center.

Approach	**Dataset**	**Evaluation**	**Score**
Classical + MLP	MIT Indoors	Acuracy	**54.18**
Complete + MLP	MIT Indoors	Acuracy	**53.88**
Classical + MLP	Pascal VOC	mAP	**44.86**
Complete + MLP	Pascal VOC	mAP	**45.11**

different multiclass datasets, the MIT Indoors (67 categories) and the Pascal VOC 2007 (20 categories). Because we do not aim to reach state-of-the-art results, but rather compare the classical and the CTL approaches, we kept an experimental setup similar to the automated melanoma screening, further detailed in Section 3.2.

The results for the MIT Indoors experiments show small difference between the classical and the complete approaches, while the results for the Pascal VOC 2007 show better mAPs for the CTL approach. This reinforces that different description sizes may be efficient in adverse situations, in which there are constraints (controllable or not) to the experiment. The limited number of neurons in the hidden layer, limited number of training iterations, number of classes from the dataset and their separability in the semantic level favored the design with less parameters to be learned. This phenomenon may emerge in diverse situations, specially in the case of very small datasets.

Because the CTL approach discards the Softmax operation in the last layer, we can say that it is, in fact, learning from rich probability estimations. On this perspective, deep models with very distinct classes may generate stronger complete features, since they are able to represent images with less correlation in the class probabilities.

We also highlight that the choice of the model and the constraints of the experiment have a fundamental part in the success of the transfer. This claim will be evident in light of the results brought by Chapter 4, which further explores the Pascal VOC 2007 dataset, using a different deep model and aggressively stressing the representations used for transfer learning.

4 Internal Representations

Despite the great performance of DCNNs in many tasks, a complete analytical description of the models built through their use is still lacking. As a consequence, deep representations are not always fully understood. A great part of this is due to the unknown nature of the learned features, which may be subject to co-adaptation between layers and diverse transformations.

With the intention of better understanding such architectures, we devise, in this chapter, a series of experiments for measuring the robustness of their internal representations, aiming to analyze their redundancy, both in terms of dimensions and precision, and for verifying whether the use of strategies inspired by high-level representations can improve the quality of the description extracted from a deep model.

Our results point to strong internal repeatability, suggesting that the studied DCNN may have properties similar to neuron redundancy in animals, for recovering from brain damage. We offer an extensive analysis of this phenomena, showing ways to exploit properties of internal representations to achieve better classification results.

4.1 Proposed Approach

Aiming at better understanding the representations extracted from DCNNs, we have formulated 6 experiments, described as follows:

Stress 1 reduces the number of dimensions of the feature vector extracted from the model, randomly dropping a determined number of neurons from the last preserved layer before taking its output. With this experiment, we expect to detect redundant information in the representation of a given layer, should the performance of the model not fall accordingly. Figure 15 exemplifies the setup for this experiment.

Stress 2 systematically removes rightmost bits from the extracted features. This experiment will give us insight on how much precision is really needed for the elements of the feature vector, in order to achieve satisfying results.

Stress 3 forces quantization on the feature vector, in a manner similar to the way dictionaries do it in the Bag-of-Visual-Words model (see Section 2.1). Because of the nature of the chosen model, we have created 4 branches for this experiment. The first two

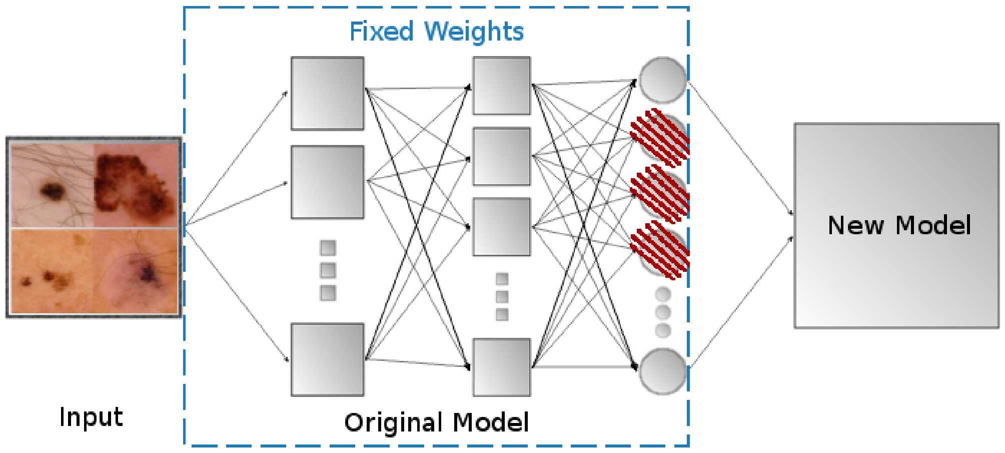


Figure 15 – Our Stress 1 experiments: A fixed number of neurons is randomly dropped from the last preserved layer and a new model is learned with the resulting features. In this example, red-scratched neurons are dropped.

branches (3A and 3B) find, each, a unidimensional Voronoi tessellation in a different interval, and the generated points are chosen as the dictionary entries. We have defined the intervals as follows:

Stress 3A $[j, k]$, where j is the minimum value and k is the maximum from all features, in all training samples;

Stress 3B $[0, k]$, where k is the maximum value from all the features, in all training samples;

The next two branches (3C and 3D) create one dictionary for each dimension of the feature vector. This is specially useful for dealing with features that are in different scales, giving them similar weight during the training phase. The intervals for the last two branches are defined as follow:

Stress 3C $[j, k]$, where j is the minimum value and k is the maximum value of the i^{th} elements from all training samples.

Stress 3D $[0, k]$, where k is the maximum value of the i^{th} elements from all training samples.

The approaches 3B and 3D are based on the idea that models with the ReLU activation function often disregard negative values during training and testing, possibly making them less informative. Finally, the feature vector is quantized by assigning to each element the value of the closest point in the dictionary.

Combined Stresses simultaneously applies stresses 1 and 2, dropping neurons in the last preserved layer of the original model and removing rightmost bits from the features at the same time. This will allow us to observe if the effects measured by stresses 1 and 2 are complementary.

Fused Descriptors tries to combine, through early and late fusion strategies, the descriptions extracted from two different layers. By associating outputs from different layers we may enhance the quality of the features, since it would allow mid-level and high-level information to be used by the new classifier. Depending on the chosen layer, spatial information may be preserved, creating a model capable of deciding based on both abstraction degrees.

High Level Representations employs strategies inspired by the approaches of Object Bank [Li et al., 2010] and Spatial Pyramid Matching [Lazebnik et al., 2006] in attempt to improve the quality of the feature vectors obtained for the transfer process.

4.2 Experimental Setup

Because we are not addressing a specific problem, the experiments described in this chapter are not bound to a determined dataset. It is desirable, though, that the size of the chosen dataset is not too small (< 2000 samples), since having more images would increase the precision of results. For this reason, we have defined that the Pascal VOC 2007 [Everingham et al., 2010] images should be used in all of the experiments of this chapter. Composed of 2501 images in the training set, 2510 images in the validation set and 4952 images in the test set, the Pascal VOC is well known in the literature, and was considered fit for our experiments.

The initial model, represented in Figure 16 and from which feature vectors were extracted, was chosen to be the M model, provided by MatConvNet [Vedaldi and Lenc, 2014] and proposed by Chatfield et al. [2014]. The expected image size for such model is 224×224 and, as suggested by the MatConvNet team, a bicubic interpolation was used for resizing the images from VOC 2007 before their use. The average image from the ImageNet training set was then subtracted, and the result was presented to the model as input.

In the M model, different operations are separated, yielding 21 layers instead of only 8. Table 7 shows the correspondence between the 8 groups represented in Figure 16 and such layers. Throughout this chapter, every reference to a *layer* points to one of the layers described in Table 7.

After extracting the image representations from the MatConvNet, we normalized them by dividing the features of each image by their L2 norm. Finally, a linear SVM was trained,

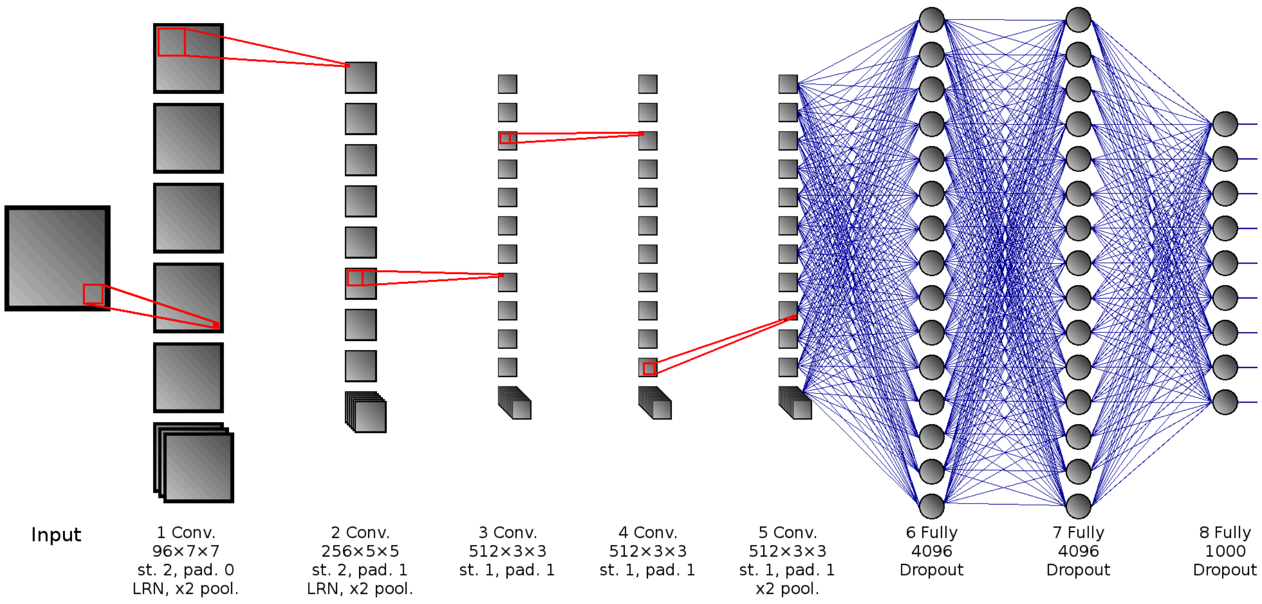


Figure 16 – Representation of the M model from the MatConvNet, used in our experiments. *Conv.* indicates a convolutional group, and *Fully* a fully connected group; the second row of each label indicates the number of filters and their size for the convolutional groups, or the number of neurons for the fully connected groups; the following lines indicate the extra operations or modifications used: *st.* for stride, *pad.* for padding, *pool* for max pooling and *Dropout* for the the dropout regularization.

Table 7 – Layers for the MatConvNet M model. *Group i* indicates the group from Figure 16; the first column of each line indicates the operation: *Convolution* for a convolutional layer, *Fully* for a fully connected layer, *ReLU* for the activation of Rectified Linear Units, *LRN* for Local Response Normalization, *Pooling* for Max Pooling and *Softmax* for the activation of the Softmax function. Each cell, belonging to a group (column) and operation (first column of each row), indicates the number of the correspondent layer. A dash indicates the absence of such operation in the group.

	Group 1	**Group 2**	**Group 3**	**Group 4**	**Group 5**	**Group 6**	**Group 7**	**Group 8**
Convolution	Layer 1	Layer 5	Layer 9	Layer 11	Layer 13	–	–	–
Fully	–	–	–	–	–	Layer 16	Layer 18	Layer 20
ReLU	Layer 2	Layer 6	Layer 10	Layer 12	Layer 14	Layer 17	Layer 19	–
LRN	Layer 3	Layer 7	–	–	–	–	–	–
Pooling	Layer 4	Layer 8	–	–	Layer 15	–	–	–
Softmax	–	–	–	–	–	–	–	Layer 21

having its hyperparameters chosen by the model with the best Mean Average Precision (mAP) in the validation set.

Before training the classifier, however, each of our experiments adopted a different strategy which may involve manipulating the normalized representations, and detailed information about them is given in the next sections of this chapter. We present the baseline for the experiments of this chapter in Figure 17, representing the transfer learning without disturbances in the descriptors extracted from the deep model. Although the layer 19 had the best results, the CTL approach (layer 20), proposed in Chapter 3, holds scores very close to it, with a difference of 1.02 p.p.

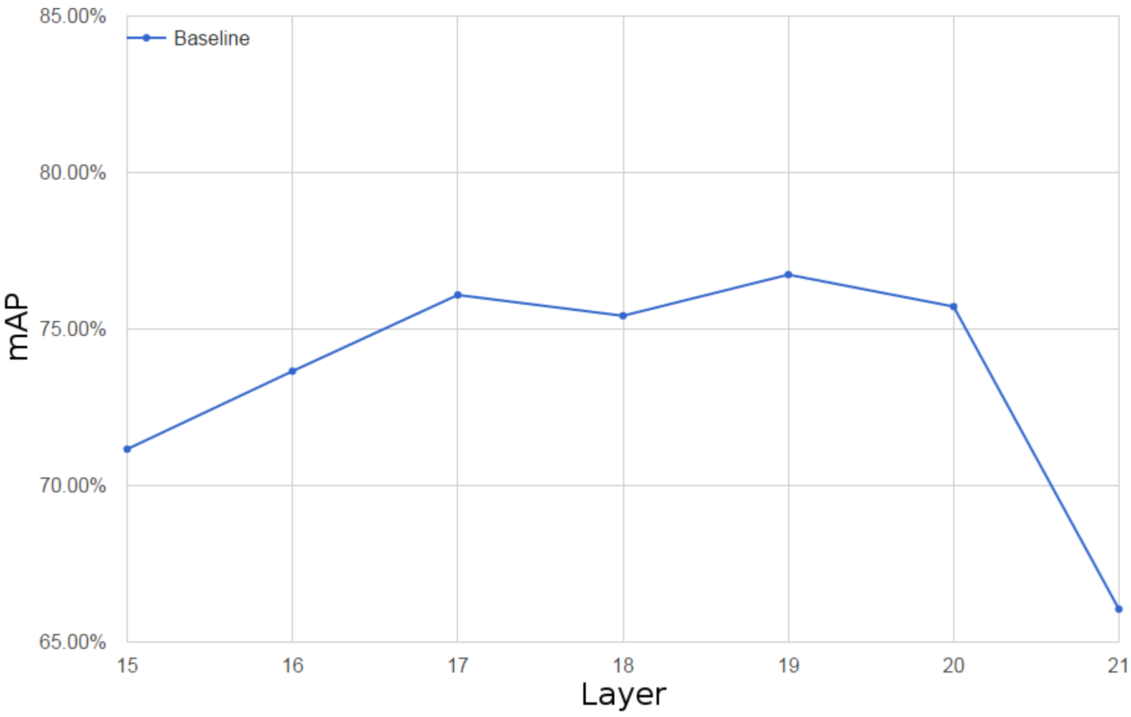


Figure 17 – Baseline for Transfer Learning. The mAP of features extracted from different layers of the MatConvNet M model for the Pascal VOC 2007 dataset.

4.3 Stress 1

In this experiment, we limit the number of features extracted from a deep architecture. Because feature vectors extracted from different layers may have different sizes (i.e. different number of dimensions), we defined that the number of features to be preserved would be determined by Pseudocode 4.1, which receives the size of the feature vector and returns an array, with each element dictating the number of dimensions to be preserved in one of the experiments to be performed.

The pseudocode generates a Fibonacci-like sequence, which is denser for smaller numbers and becomes sparser as the numbers grow. This setup was chosen as it gives us more information about the highly stressd models.

Pseudocode 4.1 Selecting the number of dimensions to be preserved for the Stress 1 experiments

```
function GET__PRESERVED__DIMENSIONS(size_of_feature_vector)
    $i \leftarrow 100$
    $previous \leftarrow i$
    $preserved\_dimensions \leftarrow [\ ]$
    while $i < size\_of\_feature\_vector$ do
        append $i$ to $preserved\_dimensions$
        $i \leftarrow i + previous$
        $previous \leftarrow i - previous$
    end while
    return $preserved\_dimensions$
end function
```

In order to reduce the effects of the randomness on subsequent runs, we wanted the set of dimensions preserved in any experiment to contain the sets of dimensions that were preserved on all other experiments with fewer features. This is exemplified in Figure 18, where the set A is contained in the set B, which is contained in the set C and so on.

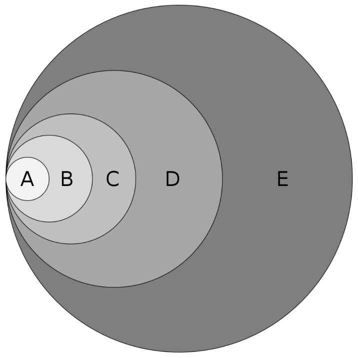


Figure 18 – A representation of the desired sets of dimensions for Stress 1. We expect each set to be contained in all of the sets bigger than itself, in order to reduce the effect of randomness in choosing their elements.

A straightforward approach to achieve this requirement, adopted in our experiments, is to start from the end of the array obtained by Pseudocode 4.1. Because the numbers will be in descending order, when a set of smaller size is required, we simply have to randomly drop a number of dimensions from the current set, corresponding to the difference between its size and the new desired number of dimensions for it, guaranteeing that the new set is contained in the set that came before it.

Finally, for each layer, we keep the same sets of dimensions for all samples, varying only their value according to their feature vector. This is crucial for the proper execution of the experiment, since changing the selected dimensions across samples would not allow the new classifier to be correctly trained due to inconsistency in the image descriptions.

Results and Discussion

Figure 19 shows the average of 10 runs for the Stress 1 experiments, with standard deviation values displayed as error bars. We chose to omit the layer 15 from our analysis because of its elevated number of dimensions.

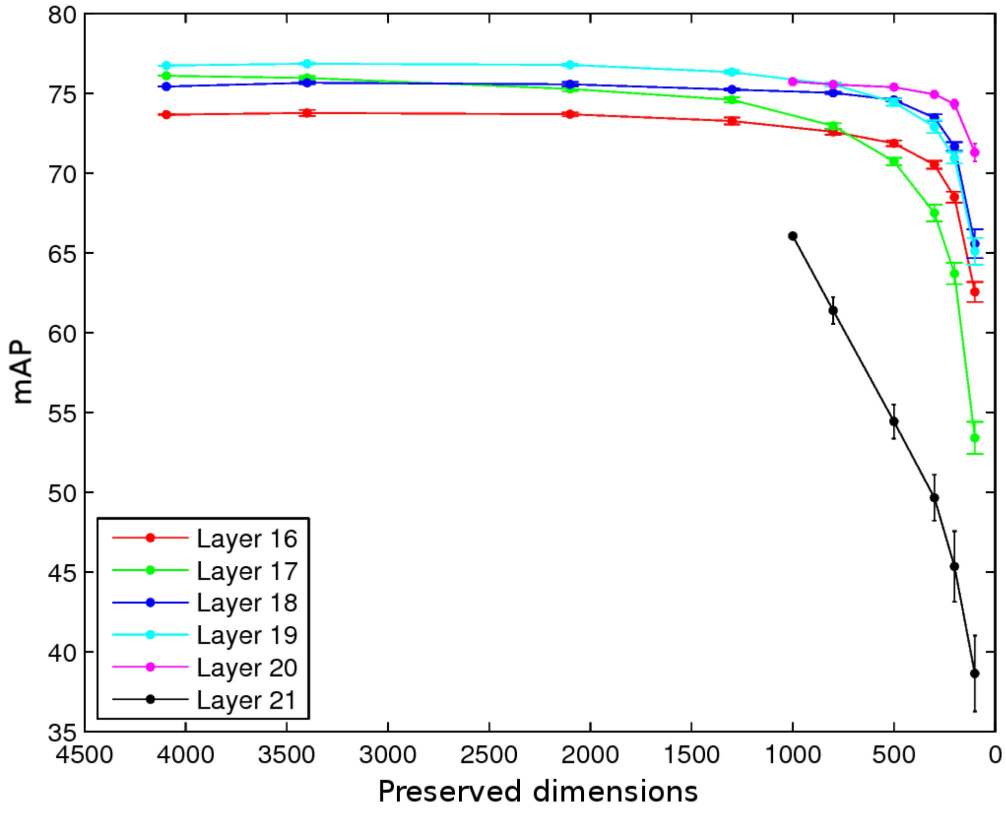


Figure 19 – Average of 10 runs for the Stress 1 experiments with standard deviation displayed as error bars. In this experiment we randomly drop dimensions from the feature vector, aiming to verify its redundancy. Layer 15 was omitted from our analysis because of its elevated number of dimensions.

Our experiments reveal strong information redundancy throughout deep representations, since small variations in the score were measured across different runs. We observe that for obtaining robust features, choosing the description extracted from layer 19 remains the best option, while the layer 20 is a strong candidate for compact features, having good performance (> 70% mAP) with only 100 dimensions preserved, corresponding to 10% of its original size, and very small loss with only 200 dimensions preserved (> 74% mAP).

They also indicate that for all of the tested layers, with the exception of the layer 21, there is usually small decreases in the score when dropping half of the dimensions in the feature vectors. As expected, the layer 19, which is usually the first choice for transfer learning, performs better for most of the configurations. For any setup with less than 1000 dimensions, however, layer 20 showed to be the best choice.

We point that most of the non-parameterized operations, such as the ones in layer 21, may negatively affect the transfer process and, even though we rely on an SVM for classifying the deep features, the effects of the scale of the input, which are known to be problematic, are reduced by the normalization applied on the feature vectors before the training stage.

4.4 Stress 2

Stress 2 reduces the precision of the features by erasing rightmost bits from their binary representation. Because we are working with the *double-precision floating-point format*, represented in Figure 20, there are 64 bits to be removed.

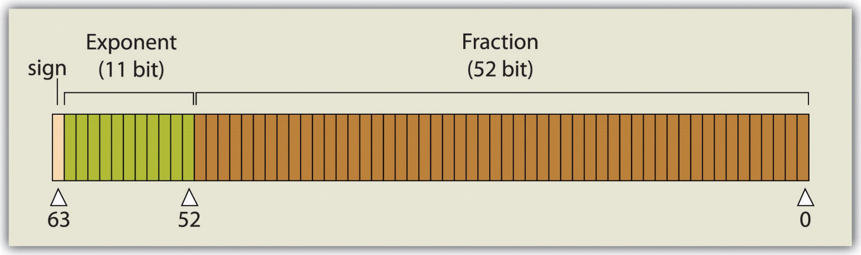


Figure 20 – A representation of the double-precision floating-point format according to the IEEE 754 standards. Adapted from Campbell and Shin [2012].

In this experiment, we start by erasing (setting to 0) the rightmost binary digit from the binary representation of each feature. After training the classifier and testing its performance, we erase one more digit (the second rightmost digit) and train another classifier. This process is repeated until all of the digits are erased, yielding 64 iterations for each layer.

Results and Discussion

Figure 21 shows the results for a single run of Stress 2 experiment, since our approach is deterministic and cannot generate different results for extra executions. We chose to omit the setups that removed less than 45 bits, as no significant loss was observed.

With up to 52 bits removed, no important change was perceived in the scores given by the model trained on the stressed features. Although the mAP of most layers dropped

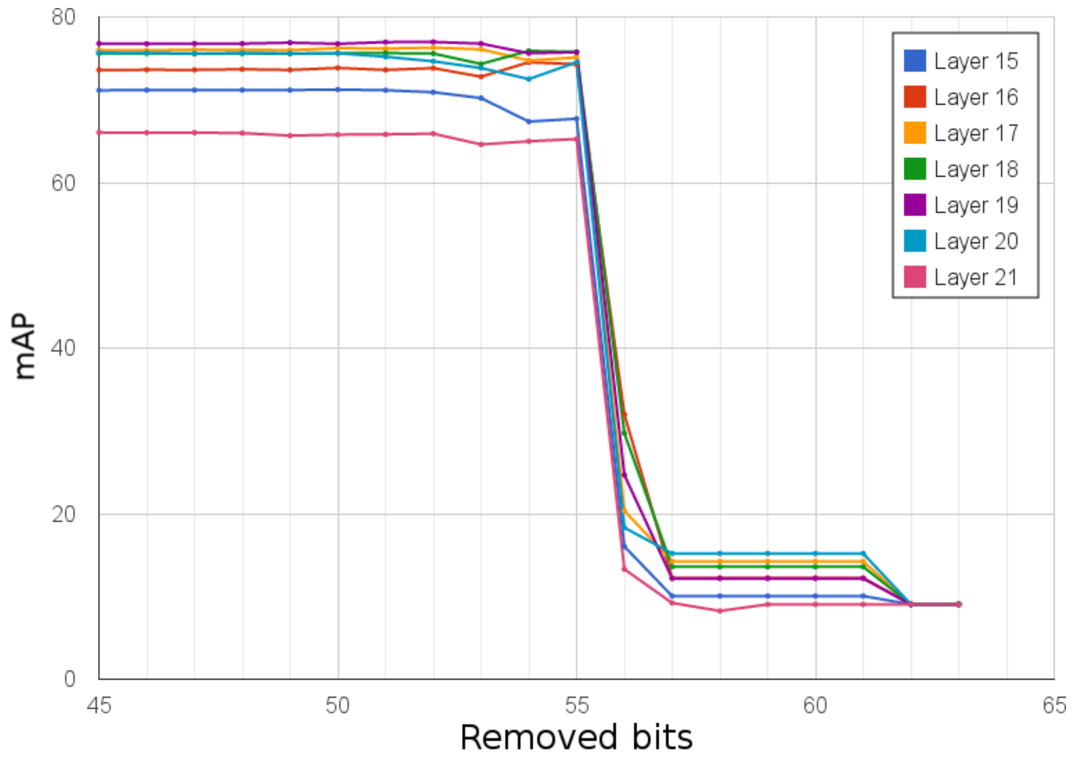


Figure 21 – Results for our Stress 2 experiments, in which we systematically erase rightmost bits from the binary representation of the features, allowing us to investigate how important is the precision for these representations.

slightly at the layers 54 and 55, their difference to the baseline scores was of only 0.91 p.p. on average (0.49 p.p. if we exclude layer 15 from the calculations). A drastic change was observed when we erased the 56^{th} bit, with an average drop of 51.49 p.p., which worsened with further erases, until all information was excluded from the features.

The results from Figure 21 and the information presented in Figure 20 indicate that the entire section reserved for the binary representation of the fraction was erased without significant loss. The fact that the double-precision floating-point format has an implicit bit set to 1 at the beginning of the fraction sector, between bits 51 and 52, allowed us to continue erasing bits without a major drop in the performance, since the fraction was not completely erased during the calculation of its value, given by $(-1)^{sign} \times 1.fraction \times 2^{exponent-bias}$.

Our experiments also indicate that the 8 most significant bits of the exponent and the sign bit play the main role in coding information into the features. These results not only suggest that there is unnecessary precision in deep representations, but also that they may be heavily compressed by exploiting this characteristic. This may be specially useful for portable devices, in which memory and representation size are critical, or in remote classification, in which data must be transferred efficiently over the network.

4.5 Stress 3

In this experiment, we quantize the features extracted from a deep architecture, limiting the number of distinct values for them. This can be seen as a translation between two languages: the features are described in a language X, and we want to rewrite them in language Y; but there are no direct translations for some words, forcing us to translate each word of a text written in X as the one with the closest meaning in Y. We divided this experiment in two categories: **AB**, in which we have one language for all of the features and **CD**, with one language for each dimension of the feature vector. The CD approach is specially useful for cases in which each dimension of the feature vector may be in a different scale. The process of quantization for the AB category is shown at Pseudocode 4.2, and in Pseudocode 4.3 for the CD category.

Pseudocode 4.2 Quantization of the feature vector for the AB category

```
1: function QUANTIZE_FEATURE_VECTOR(feature_vector, new_language)
2:     $fsize \leftarrow size(feature\_vector)$
3:     $lsize \leftarrow size(new\_language)$
4:     $quantized\_features \leftarrow feature\_vector$
5:     for $i = 1$ to $fsize$ do
6:         $winner \leftarrow 1$
7:         $distw \leftarrow feature\_vector[i] - new\_language[winner]$
8:         for $j = 2$ to $lsize$ do
9:             $distj \leftarrow feature\_vector[i] - new\_language[j]$
10:            if $|distj| < |distw|$ then
11:                $winner \leftarrow j$
12:                $distw \leftarrow distj$
13:            end if
14:        end for
15:        $quantized\_features[i] \leftarrow new\_language[winner]$
16:    end for
17:    return $quantized\_features$
18: end function
```

One of our hypothesis, as stated in Section 4.1, is that negative values in the features may be of low importance due to the nature of the activation functions adopted by such models. For this reason, we have subdivided each category into two branches, creating new languages with and without direct translations to negative values.

Pseudocode 4.3 Quantization of the feature vector for the CD category

```
function QUANTIZE_FEATURE_VECTOR(feature_vector, new_language)
    $fsize \leftarrow size(feature\_vector)$
    $quantized\_features \leftarrow feature\_vector$
    for $i = 1$ to $fsize$ do
        $winner \leftarrow 1$
        $distw \leftarrow feature\_vector[i] - new\_language[i][winner]$
        $lsize \leftarrow size(new\_language[i])$
        for $j = 2$ to $lsize$ do
            $distj \leftarrow feature\_vector[i] - new\_language[i][j]$
            if $|distj| < |distw|$ then
                $winner \leftarrow j$
                $distw \leftarrow distj$
            end if
        end for
        $quantized\_features[i] \leftarrow new\_language[i][winner]$
    end for
    return $quantized\_features$
end function
```

Pseudocode 4.4 describes our process for creating a new language, given an interval $[j, k]$ and a number of desired entries (words) n. This process is analogue to creating a one dimensional Voronoi tessellation, and taking the generated points as the new words.

The values of k and j for each branch were chosen as follows:

Stress 3A $j = min(features)$ and $k = max(features)$, where $features$ are all of the features of the training set;

Stress 3B $j = 0$ and $k = max(features)$, where $features$ are all of the features of the training set;

Stress 3C $j[i] = min(features[:][i])$ and $k[i] = max(features[:][i])$, where $features$ are all of the features of the training set, [:] indicate that we are taking all of the training samples and $[i]$ that we are taking only the i^{th} feature from them;

Stress 3D $j[i] = 0$ and $k[i] = max(features[:][i])$, where $features$ are all of the features of the training set, [:] indicate that we are taking all of the training samples and $[i]$ that we are taking only the i^{th} feature from them;

Clearly, the approaches 3C and 3D, which are part of the CD category, generate arrays of values for j and k with sizes equal to the size of a feature vector. This is necessary since we

Pseudocode 4.4 Creating a new language for the quantization process

```
1: function CREATE_LANGUAGE(j, k, n)
2:     $step \leftarrow \frac{k-j}{n}$
3:     $value \leftarrow j + \frac{step}{2}$
4:     $new\_language \leftarrow [\,]$
5:     while $value < k$ do
6:         append $value$ to $new\_language$
7:         $value \leftarrow value + step$
8:     end while
9:     return $new\_language$
10: end function
```

are building a new language for each dimension of it, and the Pseudocode 4.4 will be called once for each element of such arrays. The value of n, indicating the number of words to be used in the new language, is tested for all integer values in the interval $[2, 30]$

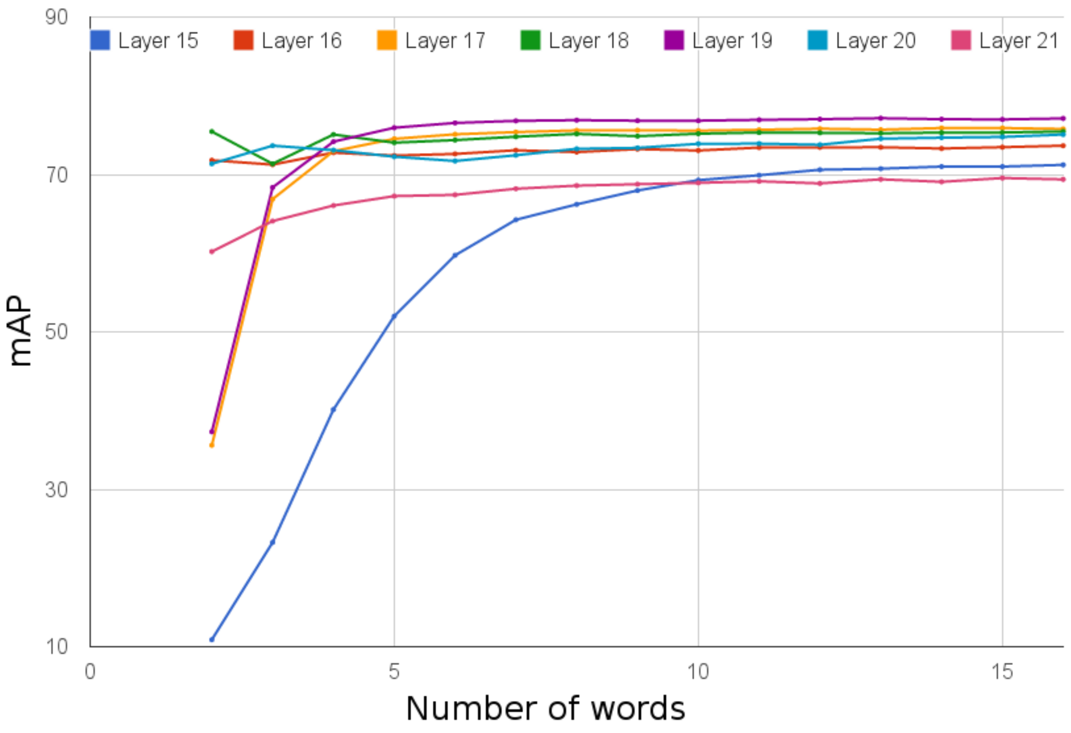
Results and Discussion

Figures 22 and 23 show our results for the Stress 3 experiments, in which we limit the number of different values (words) allowed in a deep representation by a process similar to the coding of a BoW model. The AB group (Figure 22) restricts the values in a global manner, while the CD group (Figure 23) applies a feature-wise restriction.

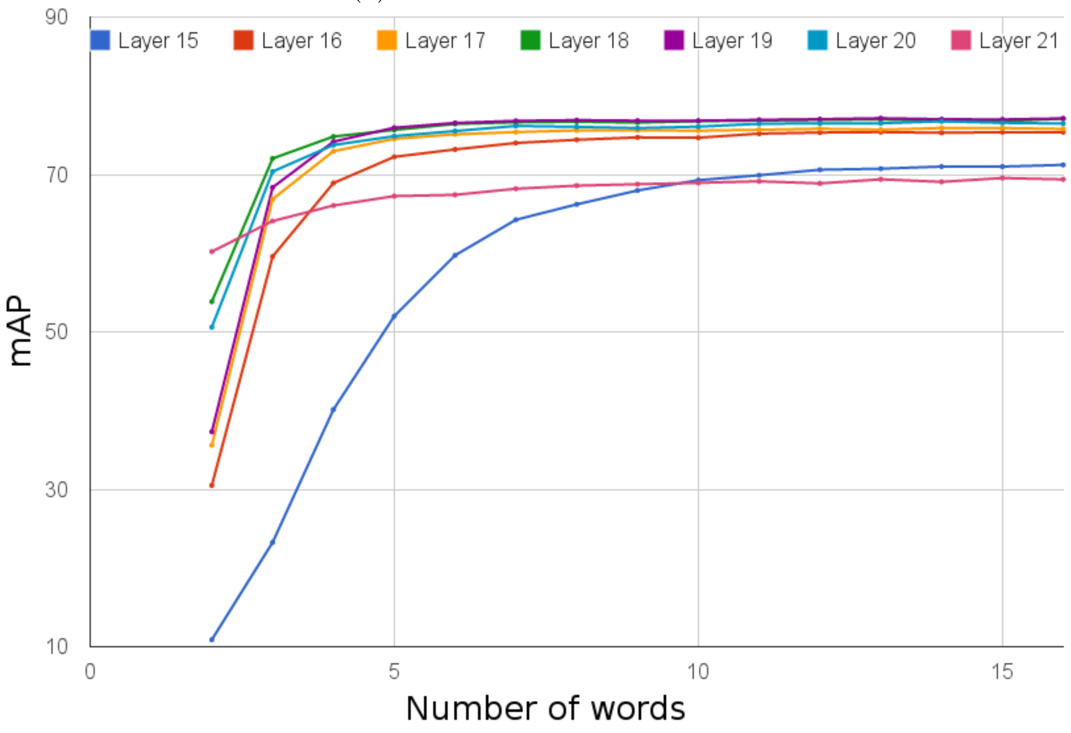
Evidently, the layers 15, 17, 19 and 21 are invariant to the changes from the experiments 3A to 3B and 3C to 3D, which discards negative values, since the first three of the mentioned layers come from a ReLU and the latter from a Softmax layer, which have non-negative outputs only.

With a small number of words ($\leqslant 3$), discarding negative values had a bad impact for the AB experiments, and a good impact for the CD experiments. Because the AB experiment create a single language to redescribe all of the features, different ranges of features may force entire dimensions to be crushed into a single point in the final representation, losing all the information they contain and reducing the classification score. These effects, however, are expected to become smaller as we increase the number of words. We exemplify this situation in Figure 24.

Clearly, the performance of the CD experiments was better than the AB's, indicating that there are dimensions from the feature vectors in very different scales. Discarding negative values also contributed to the performance of the models, as the improvements of the

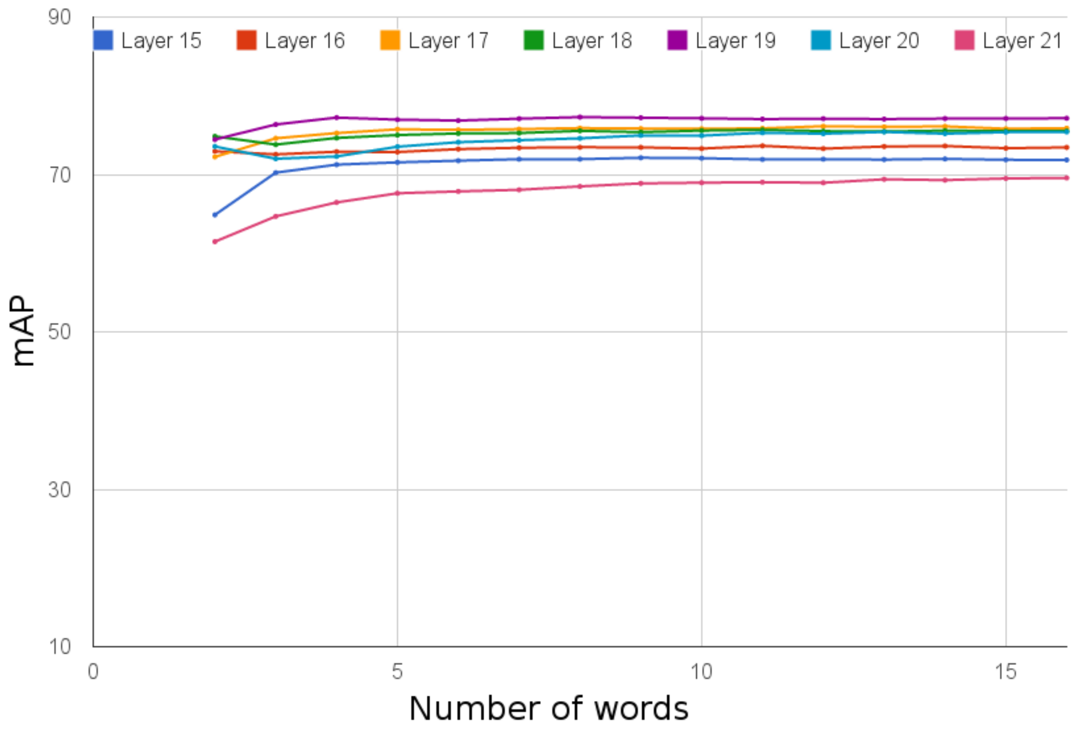


(a) Results for the Stress 3A

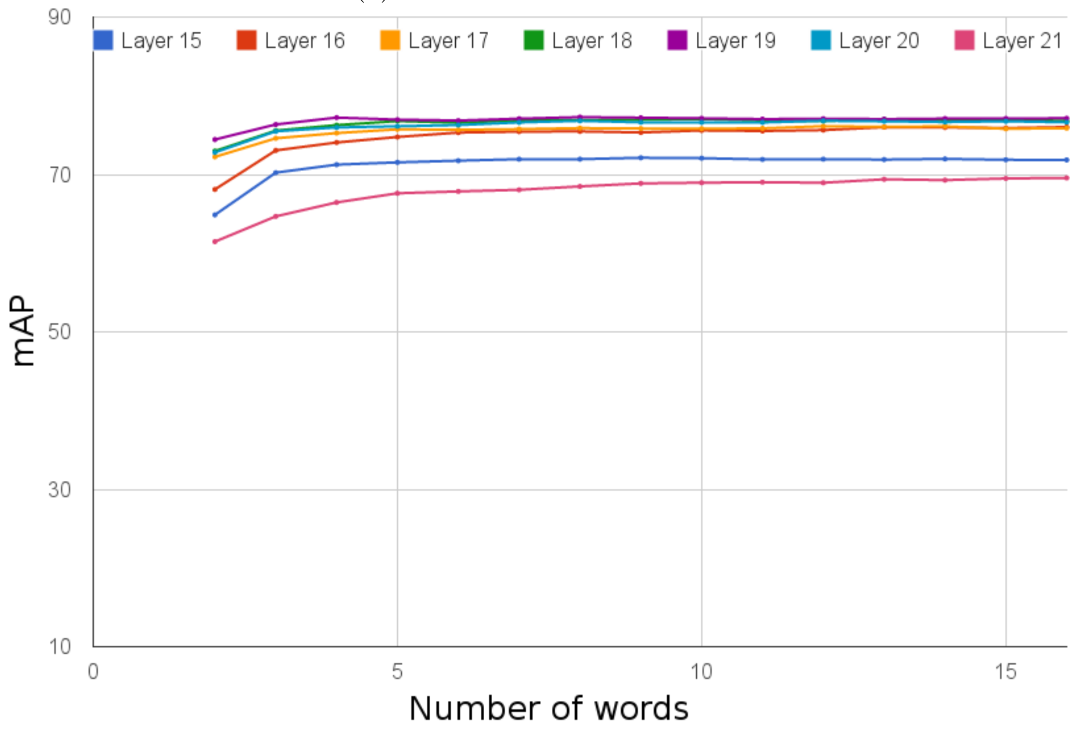


(b) Results for the Stress 3B

Figure 22 – Results for the stresses 3 AB. We create a set of words to redescribe the features extracted from the deep model. In the AB experiments, words are created based on all of the features of the training set. The **3A** branch considers negative values and the **3B** branch discards them.

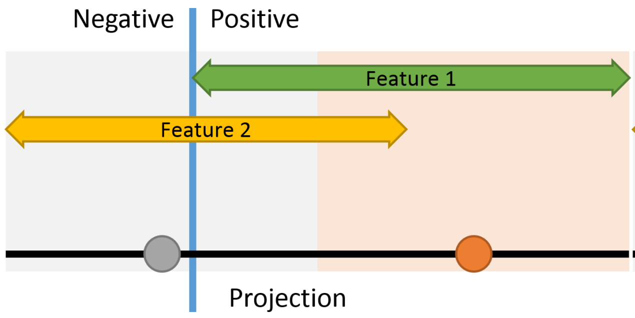


(a) Results for the Stress 3C

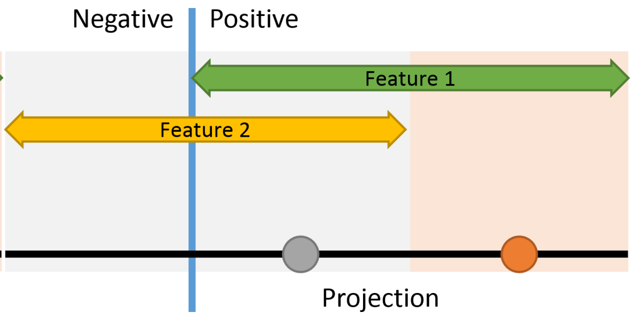


(b) Results for the Stress 3D

Figure 23 – Results for the stresses 3 CD. We create a set of words to redescribe the features extracted from the deep model. In the CD experiments the words are created based on the indexes from the features of the training set. The **3C** branch considers negative values and the **3D** branch discards them.



(a) Quantizing two features with 2 different words, considering negative values.



(b) Quantizing two features with 2 different words, disregarding negative values.

Figure 24 – A quantization problem: when disregarding negative values while choosing the words, the use of a single language to redescribe all of the features may erase large amounts of information.

experiment D (Figure 23b) over the experiment C (Figure 23a) were, on average, 1.57 p.p. bigger for the layers 16, 18 and 20.

The D branch also had the best overall performance of all four experiments. In this setup, 4 words were enough to promote an improvement of 0.67 p.p., on average, compared to the baseline experiments. With 10 words, this number was increased to 1.20 p.p.

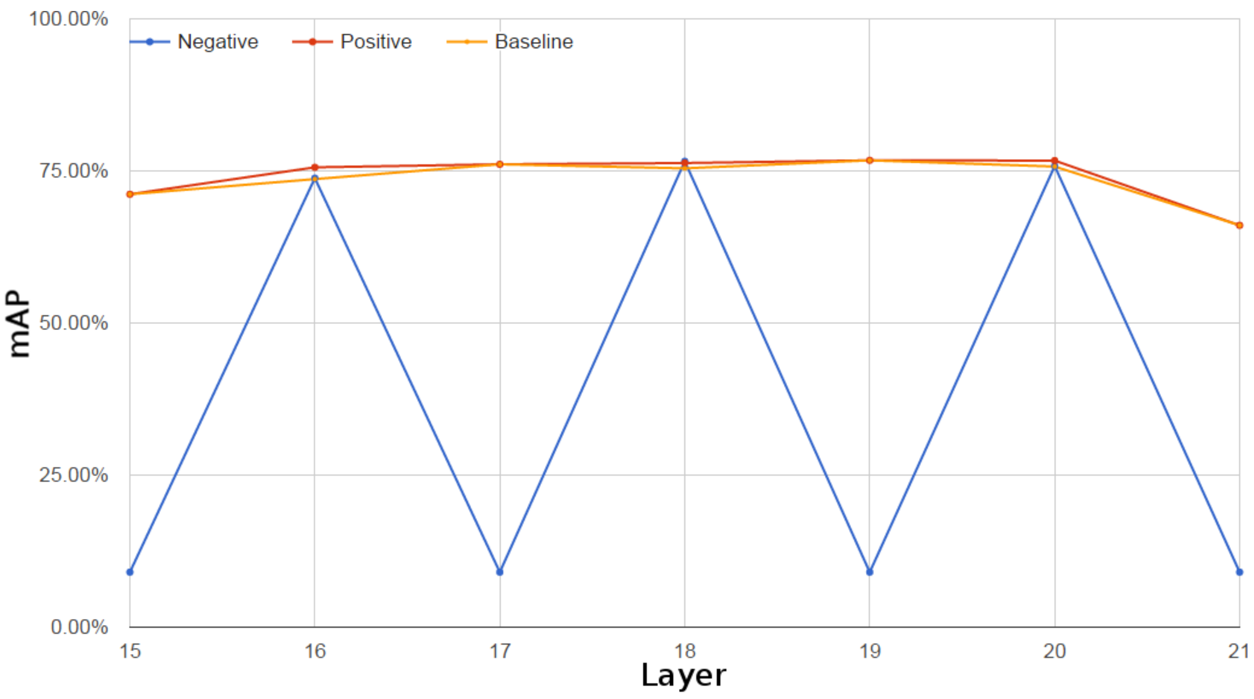


Figure 25 – Removing positive or negative values from the feature vectors. The red line corresponds to the preservation of positive values and, the blue line, the preservation of negative values.

These results lead us to believe that negative values are not as important in deep representations as the non-negative ones. To test this hypothesis, we have repeated the baseline experiments in 2 different manners: (1) setting all negative values to zero; and (2) setting all positive values to zero. The results for this approach are shown in Figure 25, and clearly indicate that, although the negative values have no big influence over the classification performance, they contain enough information to achieve good classification scores. As previously stated, layers 15, 16, 19 and 21 do not contain negative values, causing a severe drop in the mAP. We highlight that the models trained only with non-negative values held the best performance for all layers, with equal performance to the baseline on layers with no negative values in the output. The removal of negative values caused an increase of 1.55 p.p. in the scores, in comparison to the baseline experiments; this is a strong indicative that positive values are highly informative.

4.6 Combined Stresses

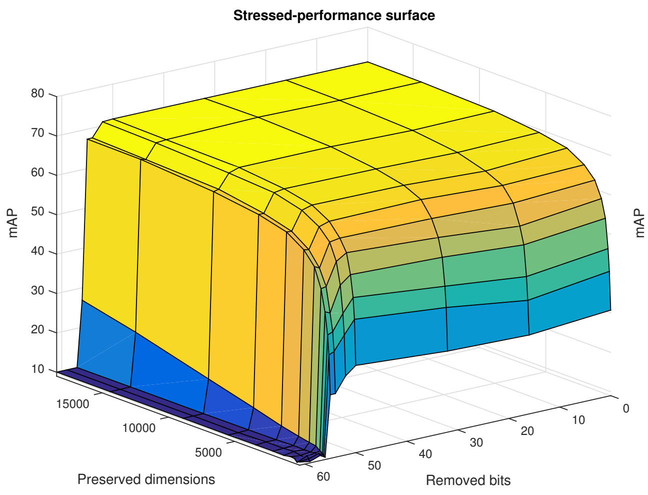
In this section, we merge Stress 1 and Stress 2, testing their performance at the same time (e.g. erasing 16 bits *and* picking 200 dimensions).

In order to decrease the execution time for this experiment, we have reduced the search space inherited from Stress 2 for testing only the removal of 0, 16, 32, 50, 52, 54, 55, 56, 57, 59 and 61 bits. These numbers were chosen based on results obtained for the Stress 2 experiments, presented in Section 4.4.

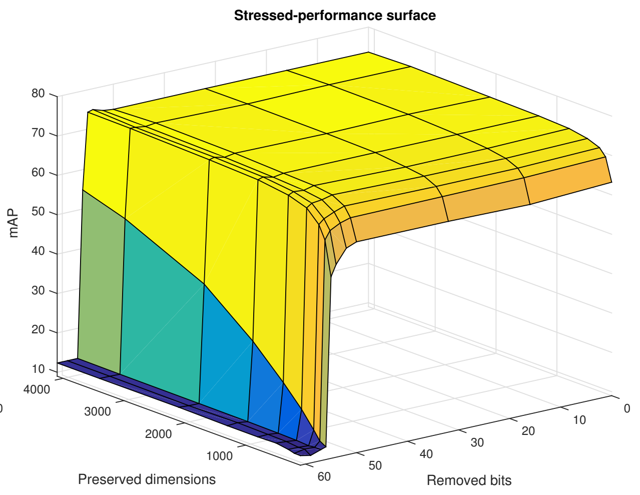
Results and Discussion

Figures 26 and 27 show our results for the combination of Stresses 1 and 2. Because each stress had, initially, 2 dimensions (preserved dimensions + score and removed bits + score), with 1 dimension in common, their combination has 3 dimensions, and is displayed separately for each layer.

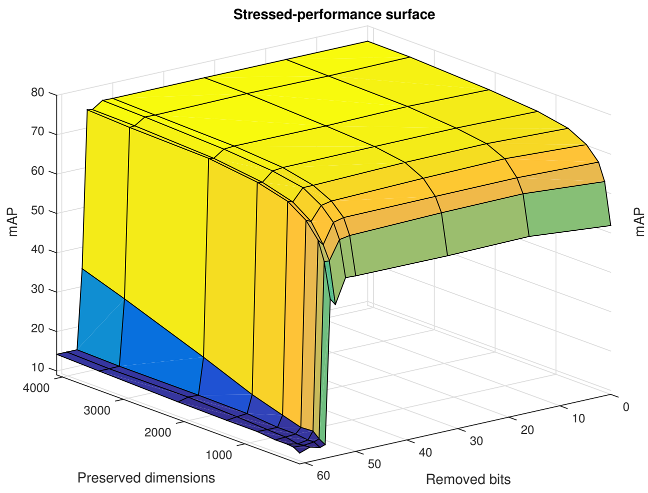
As discussed in Section 4.3, layer 21 may suffer with problems caused by the Softmax operation in the last layer, and its results are in accordance with what we have already seen in Sections 4.3 and 4.4. For the other layers, however, we have detected large regions with high classification scores. These regions correspond to combinations of parameters from Stress 1 and 2 with complementary characteristics, and indicate that deep features can be compressed in terms of dimensions and precision at the same time. We also point that, proportional to their size, higher layers tend to have broader surfaces with high classification scores, suggesting that they may be able to perform better with a reduced number of dimensions.



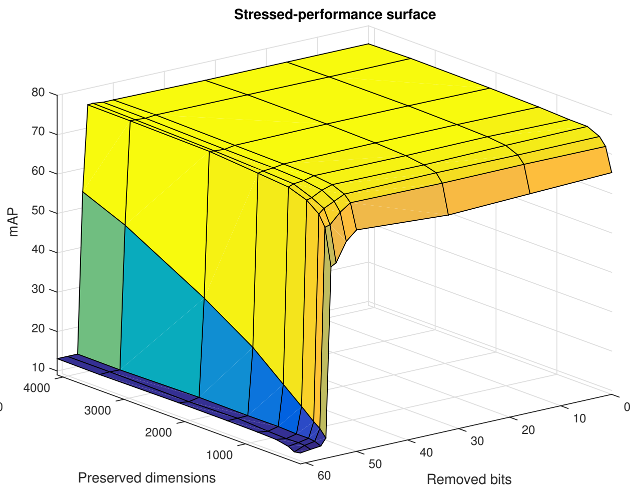
(a) Results of the layer 15 in Combined Stresses



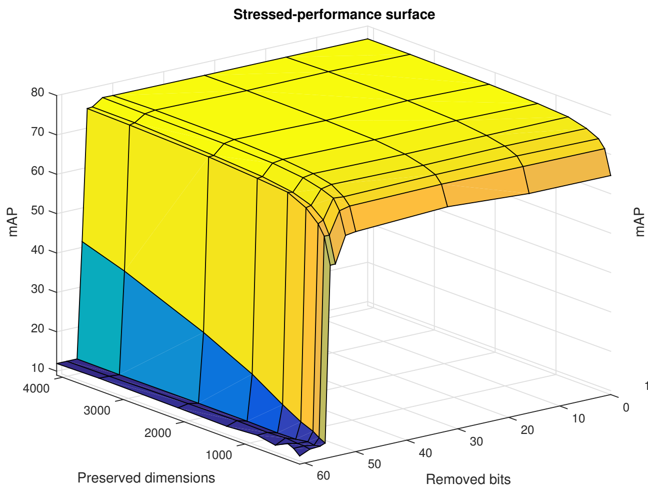
(b) Results of the layer 16 in Combined Stresses



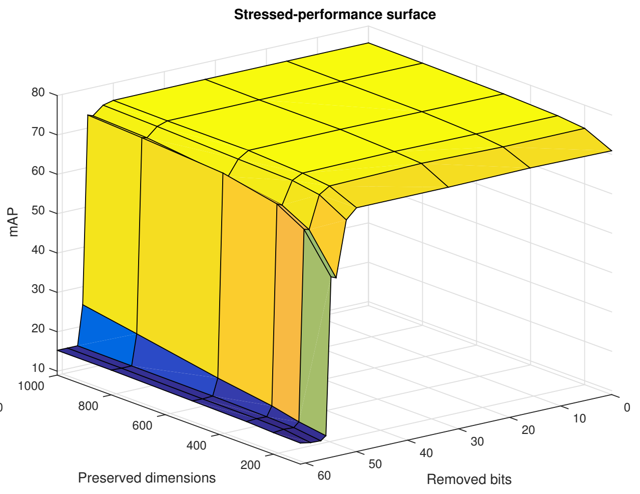
(c) Results of the layer 17 in Combined Stresses



(d) Results of the layer 18 in Combined Stresses



(e) Results of the layer 19 in Combined Stresses



(f) Results of the layer 20 in Combined Stresses

Figure 26 – Results for the Combined Stresses at layers 15-20. We combine the stresses 1 and 2 to verify if the redundancy in the dimensions of the feature vector is independent of the precision (in bits) of their description.

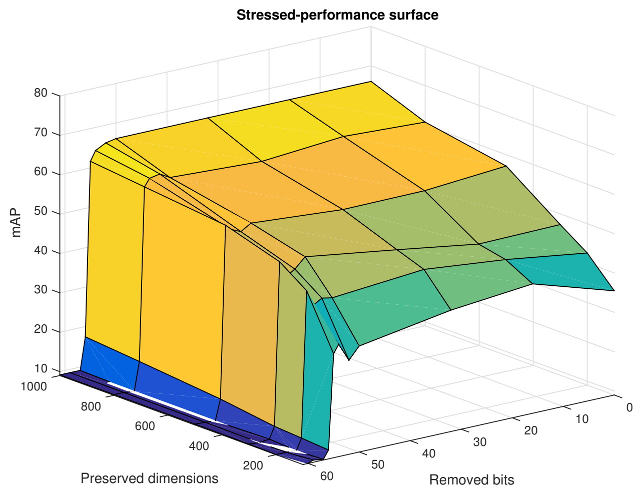


Figure 27 – Results for the Combined Stresses at layer 21. We combine the stresses 1 and 2 to verify if the redundancy in the dimensions of the feature vector is independent of the precision (in bits) of their description.

4.7 Fused Descriptors

In this experiment, we combine features from different layers in attempt to create a description with increased quality. For this purpose, we adopt two different approaches:

1. **Late fusion**. The proposal behind late fusion is to fuse information in a semantic-level. For this approach we train the classifier with the feature vectors extracted from the deep model, taking their scores (instead of labels) as output. The scores of two different layers $\mathbf{s_1}$ and $\mathbf{s_2}$ are then fused in three combinations, by calculating: (1) their sum $\mathbf{s_{sum}} = \mathbf{s_1} + \mathbf{s_2}$; (2) their average $\mathbf{s_{avg}} = 0.5 \cdot \mathbf{s_1} + 0.5 \cdot \mathbf{s_2}$ and (3) the max value for each position of the score vector $s_{max,i} = max(s_{1,i}, s_{2,i})$.

   For each one of the three combinations, a value $0.9 \leqslant t \leqslant 0.1$ is used to decide if the label is positive $s_i \geqslant t$ or negative $s_i < t$; this value is determined by maximizing the mAP in the validation set and, because this is not a convex optimization, we test all of the values in the interval $[0.1, 0.9]$ with step 0.1.

   Finally, the scheme chosen as representative for the late fusion of two layers is the one with the highest mAP in the validation set, and the mAP in the test set is calculated using the same combination and the same value of t;

2. **Early fusion**, also known as feature-level fusion. In this approach, we concatenate feature vectors extracted from two different layers, prior to training the classifier. This procedure has potential to increase the amount of information preserved in the descrip-

tion by incorporating, for example, spatial characteristics, at the cost of also increasing its size.

Results and Discussion

Tables 8 and 9 show our results for Late Fusion and Early Fusion, respectively. We highlight, in blue, the best results in each of them.

Table 8 – Late fusion of different layers. The layers were combined at semantic level: the scores given by the classifiers of two levels were fused before deciding on the label.

Layer	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**
4	–	35.6	40.6	40.5	45.2	38.7	46.7	48.5	53.7	51.1	64.7	67.3	69.6	67.7	71.1	69.5	65.2	62.8
5	35.6	–	40.6	41.0	45.5	39.8	48.3	50.0	55.0	55.8	64.9	68.0	69.1	69.9	71.4	70.8	64.6	59.5
6	40.6	40.6	–	44.3	45.3	41.8	49.2	47.7	55.6	55.3	65.6	68.8	65.2	71.7	66.8	69.3	65.2	60.2
7	40.5	41.0	44.3	–	45.3	42.2	49.8	48.4	56.4	56.5	66.3	69.2	64.4	71.2	67.0	71.1	68.6	60.7
8	45.2	45.5	45.3	45.3	–	45.8	49.8	49.9	53.9	53.9	65.3	68.2	63.3	65.4	66.8	67.2	61.2	58.0
9	38.7	39.8	41.8	42.2	45.8	–	50.4	52.2	57.0	58.1	65.7	69.5	71.4	73.4	73.7	74.9	72.8	66.9
10	46.7	48.3	49.2	49.8	49.8	50.4	–	50.6	57.4	54.4	66.4	69.1	67.4	70.8	68.3	71.9	66.1	63.4
11	48.5	50.0	47.7	48.4	49.9	52.2	50.6	–	57.6	54.4	65.9	69.7	69.3	72.0	72.7	72.3	70.6	63.9
12	53.7	55.0	55.6	56.4	53.9	57.0	57.4	57.6	–	59.4	66.6	69.2	63.9	70.1	66.6	69.6	66.6	63.4
13	51.1	55.8	55.3	56.5	53.9	58.1	54.4	54.4	59.4	–	67.8	70.2	70.6	72.7	71.0	71.5	70.8	67.6
14	64.7	64.9	65.6	66.3	65.3	65.7	66.4	65.9	66.6	67.8	–	70.4	70.8	73.3	72.6	74.3	71.9	70.8
15	67.3	68.0	68.8	69.2	68.2	69.5	69.1	69.7	69.2	70.2	70.4	–	69.5	73.2	69.7	71.3	70.5	71.0
16	69.6	69.1	65.2	64.4	63.3	71.4	67.4	69.3	63.9	70.6	70.8	69.5	–	74.5	74.4	75.4	72.6	74.9
17	67.7	69.9	71.7	71.2	65.4	73.4	70.8	72.0	70.1	72.7	73.3	73.2	74.5	–	74.8	75.3	74.8	71.7
18	71.1	71.4	66.8	67.0	66.8	73.7	68.3	72.7	66.6	71.0	72.6	69.7	74.4	74.8	–	73.5	74.2	75.8
19	69.5	70.8	69.3	71.1	67.2	74.9	71.9	72.3	69.6	71.5	74.3	71.3	75.4	75.3	73.5	–	74.8	75.3
20	65.2	64.6	65.2	68.6	61.2	72.8	66.1	70.6	66.6	70.8	71.9	70.5	72.6	74.8	74.2	74.8	–	75.4
21	62.8	59.5	60.2	60.7	58.0	66.9	63.4	63.9	63.4	67.6	70.8	71.0	74.9	71.7	75.8	75.3	75.4	–

Table 9 – Early fusion of features from different layers. The feature vectors were concatenated forming a descriptor with richer information and bigger size.

Layer	**21**	**20**	**19**	**18**	**17**	**16**	**15**
21	–	74.4	76.1	74.8	76.3	75.2	74.4
20	74.4	–	76.8	76.4	77.0	76.4	76.0
19	76.1	76.8	–	77.1	77.3	76.7	76.9
18	74.8	76.4	77.1	–	76.7	76.4	76.4
17	76.3	77.0	77.3	76.7	–	76.4	76.4
16	75.2	76.4	76.7	76.4	76.4	–	73.1
15	74.4	76.0	76.9	76.4	76.4	73.1	–

In the context of our experiments, we find no evidence to support the use of late fusion, since the best result achieved by it (75.8%) is still inferior to the results from 2 layers of the baseline (76.09% for layer 17 and 76.74% for layer 19) and very close to the results obtained by layer 20 (75.72%).

We point, however, that the best mean scores were achieved by layers 19 and 17, with averages of 72.23% and 71.91%, suggesting that they could be further explored with other setups. The overall best result for this approach was achieved by the fusion of layers 18 and 21, indicating that they may be encoding complementary information, and could also be further explored, despite the controversial performance of layer 21 in the previous experiments.

As for the early fusion, many pairs achieved better performance than the baseline experiments ($\{15, 19\}$, $\{17, 19\}$, $\{17, 20\}$, $\{18, 19\}$ and $\{19, 20\}$), and the best score for it (the fusion of layers 17 and 19) offered an increase of 0.55% over our best baseline result. We also observed that layer 19 had an average slightly better than the baseline, although not of much significance (0.07%).

Our results reveal that the application of early fusion, although expensive, because of the increased size of the feature vector, may be rewarding, since small improvements can be of great significance for many tasks (as in the case of automated screening). Furthermore, the adoption of strategies based on high level representations, discussed in the next section, may be complementary to other strategies discussed in this work, suggesting their combination.

4.8 High Level Representations

For the experiments of this section, we draw inspiration from the Object Bank representation and the Spatial Pyramid Matching method, aiming to improve the quality of the feature vectors extracted from a deep model. Although our approaches do not follow exactly their steps, both are similar in many points, as we describe below.

Scale Invariance A follows an approach similar to the Object Bank representation, which uses a series of object detectors to describe an image with respect to the activation of such detectors. We consider, however, the deep model as a multi-object detector, and its output as the activation for the many objects.

For each image, we have extracted square patches (regions) corresponding to $\frac{1}{5}$, $\frac{3}{10}$, $\frac{1}{3}$, $\frac{2}{5}$ and $\frac{1}{2}$ of its original size, with steps of $\frac{1}{10}$, $\frac{3}{20}$, $\frac{1}{6}$, $\frac{1}{5}$ and $\frac{1}{4}$, respectively. These regions were presented to the deep model and their feature vectors were combined into a single description, obtained by the element-wise *max* operation, which was then presented to the classifier.

Compared to a simple transfer approach, this setup may offer better scale invariance, since the objects of interest may be presented many times in the patches, at different scales.

Scale Invariance B tries to capture part of the invariance of the Spatial Pyramid Matching method, which introduces spatial information in the features by the use of regions extracted from the image. In our experiments, however, we adopt only the pyramid based patches from SPM; and instead of introducing spatial information to the features, we further explore invariances to scale that can be achieved by the different levels of the pyramid.

For this reason, we devised two branches of this experiment. The first, adopts pyramids of 1×1, 2×2 and 4×4, and the second adopts pyramids of 1×1, 2×2 and 3×1. These configurations were chosen based on the nature of the dataset in which we are experimenting (Pascal VOC 2007).

At each level of the pyramid, we have extracted patches and presented them to our Scale Invariance A approach with the $\frac{1}{3}$ configuration, combining the resulting feature vectors with the element-wise *max* operation. The three feature vectors obtained by the different levels of the pyramid were then concatenated, creating a single description.

Compared to the Scale Invariance A approach, this setup offer richer features, with better scale invariance, since the information was extracted at different levels and is still preserved by the concatenation in the final representation.

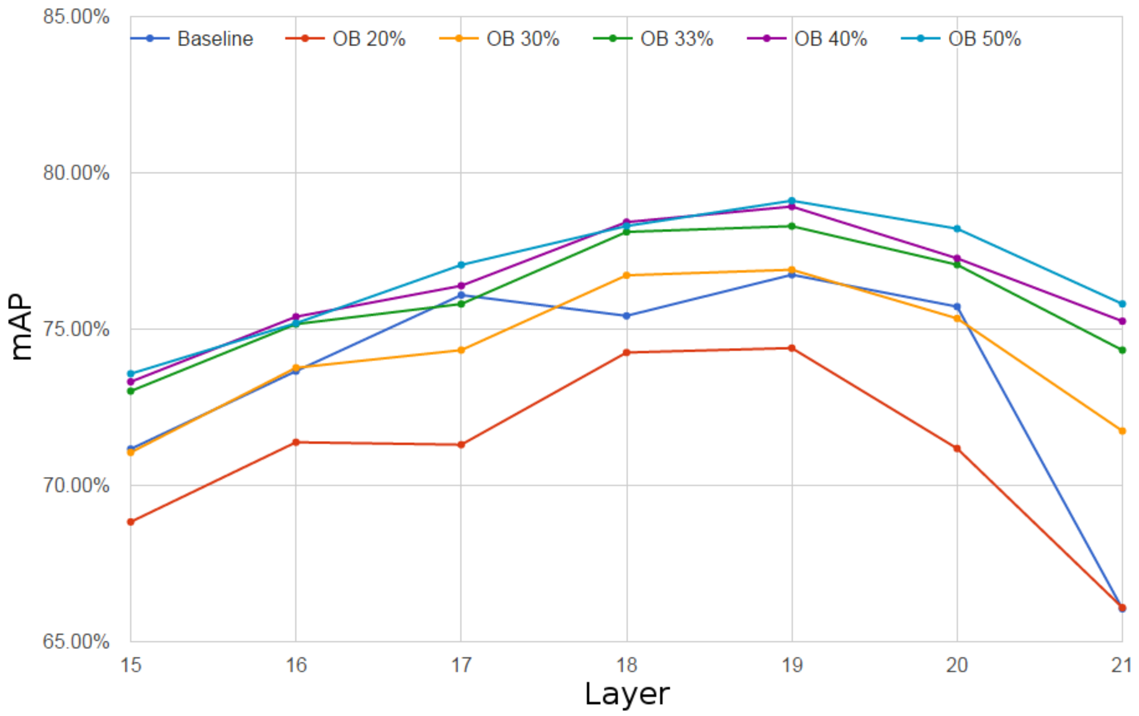
Results and Discussion

Figure 28 shows our results for the Scale Invariance A experiments (Figure 28a) and Scale Invariance B experiments (Figure 28b).

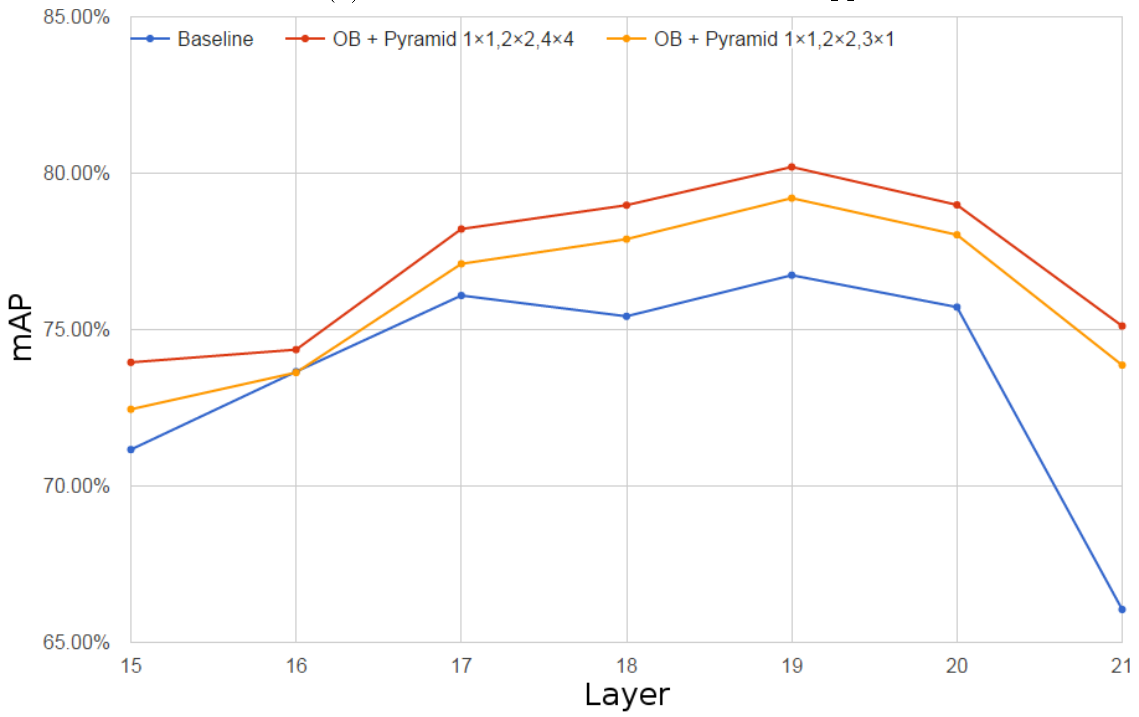
For the Scale Invariance A experiments, all configurations except for the $\frac{1}{5}$ resulted in improved scores. The best numbers were achieved by the $\frac{1}{2}$ setup, which had an average increase of 3.20 p.p. in the classification score over the baseline experiments (and 0.72 p.p. for $\frac{1}{5}$, 2.42 p.p. for $\frac{3}{10}$ and 2.87 p.p. for $\frac{2}{5}$).

These results indicate that many of the objects of interest in our dataset may be detected by looking at regions with area equivalent to $\frac{1}{4}$ of the total area of the image. Intriguing results were also achieved for the layer 21, which had the biggest increase for our experiments, with 9.79 p.p. above the baseline results.

As for our Scale Invariance B results, both setups affected positively the final score, with average increases of 3.57 p.p. for the first branch and 2.48 p.p. for the second. The first



(a) Results for our Scale Invariance A approach



(b) Results for our Scale Invariance B approach

Figure 28 – Results for the High Level Representations. We have created two experiments inspired by the Object Bank approach and the Spatial Pyramid Matching method, aiming to embed scale invariance to the feature vector.

branch achieved a score of 80.20% on the layer 19, which is the best mAP across all of our experiments.

Since these approaches preserve more information than the Scale Invariance A (which is equivalent to the first pyramid alone), we have reason to believe they can be further improved by adopting different scales for the Object Bank-like description and more levels for the pyramids.

4.9 Conclusion

In this chapter, we have presented our experiments which explored internal representations of a deep model, aiming at better understanding them and improving their discriminability.

In Section 4.3, we've shown that representations extracted from deep architectures can contain redundant properties, with information replicated across different feature indexes. Our results suggest that they can be strongly compressed, without significant loss in performance. We've also shown, in Section 4.4, that the precision in the description extracted from such models is much higher than the necessary, when working with transfer learning. After removing more than 81% of their binary representation, we still could not detect significant change in the scores.

These effects were revealed to be complementary, as we successfully reduced the precision of features extracted from a deep model and, at the same time, exploited redundancy in their dimensions (Section 4.6).

To better comprehend the extent of our findings, we have further limited the values of deep representations in Section 4.5, showing that 3 different values for each feature were enough to achieve results similar to our baseline experiments. In the same section, it was shown that positive values in the description are highly informative, and their use incremented our scores by 1.55 p.p.

In attempt to increase the amount of information coded into these representations, we have tested fusion strategies (Section 4.7), detecting slight improvements with the adoption of an Early Fusion method. Finally, we have discovered that approaches inspired by the Object Bank representation and the Spatial Pyramid method, often associated to BoVW-like strategies, can offer significant enhancements to the quality of the features. Our experiments from Section 4.8 detected an average gain of 3.57 p.p. over the baseline results.

Although deep representations have been applied with much success in many image recognition challenges and tasks, our results suggest that most of their internal representations is very redundant, even though bigger networks tend to achieve better performance when compared to smaller architectures. It remains to be tested whether the effects we have observed are also valid for their original task, rather than being particular to the transfer approach adopted in our experiments.

5 Conclusions

In this chapter, we summarize the main contributions of this work regarding the problem of automated melanoma screening and our exploration of deep features. Furthermore, we provide guidelines for future work, highlighting points that could not be thoroughly explored due to limitations in time and scope.

5.1 Main Contributions

The experiments we describe in this work revealed interesting aspects of the representations created by deep architectures, and are yet to be published. Among our many findings, we emphasize contributions related to:

Automated Melanoma Screening: In Chapter 3, we have shown robust approaches to the problem of automated melanoma screening, with state-of-the-art results. Contrary to its current art, our proposal is independent from medical classification procedures, relying only on images of the lesion. We highlight that our setup can be efficiently optimized to be used in remote regions, where the presence of a full-time dermatologist is not economically feasible, and that the automation of the melanoma screening would be able to aid in the identification of special cases, for which the attention of a specialist is needed. We also emphasize that part of our exploration of the melanoma problem, involving the use of BoVW model and in cooperation with Fornaciali et al. [2014], resulted in the following publication:

*Fornaciali, M., Avila, S., **Carvalho, M.**, & Valle, E. (2014). Statistical Learning Approach for Robust Melanoma Screening. In Proceedings of the 2014 27th SIBGRAPI Conference on Graphics, Patterns and Images (pp. 319–326). IEEE Computer Society.*

Deep Representations: Our results are valuable for better understanding the properties of representations extracted from deep models, with evidences pointing to strong internal redundancy both in terms of precision and number of dimensions, resembling the neural redundancy commonly observed in animals. We have also identified complementary characteristics that can be exploited in order to obtain compact representations, without significant loss in the classification score.

Enhanced Features: The adoption of techniques inspired by state-of-the-art representations, such as the Object Bank approach and the Spatial Pyramid Matching scheme, has proved to offer notable benefits to the process of transferring knowledge constructed by deep architectures. We have detected gains of up to 9.79 p.p. for specific configurations, with an average increase of 3.57 p.p. with our best strategy. Fusion methods are also shown to be capable of creating richer features, by combining information from multiple layers. We have reason to believe that many of our findings can be successfully combined to generate powerful image descriptors.

5.2 Open Questions and Future Work

In this section, we present questions we could not explore in this work due to limitations of time and scope. We also point further investigations that could be conducted in light of our results.

Automated Melanoma Screening: We have seen that deep features are powerful enough to achieve state-of-the-art results for automated melanoma screening. We believe, however, that our results can be further improved by different experimental setups, such as the use of other models (e.g. the MatConvNet M) and classifiers. The experiments we describe in Chapter 4 can also be employed to increase the quality of the feature vectors, with immediate impacts to the screening problem.

Moreover, recent investigations by Yosinski et al. [2014] point fine-tuning as a promising solution to correct small problems in the extracted features. This topic remains a prominent investigation for the melanoma screening community.

Finally, we believe that ensuring reproducibility for the proposed methods are of extreme importance, and should be a concern for any research in this area. Except for the proposal of Fornaciali et al. [2014], we could not successfully reproduce any results from the melanoma literature, hindering a fair comparison between different approaches.

Enhanced Features: Our results from Section 4.8 suggest that small changes to our experiments, such as increasing the size of the sliding window, could lead to better scores. Additionally, data augmentation techniques, which aim to increase the size of the training set and gain invariance to a series of transformations, could not be explored due to limitations in the scope of this project. We believe that such approaches may be complementary, and their combination can further increase the quality of the representations extracted from a deep model.

Bibliography

Abbas, Q., Celebi, M., Garcia, I., and Ahmad, W. Melanoma recognition framework based on expert definition of ABCD for dermoscopic images. *Skin Research and Technology*, 19 (1):93–102, 2012. Cited 3 times in pages 40, 41, and 51.

Aizerman, A., Braverman, E. M., and Rozoner, L. I. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and remote control*, 25:821–837, 1964. Cited in page 30.

Arandjelovic, R. and Zisserman, A. Three things everyone should know to improve object retrieval. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2911–2918, 2012. Cited in page 39.

Argenziano, G. and Soyer, H. P. Dermoscopy of pigmented skin lesions – a valuable tool for early diagnosis of melanoma, 2001. ISSN 14702045. Cited in page 40.

Argenziano, G., Fabbrocini, G., Carli, P., De Giorgi, V., Sammarco, E., and Delfino, M. Epiluminescence microscopy for the diagnosis of doubtful melanocytic skin lesions. Comparison of the ABCD rule of dermatoscopy and a new 7-point checklist based on pattern analysis. *Archives of dermatology*, 134(12):1563–1570, 1998. ISSN 0003987X. Cited in page 40.

Avila, S. *Extended Bag-of-Words Formalism for Image Classification.* Phd in computer science, Federal University of Minas Gerais and Pierre and Marie Curie University, 2013. Cited 3 times in pages 28, 29, and 30.

Avila, S., Thome, N., Cord, M., Valle, E., and De A. Araújo, A. Pooling in Image Representation: The Visual Codeword Point of View. *Computer Vision and Image Understanding*, 117(5):453–465, 2013. Cited in page 39.

Aytar, Y. and Zisserman, A. Tabula rasa: Model transfer for object category detection. *International Conference on Computer Vision (ICCV)*, pages 2252–2259, 2011. Cited in page 35.

Baeza-Yates, R. A. and Ribeiro-Neto, B. *Modern Information Retrieval.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999. ISBN 020139829X. Cited in page 23.

Baldi, A., Murace, R., Dragonetti, E., Manganaro, M., Guerra, O., Bizzi, S., and Galli, L. Definition of an automated Content-Based Image Retrieval (CBIR) system for the comparison of dermoscopic images of pigmented skin lesions. *Biomedical engineering online*, 8:18, 2009. ISSN 1475-925X. Cited in page 40.

Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding (CVIU)*, 110(3):346–359, 2008. Cited in page 39.

Bengio, Y. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009. Cited 2 times in pages 24 and 35.

Bengio, Y. and LeCun, Y. Scaling learning algorithms towards AI. *Large-scale kernel machines*, 34:1–41, 2007. Cited 2 times in pages 24 and 35.

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., and Bengio, Y. Theano: a CPU and GPU math compiler in Python. *Proc. 9th Python in Science Conference (SCIPY 2010)*, (Scipy):1–7, 2010. Cited in page 48.

Boureau, Y.-L., Bach, F., LeCun, Y., and Ponce, J. Learning mid-level features for recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. Cited in page 23.

Campbell, J. E. and Shin, M. Geographic Information System Basics, 2012. URL `http://2012books.lardbucket.org/books/geographic-information-system-basics/`. Cited in page 60.

Chang, C. C. and Lin, C. J. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27:1–27:27, 2011. Cited in page 48.

Chatfield, K., Simonyan, K., Vedaldi, A., and Zisserman, A. Return of the Devil in the Details: Delving Deep into Convolutional Nets. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014. Cited 3 times in pages 27, 43, and 55.

da Silva, I. N., Spatti, D. H., and Flauzino, R. A. *Redes Neurais Artificiais para engenharia e ciências aplicadas: curso prático*. Artliber, 2010. ISBN 978-85-88098-53-4. Cited in page 31.

Di Leo, G., Paolillo, A., Sommella, P., and Fabbrocini, G. Automatic diagnosis of melanoma: A software system based on the 7-point check-list. In *Proceedings of the Annual Hawaii*

International Conference on System Sciences, 2010. ISBN 9780769538693. Cited in page 40.

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. *International Conference on Machine Learning*, pages 647–655, 2014. Cited 3 times in pages 35, 39, and 43.

Duda, R. O., Hart, P. E., and Stork, D. G. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000. ISBN 0471056693. Cited 2 times in pages 29 and 30.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2): 303–338, 2010. Cited 4 times in pages 35, 42, 46, and 55.

Fornaciali, M., Avila, S., Carvalho, M., and Valle, E. Statistical Learning Approach for Robust Melanoma Screening. In *2014 27th SIBGRAPI Conference on Graphics, Patterns and Images*, pages 319–326. IEEE Computer Society, 2014. ISBN 978-1-4799-4258-9. Cited 6 times in pages 40, 41, 42, 51, 77, and 78.

Fukushima, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 202, 1980. Cited 2 times in pages 32 and 34.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 580–587. IEEE, 2014. Cited 3 times in pages 35, 38, and 44.

Gniadecka, M., Philipsen, P. A., Sigurdsson, S., Wessel, S., Nielsen, O. F., Christensen, D. H., Hercogova, J., Rossen, K., Thomsen, H. K., Gniadecki, R., Hansen, L. K., and Wulf, H. C. Melanoma Diagnosis by Raman Spectroscopy and Neural Networks: Structure Alterations in Proteins and Lipids in Intact Cancer Tissue. *Journal of Investigative Dermatology*, 122 (2):443–449, 2004. ISSN 0022202X. Cited 2 times in pages 39 and 45.

Gola Isasi, A., García Zapirain, B., and Méndez Zorrilla, A. Melanomas non-invasive diagnosis application based on the ABCD rule and pattern recognition image processing algorithms. *Computers in Biology and Medicine*, 41(9):742–755, 2011. ISSN 00104825. Cited in page 40.

Gong, Y., Wang, L., Guo, R., and Lazebnik, S. Multi-scale Orderless Pooling of Deep Convolutional Activation Features. *arXiv preprint 1403.1840*, 2014. ISSN 16113349. Cited 3 times in pages 35, 36, and 43.

Hammer, B. and Gersmann, K. A note on the universal approximation capability of support vector machines. *Neural Processing Letters*, 17(1):43–53, 2003. ISSN 13704621. Cited in page 23.

Haykin, S. *Neural networks and learning machines.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 3 edition, 2009. ISBN 0131471392. Cited 2 times in pages 23 and 32.

Hebb, D. O. *The Organization of Behavior: A Neuropsychological Theory.* Wiley, New York, 1949. ISBN 0805843000. Cited in page 30.

Hinton, G. E., Osindero, S., and Teh, Y.-W. A Fast Learning Algorithm for Deep Belief Nets. *Neural computation*, 18(7):1527–1554, 2006. Cited in page 34.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv: 1207.0580*, pages 1–18, 2012. Cited in page 32.

Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982. Cited in page 32.

Iandola, F., Moskewicz, M., Karayev, S., Girshick, R., Darrell, T., and Keutzer, K. DenseNet: Implementing Efficient ConvNet Descriptor Pyramids. *arXiv preprint 1404.1869*, pages 1–11, 2014. Cited 2 times in pages 35 and 38.

Iyatomi, H., Oka, H., Celebi, M. E., Hashimoto, M., Hagiwara, M., Tanaka, M., and Ogawa, K. An improved Internet-based melanoma screening system with dermatologist-like tumor area extraction algorithm. *Computerized Medical Imaging and Graphics*, 32(7):566–579, 2008. ISSN 08956111. Cited 3 times in pages 40, 41, and 51.

Jégou, H., Douze, M., Schmid, C., and Pérez, P. Aggregating local descriptors into a compact image representation. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3304–3311, 2010. Cited in page 39.

Jerant, A., Johnson, J., Sheridan, C., and Caffrey, T. Early detection and treatment of skin cancer. *American Family Physician*, 62(2):357–368, 2000. Cited 2 times in pages 39 and 45.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. Caffe. *Proceedings of the ACM International Conference on Multimedia - MM '14*, pages 675–678, 2014. Cited in page 35.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems (NIPS 2012)*, pages 1–9, 2012. Cited 2 times in pages 27 and 35.

Lazebnik, S., Schmid, C., and Ponce, J. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2169–2178, 2006. Cited 4 times in pages 25, 36, 39, and 55.

Le, Q. V., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G. S., Dean, J., and Ng, A. Y. Building high-level features using large scale unsupervised learning. *International Conference in Machine Learning*, page 38115, 2011. ISSN 10535888. Cited in page 34.

Lecun, Y. and Bengio, Y. *Convolutional Networks for Images, Speech and Time Series*, In *The Handbook of Brain Theory and Neural Networks*, pages 255–258. The MIT Press, 1995. Cited in page 32.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. Cited in page 39.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *Nature*, 521(7553):436–444, 2015. ISSN 0028-0836. Cited 2 times in pages 32 and 33.

Li, L.-j., Su, H., Xing, E. P., and Fei-fei, L. Object Bank : A High-Level Image Representation for Scene Classification & Semantic Feature Sparsification. *Advances in neural information processing systems*, pages 1–9, 2010. Cited 2 times in pages 25 and 55.

Liu, L., Wang, L., and Liu, X. In defense of soft-assignment coding. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2486–2493, November 2011. Cited in page 41.

Lowe, D. G. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. ISSN 09205691. Cited 2 times in pages 23 and 39.

McCulloch, W. S. and Pitts, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133, 1943. Cited in page 30.

Minsky, M. and Papert, S. *Perceptrons*. MIT Press, Cambridge, MA, 1969. Cited in page 31.

Monzani, E., Facchetti, F., Galmozzi, E., Corsini, E., Benetti, A., Cavazzin, C., Gritti, A., Piccinini, A., Porro, D., Santinami, M., Invernici, G., Parati, E., Alessandri, G., and La Porta, C. A. M. Melanoma contains CD133 and ABCG2 positive cells with enhanced tumourigenic potential. *European Journal of Cancer*, 43(5):935–946, 2007. ISSN 09598049. Cited in page 39.

Nachbar, F., Stolz, W., Merkle, T., Cognetta, A. B., Vogt, T., Landthaler, M., Bilek, P., Braun-Falco, O., and Plewig, G. The ABCD rule of dermatoscopy. *Journal of the American Academy of Dermatology*, 30(4):551–559, April 1994. ISSN 01909622. Cited in page 40.

Oh, K.-S. and Jung, K. GPU implementation of neural networks. *Pattern Recognition*, 37 (6):1311–1314, 2004. Cited in page 34.

Olshausen, B. A. and Field, D. J. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996. ISSN 0028-0836. Cited in page 41.

Oquab, M., Bottou, L., Laptev, I., and Sivic, J. Learning and transferring mid-level image representations using convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1717–1724, 2014. Cited 3 times in pages 35, 39, and 43.

Osadchy, M., Cun, Y. L., and Miller, M. L. Synergistic face detection and pose estimation with energy-based models. *The Journal of Machine Learning Research 8*, 8:1197–1215, 2007. Cited in page 39.

Pan, S. J. and Yang, Q. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010. ISSN 1041-4347. Cited in page 35.

Pellacani, G., Grana, C., and Seidenari, S. Algorithmic reproduction of asymmetry and border cut-off parameters according to the ABCD rule for dermoscopy. *Journal of the European Academy of Dermatology and Venereology*, 20(10):1214–1219, 2006. ISSN 09269959. Cited in page 40.

Quattoni, A. and Torralba, A. Recognizing indoor scenes. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, pages 413–420, 2009. ISBN 9781424439935. Cited in page 46.

Razavian, A. S., Azizpour, H., Sullivan, J., and Carlsson, S. CNN Features off-the-shelf : an Astounding Baseline for Recognition. *CVPR'2014*, 2014. Cited 5 times in pages 27, 35, 36, 39, and 43.

Rebecca, V. W., Sondak, V. K., and Smalley, K. S. A brief history of melanoma. *Melanoma Research*, 22(2):114–122, April 2012. ISSN 0960-8931. Cited in page 39.

Rosenblatt, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1959. Cited in page 31.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. Learning internal representations by error propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1, pages 318–362. 1986. ISBN 026268053X. Cited in page 32.

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. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015. Cited 2 times in pages 24 and 35.

Russel, S. and Norvig, P. *Artificial Intelligence: A Modern Approach*. Pearson Prentice Hall, 3 edition, 2010. Cited in page 23.

Seidenari, S., Pellacani, G., and Grana, C. Pigment distribution in melanocytic lesion images: A digital parameter to be employed for computer-aided diagnosis. *Skin Research and Technology*, 11(4):236–241, 2005. ISSN 0909752X. Cited 2 times in pages 41 and 51.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. OverFeat : Integrated Recognition , Localization and Detection using Convolutional Networks. *arXiv preprint 1312.6229*, pages 1–15, 2013. Cited 4 times in pages 35, 46, 47, and 48.

Situ, N., Yuan, X., Chen, J., and Zouridakis, G. Malignant melanoma detection by Bag-of-Features classification. In *International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 3110–3113, 2008. Cited 2 times in pages 41 and 51.

Sivic, J. and Zisserman, A. Video Google: A Text Retrieval Approach to Object Matching in Videos. In *Proceedings of the International Conference on Computer Vision*, volume 2, pages 1470–1477, 2003. Cited in page 23.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout : A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research (JMLR)*, 15:1929–1958, 2014. ISSN 15337928. Cited in page 33.

Tommasi, T., Orabona, F., and Caputo, B. Safety in numbers: Learning categories from few examples with multi model knowledge transfer. *Computer Vision and Pattern Recognition (CVPR)*, pages 3081–3088, 2010. Cited in page 35.

Tuytelaars, T. Dense interest points. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2281–2288, 2010. Cited in page 30.

Urteaga, O. and Pack, G. T. On the antiquity of melanoma. *Cancer*, 19(5):607–610, 1966. ISSN 0028-0836. Cited in page 39.

Vapnik, V. N. *Statistical learning theory*, volume 1. Wiley New York, 1998. Cited in page 30.

Vedaldi, A. and Lenc, K. MatConvNet - Convolutional Neural Networks for MATLAB. *arXiv preprint 1412.4564*, abs/1412.4, 2014. Cited 3 times in pages 35, 37, and 55.

Wadhawan, T., Situ, N., Lancaster, K., Yuan, X., and Zouridakis, G. SkinScan©: A portable library for melanoma detection on handheld devices. In *Proceedings - International Symposium on Biomedical Imaging*, pages 133–136, 2011. ISBN 9781424441280. Cited 2 times in pages 41 and 51.

Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., and Gong, Y. Locality-constrained linear coding for image classification. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3360–3367, 2010. Cited in page 39.

Werbos, P. J. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University, 1974. Cited in page 32.

Whitby, B. *Artificial Intelligence: A beginner's guide*. Oneworld, 2008. Cited in page 23.

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, pages 3320–3328, 2014. Cited 3 times in pages 36, 39, and 78.

Zeiler, M. D. and Fergus, R. Visualizing and Understanding Convolutional Networks. *Computer Vision – ECCV 2014*, pages 818–833, 2014. Cited 4 times in pages 35, 36, 39, and 43.

Zhang, N., Paluri, M., Ranzato, M., Darrell, T., and Bourdev, L. PANDA: Pose Aligned Networks for Deep Attribute Modeling. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1637–1644, June 2014. ISSN 10636919. Cited in page 38.

Zhou, X., Yu, K., Zhang, T., and Huang, T. Image classification using super-vector coding of local image descriptors. In *European Conference on Computer Vision (ECCV)*, pages 141–154, 2010. Cited in page 39.

Zouridakis, G., Wadhawan, T., Situ, N., Hu, R., Yuan, X., Lancaster, K., and Queen, C. M. Melanoma and Other Skin Lesion Detection Using Smart Handheld Devices. In *Mobile Health Technologies: Methods and Protocols*, pages 459–496. 2015. Cited in page 40.