Dublin City University
Insight Centre for Data Analytics

Universitat Politecnica de Catalunya
Escola Tècnica Superior d'Enginyeria de Telecomunicacions de Barcelona

Eric Arazo Sánchez

The Impact of Visual Saliency Prediction in Image Classification

Supervisors: Kevin McGuinness, Eva Mohenano and Xavier Giro-i-Nieto

Dublin 2017

Abstract

This thesis introduces an architecture to improve the accuracy of a Convolutional Neural Network trained for image classification using visual saliency predictions from the original images.

In this thesis the accuracy of a Convolutional Neural Network trained for classification has been improved using saliency maps from the original images. The network had an AlexNet architecture and was trained using 1.2 million images from the Imagenet dataset. Two methods had been explored in order to exploit the information from the visual saliency predictions. The first methodologies implemented applied the saliency maps directly to the existing layers of the CNN, which in some cases were already trained for classification and in other they were initialized with random weights. In the second methodology the information from the saliency maps were merged from a new branch, trained at the same time as the initial CNN.

In order to speed up the training of the networks the experiments were implemented using images reduced to 128×128. With this sizes the proposed model achieves 12.39% increase in Top-1 accuracy performance with respect the original CNN. Additionally reduces the number of parameters needed compared to AlexNet. Regarding the original size images (227×227) a model that increases 1.72% Top-1 accuracy is proposed.

Acknowledgments

First of all, I would like to thank to my advisors Xavier Giro-i-Nieto, Kevin McGuinness and Eva Mohenano. Not only for the help and guidance during the whole project, but also for sharing their passion for deep learning and initiating me in this amazing field.

I want to thank to my family for its support and wise advise during these months. As well as to my friends for their patience and kindness.

A special thank to Maria Arazo for being that sister who helps with whatever you might need.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recognition of objects and scenes for humans is an almost instantaneous, precise and extremely adaptable process, since we keep learning during all our life. However, we are seldom aware of the time and energy spent on creating those neurological structures that make such a complex process possible. It is well known how the human brain lowers this complexity by filtering out part of the information and only processing those things that capture our attention. In this work we propose a new model that achieves 12.4% increase in accuracy reducing the amount of parameters to tune when adding to the network information about the most salient parts of the images.

In the last years, the computer vision community has been approaching this complexity through deep learning, whose basic structures, called Convolution Neural Networks (CNN), are inspired on those found in the mammal brain. Their basic structure consists on the composition of layers with several nodes each. Those nodes, which are called neurons, compute a basic operation. The parameters involved in these operations are modified during the learning process of the network until it is finally able to perform the task it was designed for.

Not only a structural resemblance links those networks to the human brain; it also has a similarity regarding its functionality. The human visual system bases its performance according to how several features from images or scenes relate to previously learned features. Going from higher to lower levels of abstraction, those features aim to recognize borders or edges on one level and more complex structures such as eyes or wheels on the other. The same happens with neural networks, where the random initial parameters become defined shapes after the training.

One of the main traits of our brain is the ability to select which information coming from the outside world is important and which is not, particularly regarding our visual sense. We are able to recognize the main components of an image after a quick glance, and we are even capable of giving a rough description of that image based on the information provided by the few interest points that first caught our attention. However, as we keep exposing ourselves to

that image, we start to identify and take into account other details that eventually help to the overall recognition of the scene.

One of the factors that make those secondary details more or less important is how much they stand out from the image. This, mixed with our biological predisposition to respond to certain shapes or colors, allows us to recognize in a simple glance the most important sections from an image. This is reflected on where do we place our attention, and consequently on where do we fix our eyes when an image is showed to us.

The most reliable way to record this behavior is by tracking eye movements while several images are shown; other methods consist on asking to the subjects to click with the mouse on the areas of interest. Whichever the recording method is, the results can be used to build an image the same size as the original image, where each pixel represents the probability for the eyes of the subjects to fix their gaze on that pixel. This way the saliency map of the image is created: a map which indicates the sections where humans most probably look at.

Since this trait is crucial to enhance our performance at identifying or labeling images, providing to a CNN information about the most salient parts from an image seems to be a logical step in the improvement of the CNN at classification tasks.

1.1 Motivation

This master thesis aims to explore the practical applications of visual saliency prediction. Historically, the literature around this topic has been focused on improving the models to generate more accurate saliency maps.

The simplest working frame to test how valuable is that information is the image classification task. On one hand it can be easily tested if the performance of a model improves after introducing the information from the saliency maps. And on the other hand there are several datasets for image classification that can be used. From that point, the question that arises is how the saliency maps should be used. Several methodologies are proposed further in the thesis, which performance will be compared with the results obtained without the extra information of the salient sections of the images.

A priori it might seem reasonable to consider the approach as a data augmentation or a transfer learning strategy, since adding the saliency maps provides more information during training to the CNN. At first glance, it is similar to increasing the size of the dataset. However, some considerations should be taken into account.

- First, the information that each saliency map provides is not as precise and with as many features as the original images.

- Second, they are partially eliminating the information from the sections of the image that are not salient.
- Third, saliency maps provide clues of the more important and distinctive features of an image. That might help in the training of a CNN, and consequently achieve the same accuracy in less epochs.
- Opposed to the previous point, since the saliency maps provide more information for the network to train with, it might take longer to achieve the same accuracy.

Consequently, the way to use the saliency maps to improve image classification might not be trivial.

1.2 Objectives

The main goal of this work is to study the practical applications of visual saliency prediction. Specifically, how it affects the training of a convolutional network in the task of image classification.

Bearing in mind the ideas mentioned in the motivation (section 1.1), the specific goals of the thesis are synthesized as follows:

- Provide a broad idea on how the information provided by the saliency maps can be used to improve other tasks.
- Study how the introduction of saliency maps affects the training of a CNN in classification.
- Test which is the best way to add the saliency to a network already trained for classification.

1.3 Contributions

A modification of a classical CNN architecture for classification is proposed consisting on a two branch network that exploits saliency information. We will refer to this model as *Fan-in* network. The new model achieves a 12.39% increase in accuracy on the validation set with respect the original CNN network. Remarkably, the proposed model achieves this in spite of the fact that it has fewer parameters than the original.

A comprehensive study in how to apply saliency information at different layers of a CNN has been done leading to test several strategies along the project. Based on the conclusions

of [28] the problem has been approached first with downsampled images to speed up training and allow more configurations to be tested in a reasonable timescale, and then tested on larger images. The purpose of the methodology applied was accelerate the training process in order to have some results to guide future experiments. Trying to make the training even faster, several experiments were done using pre-trained weights and adding the saliency information at different layers of the network. Those experiments provided information of the behavior of the network depending on how the saliency information was added.

The strategy that provided best performance was adding the saliency information as extra channels into intermediate layers. Those experiments were reproduced by training from scratch instead of using already trained weights and training for longer times. Later, they were scaled and carried out with the higher resolution images. The two branch CNN architecture come from the discarded experiments which showed poor performance when the saliency maps were multiplied by the outputs of the layers. This strategy outperforms the approaches were the saliency is concatenated as an extra channel. Several modifications of the two input network are tested in the reduced images and one of them in the larger images.

This path followed, exploring several solutions, lead to an architecture that successfully introduces the visual saliency predictions to a classical CNN for image classification reducing the number of parameters and increasing its accuracy performance.

1.4 Software requirements

The main framework used during the development of the thesis has been Keras [5]: a high-level neural networks Python library. It provides an easy way to create and manage deep learning models, and its main purpose is enabling fast experimentation. The low level computations are carried out by another low-level Python library, as well as the management of tensors. This is known as the backend for Keras and it can be Theano [41] or TensorFlow [2]. The experiments done in the thesis regarding the classification task used Keras over Theano.

To approach the generation of the saliency maps the pre-trained SalNet [31] CNN has been used, which runs over other deep learning framework, Lasagne [8]. More information regarding SalNet is provided in the section 3.1.

Since the dataset used in the experiments is considerably large, a way to compact the images was required. For this purpose a compaction code [26] developed by Kevin McGuiness was used. It allowed to yield batches of images from the training and validation sets to the network, at the same time as performs a data augmentation consisting on cropping and flipping the images when required.

1.5 Work plan

The project was developed as a master thesis in a exchange program in the Insight Centre for Data Analytics at the Dublin City University. Weekly meetings were held to guide the evolution of the thesis. Thus, after showing the results obtained, several new proposals were discussed in order to design new experiments.

In the following section the work plan is split into several work packages that describe the process followed during the developement of the thesis.

1.5.1 Work packages

- WP1: Introduction to Deep Learning and Python.
  - Learning Python.
  - Deep Learning and CNN.
  - Learning Keras.
- WP2: Project proposal and work plan.
  - Project proposal.
  - State-of-the-art-research.
  - Project approval.
- WP3: Preparing the image classification task.
  - Introduction to image classification.
  - Training exercises with small datasets.
  - Introduction to ImageNet.
  - Learning to deal with large datasets.
- WP4: Generating the saliency maps from the dataset.
  - Choosing model for saliency predictions.
  - Generating the saliency maps from ImageNet.
- WP5: Training CNNs.
  - For image classification (raw images).
  - With saliency maps in the input.
  - Reducing image sizes.
  - Training a CNN with small images.

- WP6: Experiments to improve classification.
  - Pre-trained weights (30 epochs).
  - Random weights (90 epochs).
  - Using large images.
  - With two input branches structures.
- WP7: Report of the thesis.
  - First draft.
  - Second draft.
  - Report delivery.
- WP8: Oral presentation and defense of the thesis.
  - Prepare presentation.
  - Rehearsal of the presentation.
  - Oral defense.

1.6 Document structure

During the development of this thesis several experiments have been done, each implementing different strategies. Consequently, to provide a complete view of the methodologies implemented and all the possibilities still to be explored, the thesis has been written with enough detail to give a sufficient background to the reader.

Starting with the introduction, in the Chapter 1 the motivation of the thesis is described, along with some hypothesis of the usability of the visual saliency predictions in other applications. Then, the objectives are presented and the contributions made throughout the project are summarized. Additionally, a brief overview of the software used is given and finally a description is provided of how the project was developed in terms of organization.

Chapter 2 provides a basic background of deep learning and convolutional neural networks, as well as an overview of the most used dataset. Then, the first steps towards the problem concerning the thesis are done: a selection of literature of the state of the art dealing with practical applications of saliency is discussed. In the final section the strategies followed throughout the thesis are introduced.

The core of the practical part of this thesis is comprised between the third and fourth chapters. First in chapter 3, the general methodology followed is explained in order to dig later into the details of each experiment done. In the chapter 4 the results obtained in the experiments

are presented and analyzed To finalize the report, the chapter 5 provides an overview of the contributions done in this thesis, as well as some conclusions which lead to other questions and hypothesis, briefly explained in the last chapter 6.

Chapter 2

State-of-the-art

These last two decades artificial intelligence (AI) has been a very explored topic of discussion, not only in academic environment but also among industry. The last technological advances that made it so popular are also making easy to forget that it already has been around for decades, going through periods of time when it was highly appreciated, as well as periods where enthusiasm decayed dramatically [43]. Nowadays, the advances achieved in parallel computing and GPU made AI an area of study much more accessible, specifically regarding machine learning, which has outperformed other strategies when addressing tasks like image classification or speech recognition.

Focusing in the topic concerning this thesis, the computer vision community has been considerably influenced by the application of a branch of machine learning called *deep learning*. It has shown to produce state-of-the-art results on various tasks. The main idea behind deep learning is to compose multiple layers of features so as to learn increasingly abstracted representations by backpropagation. In other words, learning certain features with high levels of abstraction which are defined, each of them, in terms of their relation with simpler features. This hierarchy makes learning complicated concepts out from simpler ones possible, creating thus long concatenations of features that give this approach its name.

At the same time, the use of smart phones and devices capable of recording data has increased exponentially since the beginning of the century. Consequently, the generated multimedia content has increased in a way that processing that information requires new techniques of representation. Here is where deep learning provides a very efficient solution, since its nature allows us to extract high-level abstractions as representations in order to represent high quantities of data.

Concerning computer vision some of the applications of this technology approach specific tasks such as face recognition [40], video scene classification and object tracking [19, 30, 42] or even in the field of health care with medical imaging applications [17]. However, the purpose of

this thesis leads us to point our research towards image classification [21] and saliency prediction [31, 35, 22].

The purpose of this chapter is to provide an overview of the practical applications regarding saliency prediction as well as a theoretical background of the topics involved in the development of the thesis. Before studying how visual saliency have already enhanced the performance of other applications, the following sections will cover the basics of the technology involved.

2.1 Deep learning

This section covers the basic knowledge needed to understand the principles behind the basic strategy employed in this thesis. As was mentioned before, deep learning is a class of machine learning algorithms that aims to provide a multidimensional space to represent data. Hierarchical structures called Convolutional Neural Networks (CNN) are used for that purpose.

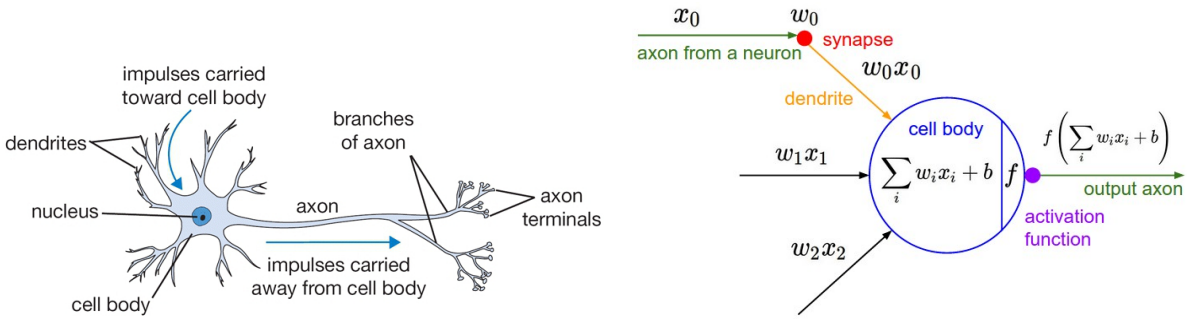
Specifically in computer vision, CNNs have shown impressive results. To have a picture of the improvements, it is interesting to see the evolution of some specific tasks such as one of the first approaches towards image recognition, which dealt with character recognition [23]. From approximately an error rate of 10% in the 1998 it improved to a point where it might have achieved human accuracy with an error rate inferior to 0.30% [6]. Other task being addressed is image classification that through the ImageNet competition [36] has been providing more accurate models and a better understanding of the convolutional networks themselves. A particular discovery showed that those models trained for that competition could be used in other situations and were performing extremely well [9, 37], even outperforming those models specifically trained for that task. The main reason for that is the variety of images showed in the datasets provided for the ImageNet competition, which trained the models to see the world in a way very close to the human visual system. It is not that surprising then, that after some decades of trying to develop better architectures (to improve performance on visual tasks) they converged to networks that potentially function in a similar way to the visual cortex of some primates as was found in [4].

Several elements have to be considered in order to build a comprehensive view of the topic under discussion, going from the CNNs themselves and the elements they are made of to the data used to train them and strategies to deal with it. They are explained in the following subsections.

2.1.1 Convolutional Neural Networks

The basic structure conforming this hierarchical structures is known as a neuron. This name comes from its similarity to the biological neurons from the brain, since it tries to replicate

its functionality. Figure 2.1 shows the structural and functional resemblances. Both of them receive several inputs which are weighted and then computed in order to produce an output. Each of the input is multiplied by some parameters, called weights, and introduced to the body of the neuron. Commonly, after the computation is done a non-linearity is applied, that settles a threshold for when the neuron provides an output or not. This is why the non-linearity is also called activation function.

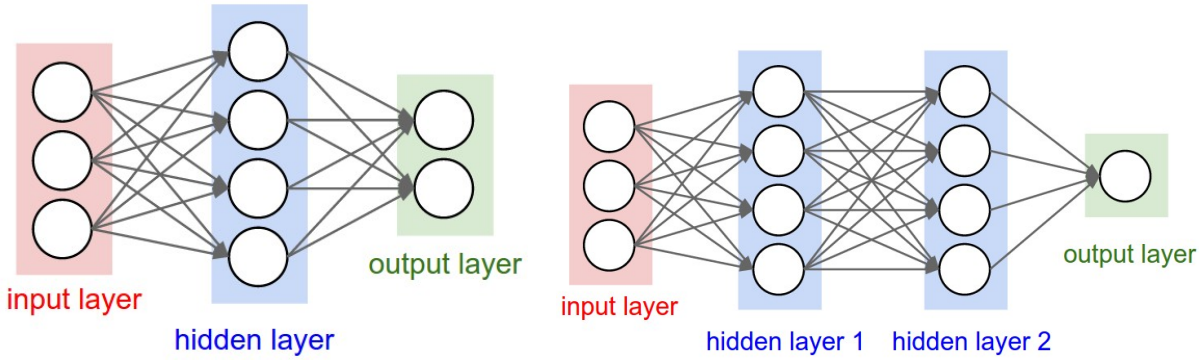


(a) (b)

Figure 2.1: A cartoon drawing of a biological neuron (a) and its mathematical model (b) [1].

When several of this neurons are clustered together they constitute a layer, that once is provided with the input data, computes its activations and generates an output that most probably will be the input for another layer. When several layers are concatenated they form a neural network and those layers that are situated between other layers are called hidden units or hidden layers. In figure 2.2 can be seen a two layers neural network on the left and a three layers NN on the right.

In the following sections different layers are explained in order to build a better understanding of the structure of a neural network.



(a) (b)

Figure 2.2: A two layers Neural Network (a) and a three layers neural network can be seen in the image [1].

Fully Connected Layers

In the layers explained before, each of the neurons receive all the values from the previous layers or from the input data. This can be appreciated in figure 2.2. This structure requires a high number of connections, what means a large number of parameters and consequently a large amount of memory is required. These layers receive the name of fully connected layers and are mainly used to generate the features that deal with the class probabilities.

Convolutional Layers

These layers are the ones that give the name to the convolutional network structures. Its main characteristic is that they group together several values of the input, forming matrices. The weights applied to the input values receive the name of filters and they are different for each neuron. This strategy provides to the network spatial coherence and consequently make this layer really efficient in the computer vision field. Moreover, avoiding the connection of all the values of an input to each of the neurons requires far less parameters. The inputs of a neuron are the values of a two dimensional matrix in the image. This will compute an output and move to another region of the image, covering this way all the image and generating an output. This output is called feature map and each pixel represents how does the weights of that neuron (a filter) react to a particular region of the image.

Max-pooling Layers

The purpose of using pooling layers along the network is to gradually reduce the spatial size of the representations of the images reducing at the same time computation for the following layers. To do that a window of pixels from the representations is selected and an operation is applied. In the case of the max-pooling layers, only the highest value from the pixels of the selected window is passed to the output of the layer. Then this operation is done along all the representation.

Batch Normalization Layers

As it will be explained in future sections, during training the parameters of the layers change until they achieve an optimal representation of the images. Consequently, the distribution of the layers' inputs also change. Thus, the layers have to constantly adapt to the new distributions. In [15] they introduce the term *intern covariance shift* to refer to the phenomena of the changing distributions, that forces to use lower learning rates, what slows down the training.

The main objective of the batch normalization layers is to reduce the internal covariance

shift and this way improve the training. This is done by whitening the input of each layer; i.e. forcing the mean of the images to be close to zero and its variance close to one.

In the aforementioned study [15] they accelerate the training of CNN adding batch normalization layers, since it allows several changes in the network such as: increasing the learning rate and accelerating its decay, remove dropout, reduce the L_2 weight regularization, and remove local response normalization among others.

Dropout Layer

To avoid overfitting the network when using small datasets, in [13] they introduce the technique called *drop out*, that gives name to the layer. It randomly omits some of the activations of the previous layer, this prevents from having features that depend one to each other.

2.1.2 The architecture

Regarding the task of image classification most of the advances come motivated by the Imagenet challenge [36] that brought improvements on the task involved and in other. Specifically in 2012 there was a highly remarkable improvement. The network proposed by [21] not only outperformed all its competitors but also made a qualitative jump being the first CNN used to achieving a top 5 test error rate of 15.4% (the next best entry was 26.2%). This network, called AlexNet, settled the basis for new structures and is the one selected to develop the practical part of this thesis.

This section will deal with the technical details of the network and give an overview of the networks that followed AlexNet in the ILSVRC challenge.

Alexnet

This section is also meant to give a brief insight of the role of the different layers when they are used together to build a CNN. This will be done from the perspective of the AlexNet network, that is an 8 layers CNN. The basic structure consists of five convolutional layers and three fully connected layers. As it can be seen in the figure 2.3, the original structure used two strategies that in this thesis are replaced by batch normalization; they implemented a local response normalization and a parallelization in some of the convolutional layers.

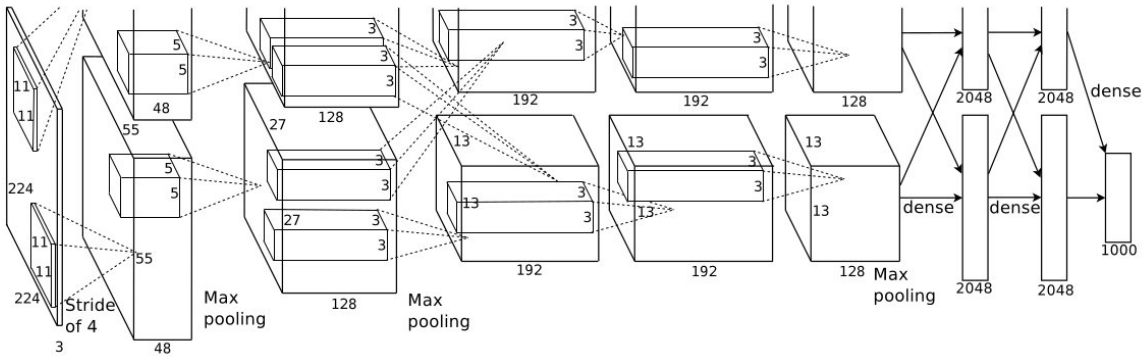


Figure 2.3: Illustration of the architecture of the AlexNet CNN introduced in [21]

The purpose of the convolutional layers is to extract features that represents the input images. The earlier layers provide features for low level representations such as edges or colors, whilst those from deeper layers provide higher level representations such as parts of objects or faces.

In the implementation used for the practical part of this thesis the first and second convolutional layers are followed by a batch normalization layer and, after the first, second and fifth layers a maxpooling operation down-sample the activations of the previous layers. Then, three fully connected layers can be found, they adapt the activations from the convolutional layers to a feature vector that will be associated to the classes of the objects. Each of the layers commented is followed by a Rectified Linear Unit (ReLU).

Other networks

Since AlexNet was introduced in 2012, the CNNs have been deeply explored and year after year the performance of the models for image classification have been improving until achieve and even surpass human-level performance on several tasks. In the following there is a brief summary of the innovations that followed this first introduction of the CNN in the image classification task.

Right after AlexNet, in 2013 [46] presented the winner of the challenge for that year, the ZFNet. Zeiler and Fergus showed that reducing the filters from in first convolutional layer from 11×11 to 7×7 and using a stride 2 instead of 4 significantly improves performance. This is using smaller filters but applied more densely. They found that increasing the sizes of the convolutional layers from the middle of the network also improves accuracy. With these modifications they managed to achieve a top-5 error rate of 14.8%.

Next year a very homogeneous and uniform structure was introduce, VGGNet [38]. It was not the winner but worth mentioning due to its simplicity. It consist on 3×3 convolutions

with stride 1×1 and 2×2 maxpoolings with stride 2, followed by three fully connected layers. The winner of 2014 ImageNet Challenge was GoogleNet [39], where the size of parameter was drastically reduced to 5 million parameters (against the 138 million of VGGNet). They showed an increase in the performance by removing two of the fully connected layers. With this procedure they used even less parameters than AlexNet with better performance (top 5 error rate of 6.67%).

In 2015 a new architecture was introduced in [12], the Residual Neural Networks (ResNet). The model they not only won the ImageNet competition, but also was pre-trained on ImageNet to outperform other networks in several other tasks. One of the most remarkable characteristics is that, unlike in the plain nets seen until now, more layers entails more performance. Consequently, they trained a 152 layers network and achieve a top 5 error rate of 3.57% (beating human-level performance on this dataset).

2.1.3 Training

In order to make a CNN efficient for an specific task it has to go through a process of training where the weights from the layers change until the output of the network is close enough to the target value. This process consists of showing the network data from a training dataset and computing a loss function that measures the quality of the parameters based on how well the output agrees with the ground truth. Then the weights are updated to minimize this loss function. Once this is done, more data is given to the network to repeat the process until the parameters converge. The loss function can be seen as a multidimensional surface in which the parameters are moved until they converge to a critical point.

The method used to update the weights is called backpropagation, an algorithm that propagates the error of the prediction at the output of the network backwards towards the input. The gradient descent algorithm is used to optimize the loss function, which is parametrized with the parameters of the network. The gradient of that function is computed in the output of each layer and used to update the weights (see equation 2.1). The size of the step made in each update is determined by the learning rate.

$$\omega_i = \omega_{i-1} - \lambda \nabla_{\omega_{i-1}} L(\omega_{i-1}), \tag{2.1}$$

where $L(\omega_{i-1})$ is the loss function defined accordingly to the necessities of each case, ω_i are the weights for the current iteration, ω_{i-1} the weights from the previous iteration and λ the learning rate.

2.2 Datasets

The strength of deep learning is highly related to how does a network adapt to a certain dataset. However, the quality of the datasets used during the training plays an important role in the end performance of the network. This is strongly linked to the process of collection and labeling the data, one of the biggest milestones concerning this area of study.

The lack of labeled data has been the motivation of several studies, whether it was to generate new datasets or to provide more information from an already existing dataset. One of the drawbacks commonly faced when generating certain datasets is the precision required to properly label the images, since in some cases it requires experts on that particular topic to recognize and differentiate between different classes. Thus, the presence of unlabeled data is more common than what is desired. An approach regarding this situation is to exploit the natural ability of humans to distinguish objects without previous training on the field [18], using human gaze data as auxiliary information for zero-shot image classification.

There are several datasets that have been shown to be very useful in image classification tasks. Some of the most cited in the literature are CIFAR10 and CIFAR100 [20]. They were released in 2009 and provide low resolution images of 10 or 100 classes respectively. Another dataset among the most used ones is ImageNet, coming from the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [21] with 1.2 million images. A list of the most used datasets for image classification and other applications can be found in [44].

Since saliency prediction is a big part of this thesis, it is of relevant interest to mention some of the datasets currently being used in this task. Released by the MIT saliency benchmark [3] the MIT300 and CAT2000 provide images and fixations for both training and testing. Another well known dataset for saliency is provided by SALICON [16], which offers a large set of saliency annotations on the popular Microsoft Common Objects in Context (COCO) [25].

2.2.1 Data augmentation

Deep learning has shown that in most of the cases, increasing the sizes of the datasets results in a consistent improvement in the performance. Consequently, using certain techniques to enlarge a dataset has become a common practice. This procedure is called *data augmentation* and aims to artificially increase the size of the dataset achieving at the same time several improvements such as invariance to certain transformations or reducing overfitting. The most common of them consists on flipping the images and using random crops of the original image for training the network.

2.2.2 Unsupervised learning

Another approach to avoid the problem of poorly annotated datasets is using unsupervised learning techniques where the network models a function to describe unlabeled data. This is a useful technique to learn about the structure of the data. Although this is a topic that lies out of the scope of this thesis, it is worth mentioning [24, 33] since a huge quantity of unlabeled data is being recorded constantly by smart phones, wearables and other devices.

2.3 Applications of saliency

A thorough revision of the literature has been done to understand the possibilities offered by visual saliency predictions. As it has been stated in the motivation, in section 1.1, they certainly provide information about the image and in a broader picture, bring to the network information learned in a different dataset; what a priori might look like a data augmentation or a transfer learning strategy. However, the question concerning this thesis is whether this information can be used to improve the performance of a CNN in the task of image classification.

In the case of image retrieval for instance, some examples show how the saliency maps have been shown to be useful are [10, 34, 7, 14]. In [10] the first example the saliency maps are computed with the model proposed in [11]. They provide a method to choose the image features that describe the content of the images, and achieve greater performance than with the standard representation. Moving towards egocentric vision, the work from [7] introduced a method to build saliency maps to be used for active object recognition. On the other hand, in [34] they use visual saliency information to enhance the performance of a retrieval system to find personal objects in egocentric images. They show that when the target images are weighted with the saliency maps the results improve significantly. In this case the saliency maps are obtained using the pre-trained SalNet [31] CNN, and the method used to add them to the network follows [29].

More evidence of the contribution in terms of information provided by the saliency maps can be found in [18] where, as mentioned in section 2.2, two gaze-annotated datasets are introduced. This study proves that human gaze data is class discriminative, provides an alternative to expert annotated-attributes and outperforms other baselines for zero-shot image classification.

In the field of health-care we find the study [7]. As mentioned before, dealing with an egocentric perspective is another example of how saliency maps provide a powerful tool for computer vision applications, this time in real time object recognition and localization they propose an interactive user-friendly environment for upper-limbs amputees. In the mentioned work vision helps to control the prosthesis in those cases where the amount of remaining muscles is insufficient to perform EMG control efficiently. Here the saliency maps are used to select the bounding boxes and generate windows of interest that will be submitted to the CNN and then

recognized.

A recent study [45] shows how attention maps can improve significantly the accuracy of a network in classification tasks. Even though it is not using visual saliency predictions nor gaze fixations, a successful method is proposed to transfer knowledge from one network to another. In the case concerning this thesis it can be applied using a network for classification and a network that generates visual saliency predictions. This will be discussed in the chapter 6 as a possible future work. The methodology proposed in the mentioned study consists on computing the attention maps from one network (which is the teacher network) and use them to guide the training of a student network. It has been shown that this methodology provides significant improvements even when the student network is smaller than the teacher network.

2.4 Improving image classification

In this section the previous theoretical background and the studies mentioned are merged to bring us to the practical part of the thesis. Different methods to use saliency maps are proposed in order to improve the performance of a CNN in the classification of images.

All the strategies used by the studies mentioned above try to use the visual saliency information to emphasize the sections of the images that might contain more information, or at least, those that seem more important for humans. Some of these strategies use the information to select the sections of the image that will be submitted to the CNN. This approach leads us to apply the saliency maps to the input images of the network, which leads to the first proposal. As it did with image retrieval tasks, adding this extra information to the input images should make the training of the CNN faster and provide a higher final accuracy.

Other authors decided to introduce the saliency maps in deeper layers of the network. In [34] for instance, they are introduced into the fifth convolutional layer of a VGG16 network [38]. In that case the saliency maps are resized to match the dimensions of the output from the layer, which is 56×56 pixels; a size that a priory seems big enough to provide significant spatial information. A similar case can be found in [45]. In that study they do not deal with visual saliency but with attention maps. However, the basic principle consisting on adding the information in the middle of the network remains the same. Due to those approaches, some of the main experiments during the development of the thesis introduce saliency information along the CNN. The network chosen for the experiments slightly limited the options when adding the saliency maps, since for the deeper layers those had to be reduced to very small sizes. In Chapter 3 more technical information is provided.

Whether the saliency maps are introduced at the beginning of the network or in later layers, the way to add the maps still needs to be carefully studied. In the following chapter two strategies are tested regarding that matter. However, in order to have an approach less

dependent on the criteria followed, a third proposal was made, which gives more freedom to the network for processing the information coming from the saliency maps. This strategy consist on, with two input branches, introduce the saliency maps in an intermediate layer.

Chapter 3

Methodology

The purpose of this chapter is to describe the methods and procedures involved in the development of this thesis. The first section describes the approaches for the tasks involved: image classification and saliency prediction. The following sections deal with the preparation of the data, the technical details concerning the training of the CNN and the description of the different experiments.

The process that leads us to the final results can be divided into three stages, each of them comprising several experiments. In the first stage (explained in section 3.4), we study the behavior of the network using already trained weights; the objective of this step is to obtain results within short times to guide future experiments. The second stage (explained in section 3.5) starts from the results obtained in the previous one, and it aims to validate different strategies when the network is instead trained without previous knowledge. Finally, the last stage consists of a slightly different strategy that provides a significant increase in the accuracy, as is explained in section 3.6.

3.1 Computer vision tasks involved

3.1.1 Image classification

The fact that many studies used saliency maps with tasks such as image retrieval, but none tried to use them to improve image classification, made it a very challenging task to start with. Moreover, this task provides a framework where the saliency maps have applicable meaning, since the salient parts of an image play an important role in how humans classify objects. To approach image classification we selected the CNN Alexnet (more technical details will be explained in 3.3.2) and the ImageNet dataset.

ImageNet

We selected this dataset to train and validate all the models that have been proposed. The main reason for this choice is the success it has showed to provide in terms of quality of the features learned. Actually, it has become a common practice training the CNN on ImageNet to then adapt the features for a new task. Moreover, improving the results in a dataset as challenging as ImageNet with saliency predictions computed with a pre-trained CNN, proves that saliency maps provide valuable information of the images, and that this information can be used in the training of another CNN to improve performance.

The ImageNet project provides an image database for non-commercial purposes, organized accordingly to the WordNet hierarchy [27]. For this thesis we selected the subset published for the ImageNet Large Scale Visual Recognition Challenge from the 2012 (ILSVRC1012), containing 1.2 million images manually annotated with one single label per image and distributed within $1,000$ categories. Specifically, the dataset used for training contained $1,281,167$ images and the dataset for validation $50,000$.

Having such a large number of images makes it complicated to manage the dataset. Furthermore, the images are organized by folders, one per each class, and therefore the access to the images is not a trivial task. In section 3.2 the procedures followed to use ImageNet in training and for validation are explained, as well as the details about how the images were processed in order to be provided to the network.

3.1.2 Saliency prediction

We used the pre-trained SalNet [31] CNN to generate the saliency maps for the ImageNet datasetwe needed a model trained for that purpose, so we chose the pre-trained CNN SalNet [31]. This model provides an accurate and fast implementation for computing the saliency predictions. It was trained on the training and validation data provided by SALICON and developed using the deep learning framework Lasagne. The model is described in [32], as well as more detailed information about their work.

3.2 Processing the data

3.2.1 Preparing the dataset

For any computational task that deals with large datasets, as in the case being, it is convenient and almost indispensable to prepare the data before actually using it, and thus optimize the computing resources. The common approach in this situations is compacting the data to ac-

celerate the reading process. At the same time some transformations have to be applied to the images (and to the saliency maps). These processes are discussed in this section.

Since in the ImageNet dataset the images are organized by classes, first they have to be shuffled. Otherwise all the images with the same label will be given to the CNN one after the other. Then, at the same time as the compaction takes place, the images are rescaled and cropped. The procedure is as follows.

First, the image is scaled, the small side of it is reduced to 256, and the large side to the correspondent size to maintain the original aspect ratio. Then, the center of the image is cropped so that the size of the final image is 256×256. Once this is done, the images are compacted consecutively in a binary file along with their corresponding labels and identification number. This way the size of the dataset has been reduced, and the images can be accessed sequentially, which reduces I/O llatency.

The next step consists of computing the saliency maps of the images. For that purpose each image is extracted from the compacted file and the saliency map is computed with SalNet, previously mentioned. Even though the images and saliency maps could be provided right away to the network, we decided to compact them together in another binary file, as well as the labels, computing the saliency maps only once for all the experiments.

The final file that we used into our model is a 26.9GB file for training with $1,281,167$ images and 1.1GB for validation with $50,000$ images (and the corresponding number of saliency maps).

3.2.2 Processing during training

The processing described in this subsection is called "*on the fly*" since it is done during training. The previously described process leads to a dataset with the images almost ready to be provided to the network. However, before that, some more processing has to been done depending on the case. For all the experiments, the values of the pixels were normalized (only values from 0 to 1), but only in some of them the mean of the images was subtracted (specified in the section dedicated to each of the experiments).

Additionally, to perform the same data augmentation implemented in [21], some images (randomly chosen) are horizontally flipped and cropped by boxes of 227×227 randomly positioned along the images. This also has to be done with the maps: the same crops have to be done and the same maps have to be flipped.

The average time required to run the experiments in the conditions mentioned before is approximately nine days (more technical details concerning the training can be found in each of the sections dedicated to each of the experiments). To accelerate the process and thus be able to carry out more experiments in the time available, we decided to reduce the sizes of the images

and saliency maps. We based this approach in the study made in [28], where they show that the improvements achieved on downsampled images transfer to similar gains at higher resolutions. The scale we chose to reduce our data is the same that they proposed. The dataset used for the majority of the experiments consisted in 144×144 images and saliency maps that after being cropped provided to the network inputs of 128×128 pixels. Using this downsampled dataset the network could be trained within five days. Finally, once we established how to include the saliency maps to improve accuracy, the experiment was redone with the original images 4.6.

3.3 Setup for the experiments

This section describes the configuration and technical aspects involved in the training of the CNN, independently of the specifics settings of an experiment.

3.3.1 The baseline

We used the results obtained when training the network without saliency maps as a baseline to evaluate the different proposals. For this we followed as much as possible what was done in the original paper [21]. The training details for the baseline as well as for the other experiments are provided in the section 3.3.3.

The network was trained with 227×227 and 128×128 images. In both cases the images were processed following the procedure described in the previous section. Since the CNN uses batch normalization the mean of the images was not subtracted for these baselines. In the section 4.5 some experiments show the effect of this operation in the training of the network.

Table 3.1: Specifications for the baselines

Parameters	Description
Image sizes	128×128 and 227×227
Initial weights	Random weights
Mean subtraction	No
Number of epochs trained	90
Methodology to add saliency	No saliency involved

3.3.2 Architecture of the CNN

The convolutional neural network used in the experimental part of this thesis ins based on an AlexNet architecture. Aiming to an easier manipulation of the network, and at expenses of around 2% of the end accuracy, we decided to simplify the original architecture. Hence, we avoided parallelizing and then assembling the convolutional layers allocating all the nodes in one single layer. Table 3.2 shows the structure of the network, as well as the size of the filters and output of each layer.

The main reason why we use AlexNet instead of any other CNN is its relatively small training times and low memory requirements — 61 million parameters compared with the 138 million from VGG16 — made it the best choice for our experiments. Moreover it is a fast and accurate model which simplicity makes the manipulation of layers and its weights a simple task.

Table 3.2: Structure of AlexNet and the output sizes with 128×128 images.

	Output sizes of each layer
Input images (3, 128, 128)	
Convoltion 1 (96, 11, 11)	$96 \times 30 \times 30$
Batch Normalization	$96 \times 30 \times 30$
MaxPooling	$96 \times 14 \times 14$
Convolution 2 (256, 5, 5)	$256 \times 10 \times 10$
Batch Normalization	$256 \times 10 \times 10$
MaxPooling	$256 \times 4 \times 4$
Convolution 3 (384, 3, 3)	$384 \times 4 \times 4$
Convolution 4 (384, 3, 3)	$384 \times 4 \times 4$
Convolution 5 (256, 3, 3)	$256 \times 4 \times 4$
MaxPooling	$384 \times 1 \times 1$
Fully connected 1 (4096)	4096
Fully connected 2 (4096)	4096
Fully connected 3 (1000)	1000
Output	

3.3.3 Training

To have an accurate description of the practical part of the thesis, in the following lines the parameters shared among all the experiments are described Then, we provide the particular parameters used in the training of each experiment in the corresponding section.

During training and validation we feed the network with batches of 128 images processed as described in the section 3.2. The data augmentation is only implemented in the training set, using for validation centered cropped images of 128×128 or 227×227 pixels respectively. The initial value of the learning rate (λ) used in the optimizer is $\lambda = 0.01$ and is reduced by a factor of ten three times before finishing the training. The epochs where λ is reduced depends on the experiment. The optimizer used is stochastic gradient descent (SGD) with a momentum of $\mu = 0.9$ and the loss function to minimize is categorical crossentropy. Moreover, a L^2 normalization is applied to the weights of the network as regularization method with a penalty of $l2 = 0.0005$.

After a general description of the setup used in the experiments, the following sections provide a more detailed overview of each of them. First in 3.4, the experiments done with pre-trained weights provide a first approximation to which is the best way to add the saliency maps directly to the network. Once the results show some inclination towards one method or another those are trained with random weights for longer periods of time 3.5. Finally, some approaches where the saliency maps are pre-processed by the network itself are proposed and tested (see section 3.6).

3.4 Saliency with pre-trained weights

A first question that arises when designing the strategies to add the saliency maps is whether they should be multiplied by the weights or added as an extra channel. A second question is at which depth should the saliency map be inserted. A first answer might be that they have to be added in early layers of the network in order to maintain full resolution and provide more information. Several experiments have been done to see how the network reacts to different ways of introducing the saliency maps, and later implement those strategies that show best performance.

To reduce the computational time and be able to test more strategies, the network was trained for 30 epochs with pre-trained weights that were obtained from the network trained as a baseline. With this approach each experiment takes an average of 35 hours, which is better suited for testing strategies than the five days required for the 90 epochs trainings. To carry out the experiments, the model from the network trained as a baseline 3.3.1 is loaded and then the saliency maps are added. For those strategies where the introduction of the saliency maps implies multiplying them by the feature maps, the only process required is the down-sampling of the maps, so that they match the sizes of the output of the layer. In the case where the saliency maps are concatenated as an extra channel, apart from resizing, random weights have to be introduced alongside the pre-trained weights of the following layers, making them match the new dimension of the output of the layers where the saliency maps are introduced. Those random weights are drawn from a uniform distribution from 0 to 1.

To approach the questions addressed in the first paragraph, six experiments have been executed where the saliency maps are introduced in three different parts of the network (see figure 3.1). There are three experiments where the maps are concatenated and three where they are multiplied with the feature maps. The experiments are listed below, along with a brief explanation.

1. Maps multiplied by the input images (figure 3.1a): each saliency map is multiplied by each of the three channels of the input images.

2. Maps multiplied by the output of the first and second convolutional layers (figure 3.1b): this two layers have been selected due to their output sizes; a priori they are large enough for the saliency maps to contain spatial information. In this case the maps have been downsampled to 30×30 and 10×10 to make them match with the outputs of the corresponding layers.

3. Maps multiplied by the input images and by the output of the first and second convolutional layers (figure 3.1b): as in the previous case, the saliency maps corresponding to the convolutional layers have to be downsampled.

4. Maps concatenated as an extra channel in the input images (figure 3.1a: the network has the same structure but now the input images have four channels, one for each color and one for saliency. In this, case random weights were added to the first convolutional layer.

5. Maps concatenated to the output of the first and second convolutional layer (figure 3.1b): as with the multiplication strategy, the maps are downsampled to 30×30 and 10×10. However, in this case, random weights are added because there are no pretrained weights for these new channels. Those weights are concatenated to the pre-trained weights. They are concatenated to the layers after the introduction of the maps: in the second and third convolutoinal layers, as well as in the first and second batch normalization layers.

6. Maps concatenated as a fourth input channel and to the output of the first and second convolutional layer (figure 3.1b): the procedure in this case is very similar to the previous one. The maps are resized to the sizes of the outputs of the corresponding layers and random weights are concatenated to the first, second and third convolutional layer, as well as to the first and second batch normalization layers.

In all the previous cases the CNN was trained for 30 epochs with the setup described in the section 3.3.3. The initial learning rate was 0.01 and it was reduced three times throughout the training.

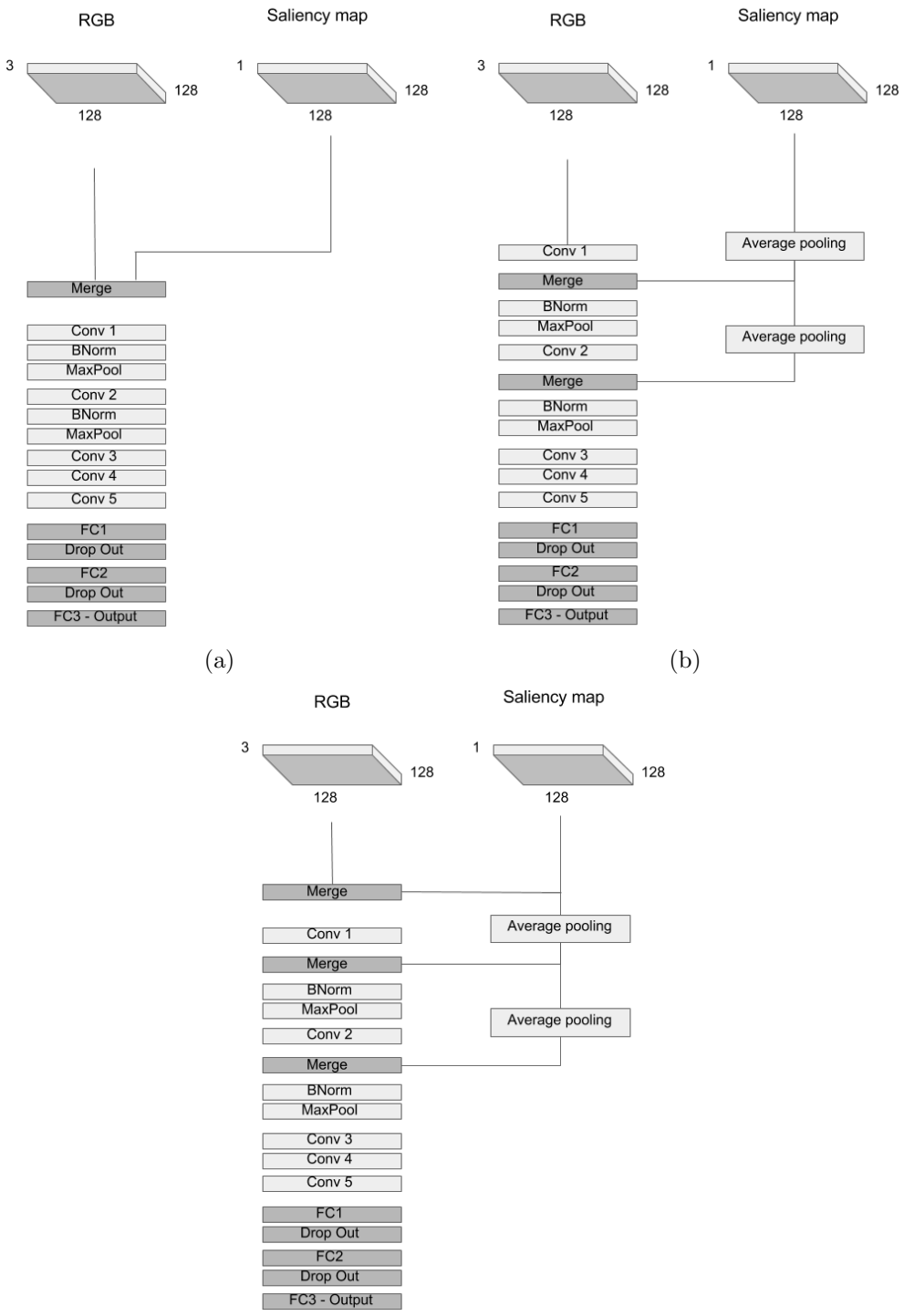


Figure 3.1: The saliency maps are added into the input (a), after the first and second convolutional layers (b), and in both at the same time, into the input and to the first and second convolutional layers (c).

Table 3.3: Specifications for the experiments with the pre-trained weights

Parameters	Description
Image sizes	128×128
Initial weights	Pre-trained 90 epochs
Mean subtraction	No
Number of epochs trained	30
Methodology to add saliency	Concatenated or multiplied depending on the experiment

3.5 Saliency concatenated to existing layers

This section describes the experiments that aim to study the accuracy of the network after the whole process of training, as well as the learning of the CNN during this process. For that purpose, in all the experiments the network was initialized with random weights and trained during 90 epochs. The initial learning rate is set to 0.01 and dropped by a factor 10 in the epochs 24, 45 and 62. The strategies followed to add the saliency maps are based in the results from the previous section 3.4:

- Saliency maps concatenated to the input images as a fourth extra channel.
- Saliency maps concatenated to the output of the first and second convolutional layers as an extra feature map.

The setup of the experiments listed above is described in the section 3.3.3. Since training takes approximately five days, only two experiments were done: those that showed the best accuracy while training for 30 epochs over pre-trained weights.

Table 3.4: Specifications for the experiments done with random weight weights

Parameters	Description
Image sizes	128×128
Initial weights	Random
Mean subtraction	Yes
Number of epochs trained	90
Methodology to add saliency	Concatenated to the feature maps as extra channels

3.6 Fan-in network

In the experiment described in this section a different approach was considered. In this case, instead of choosing how to add the saliency maps to the networks (multiplication or concatenation), we decided to give it more freedom to process them before being merged into the original structure of the CNN. These approaches consist on introducing the saliency maps after those have passed through some layers, and training the network to classify the images together with the saliency maps for 90 epochs. When the two input branches are merged each of the feature maps is multiplied one by one. It might be understood as if the saliency branch provide to each feature map information on the salient regions of this specific feature.

3.6.1 Merging before conv-3

The figure 3.2 depicts the structure of the network for this first approach. It consists on two input branches of two convolutional layers, two batch normalization layers and two maxpooling layers that are merged into the last part of the AlexNet structure: three more convolutional layers and two fully connected layers. With this approach, the network is learning which are the features that best describe both the saliency maps and the images in order to be merged for the classification task.

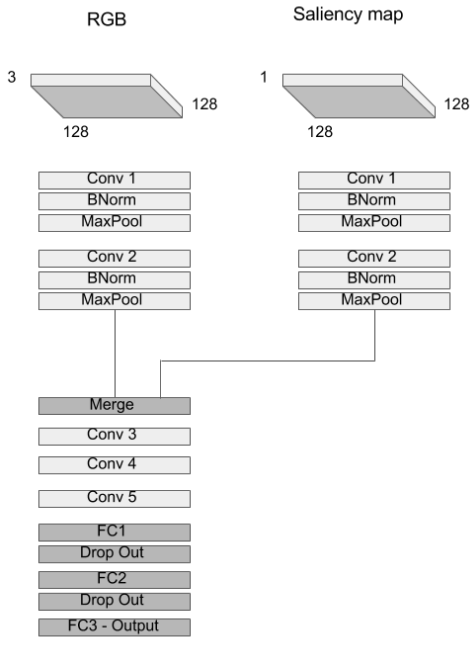


Figure 3.2: Two input architecture where both the saliency maps and the images, pass through two convolutional layers, two batch normalization layers and two maxpooling layers before they merge into a third convolutional layer.

Merging those two branches after the second convolutional layer is a conservative approach. The two main directions for this problem are, on one hand, introducing the saliency maps where the features of the layers are larger but with less parameters in the branch for saliency (the firsts layers). And on the other hand, introducing them in later stages with more layers in the branch for saliency (and consequently more parameters), but where the features maps are smaller. In the chosen approach the output of the branch for saliency is large enough to provide meaningful visual information and at the same time there are enough parameters to represent a function that properly represent the saliency maps.

The main inconvenience of this approach is the number of parameters added to the network. The secondary branch added for the saliency maps introducing a total of $626,016$ paramenters, $11,616$ from the first convolutional layer and $614,400$ from the second.

Table 3.5: Specifications for the first proposal for the two branches network

Parameters	Description
Image sizes	128×128
Initial weights	Random
Mean subtraction	No
Number of epochs trained	90
Methodology to add saliency	Two branches merged after two convolutional layers

3.6.2 Merging before conv-2

To test whether the conservative approach is a good strategy and consider possible future lines, another similar experiment was carried out. This time the two input branches only present one convolutional layer, one batch normalization layer and a maxpooling layer, as shown in the figure 3.3. This approach aims to take advantage of the larger features from the first convolutional layer requiring less parameters, since only $11,616$ parameters are added to the original structure.

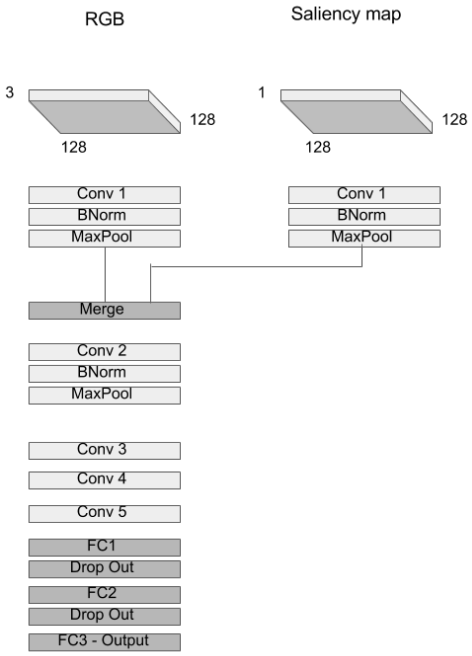


Figure 3.3: Two input architecture where both the saliency maps and the images, only pass through one convolutional, batch normalization and maxpooling layer. Then, they merge into the second convolutional layer.

Table 3.6: Specifications for the second proposal for the two branches network

Parameters	Description
Image sizes	128×128
Initial weights	Random
Mean subtraction	No
Number of epochs trained	90
Methodology to add saliency	Two branches merged after one convolutional layer

3.6.3 Skipped conv-2 layer

The last approach concerning the fan-in strategy is depicted in the figure 3.4. In this case, to reduce the number of parameters from the CNN, the second layer is removed from the previous experiments (convolutional layer, batch normalization and maxpooling layers). After merging the two branches, the output goes directly to what before was the third convolutional layer. This structure aims to reduce the parameters from the network.

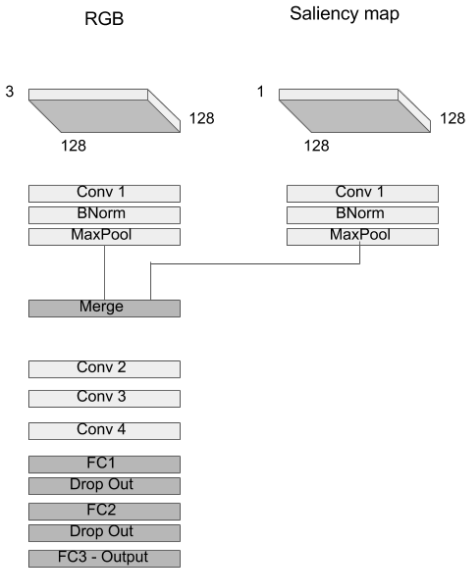


Figure 3.4: In this architecture, which present the same architecture as the previous one, the second convolutional, batch normalization and maxpooling layers were removed. Consequently, the branches merge into what before was the third convolutional layer.

Removing the mentioned layers implies that the parameters from the third convolutional layer are also reduced, from the $884,736$ in the original layer to $332,776$ after removing the previous convolution. This reduction of parameters from the third convolutional layer ($884,736 - 332,776 = 551,960$), together with the removal of the second convolutional layer ($614,400$) and the addition of the parameters from the second branch ($11,616$), leaving the network with $1,154,744$ parameters fewer than the original AlexNet. Surprisingly, as it is explained in the section 4.4.3, it still brings an improvement of the accuracy.

Table 3.7: Specifications for the third proposal for the two branches network

Parameters	Description
Image sizes	128×128
Initial weights	Random
Mean subtraction	No
Number of epochs trained	90
Methodology to add saliency	Same as previous removing the second convolutional layer

Chapter 4

Results and evaluation

In this chapter the results obtained during the development of this work are exposed. Before presenting the results that show the highest increase in the accuracy, preliminary experiments are presented and discussed. This chapter shows the graphs obtained during the different stages, compare them and evaluate the performance of the CNN under diverse circumstances

First, the baseline used to validate the results of other experiments is defined. Then, the results from all the experiments that lead to the architecture that successfuly introduces the saliency maps are showed and evaluated. Finally, a brief analysis of the accuracy showed by the models per each class is evaluated.

4.1 Baselines without saliency)

The results plotted in this section correspond to the AlexNet CNN trained with ImageNet for 90 epochs. The curves showed in the figure 4.1 were obtained during training and validation with the 128×128 and 227×227 images. The metrics showed are the Top-1 and Top-5 accuracy and the loss, as well as the values for the learning rate during training. These curves are meant to be the reference for following experiments. These results constitute a good basis to compare with the following experiments, even though the results are not reaching the accuracies reported in previous studies [28].

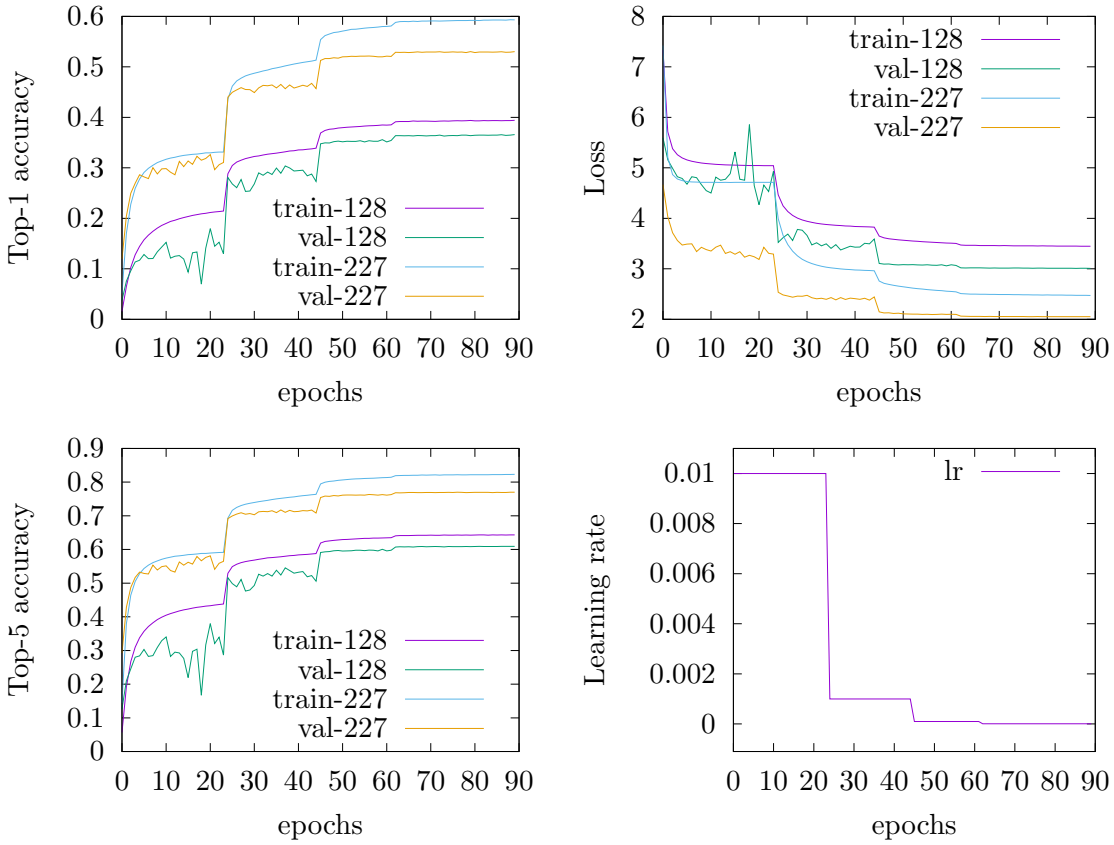


Figure 4.1: Training and validation curves for AlexNet with 128×128 and 227×227 images.

4.2 Saliency with pre-trained weights

The results presented in this section show how a trained network evolves after introducing the saliency maps and training for 30 more epochs. These results are used to point the following experiments toward one or another direction, as well as to provide an example on how does affect to the CNN the manipulation of already trained weights. Figure 4.2 shows the results from the six experiments described in section 3.4.

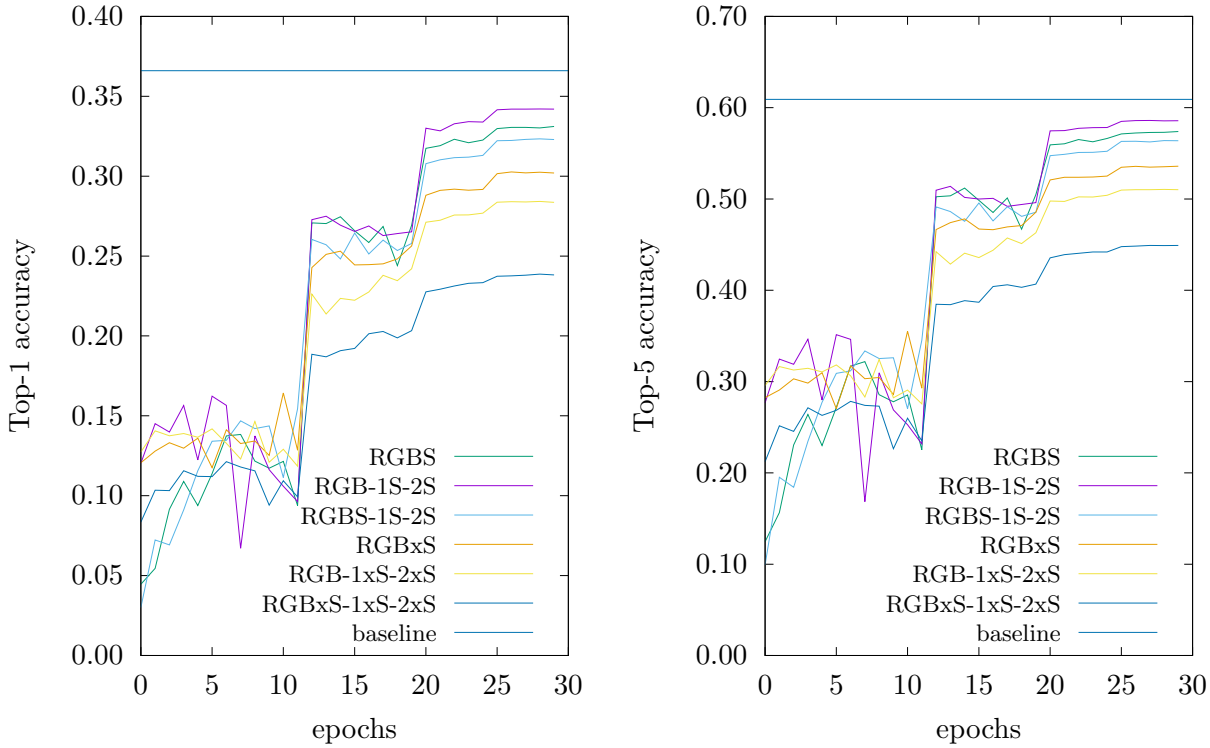


Figure 4.2: The six experiments done with pre-trained weights are ploted in the figure. *RGBS* refers to the experiment where the three channels of the images (*RGB*) and the saliency maps (*S*) are concatenated as a four channels input. In *RGB-1S-2S* the saliency maps are concatenated to the first and second convolutional layers. In *RGBS-1S-2S* the saliency maps are concatenated to the images and to the first and second convolutional layers. In *RGBxS* the saliency maps are multiplied by each of the three channels from the input images. In *RGB-1xS-2xS* the saliency maps are multiplied to the feature maps from the first and second convolutional layers. Finally, in *RGBxS-1xS-2xS* the saliency maps are multiplied by the three channels of the images and by the feature maps of the first and second convolutional layers.

From the plots in figure 4.2 it can be seen that the strategies where the saliency maps are concatenated (curves *RGBS*, *RGB-1S-2S* and *RGBS-1S-2S* from 4.2) outperform those where they are multiplied (*RGBxS*, *RGB-1xS-2xS* and *RGBxS-1xS-2xS*); multiplying the saliency maps appear to be eliminating contextual information useful for classification. Furthermore, we can see how the best performance comes from *RGB-1S-2S*, where the saliency maps were concatenated to the first and second convolutional layers. That is not surprising since we are concatenating the saliency maps in more layers of the CNN than in the experiment *RGBS*, where the saliency is only concatenating at the beginning. However, the fact that the accuracy for the experiment *RGBS* is higher than the accuracy for *RGBS-1S-2S* is an unexpected result, since in this last one, more saliency maps are concatenated. A reasonable explanation to this results is that the network with more saliency maps needs more time to train since it has more parameters. Additionally, since more weights have been introduced the solution is farther from a local optimum.

4.3 Saliency concatenated to existing layers

The purpose of this section is to show the results obtained from the experiments that provided the two best accuracies in the previous section. Those are the experiments where the saliency maps are added as an extra channel in the input RGB image4.3, or to the first and second convolutional layer (figure 4.4). These results are plotted together in figure 4.5 to compare its performance and evaluate the consistency of the procedure followed to choose this experiments.

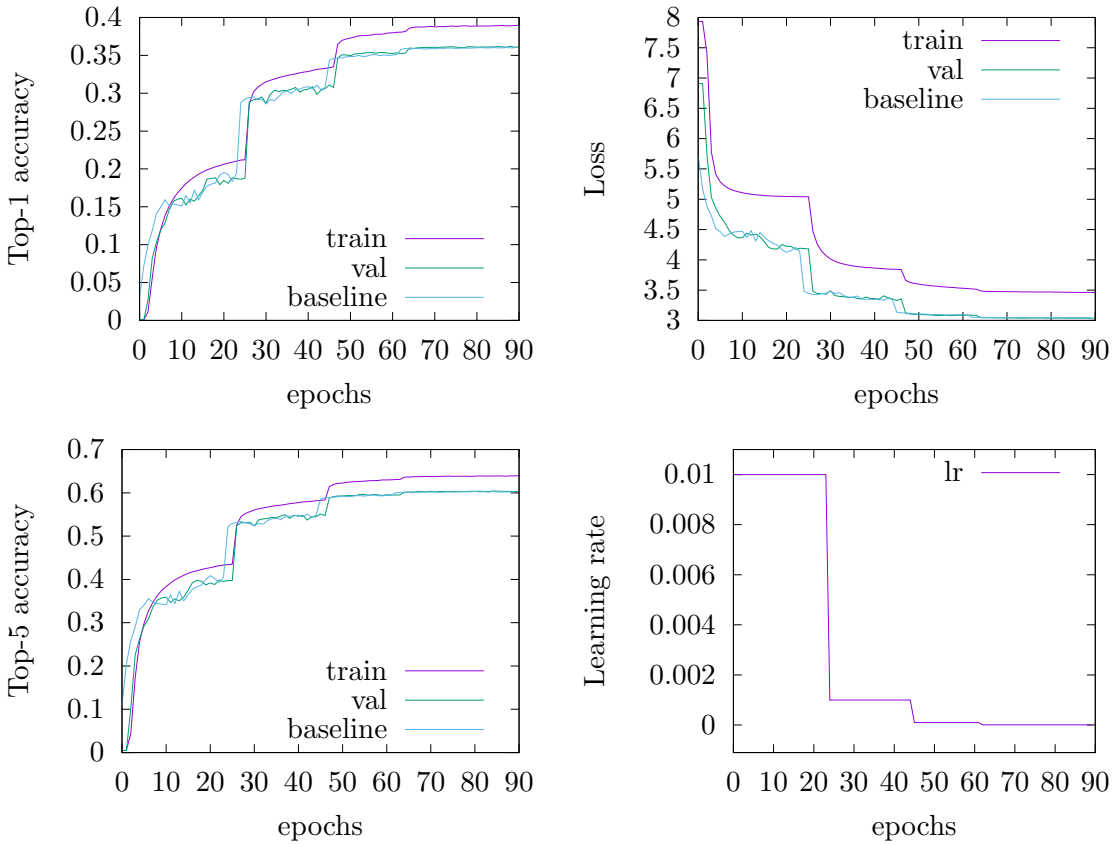


Figure 4.3: Training and validation curves for the experiment where the saliency maps are concatenated as a fourth channel in the input (*RGBS*).

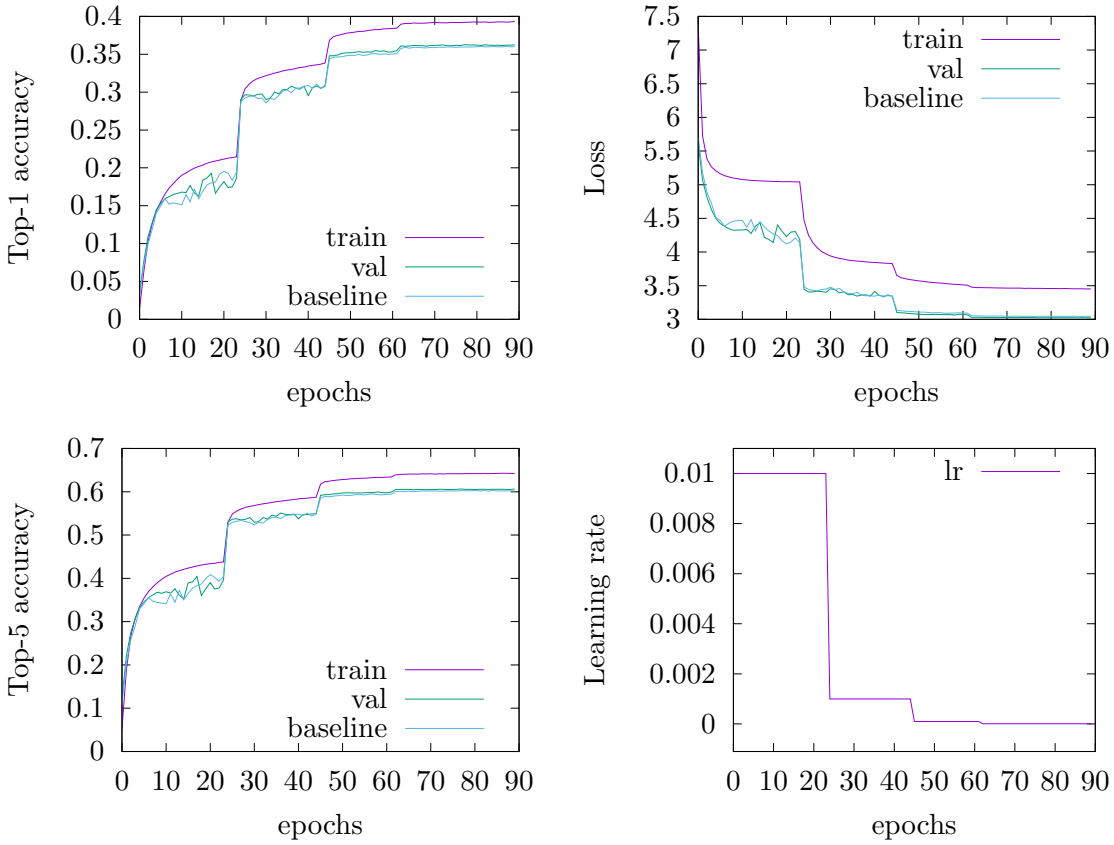


Figure 4.4: Training and validation curves for the experiment where the saliency maps are concatenated o the feature maps from the first and second convolutional layers (*RGBS-1S-2S*).

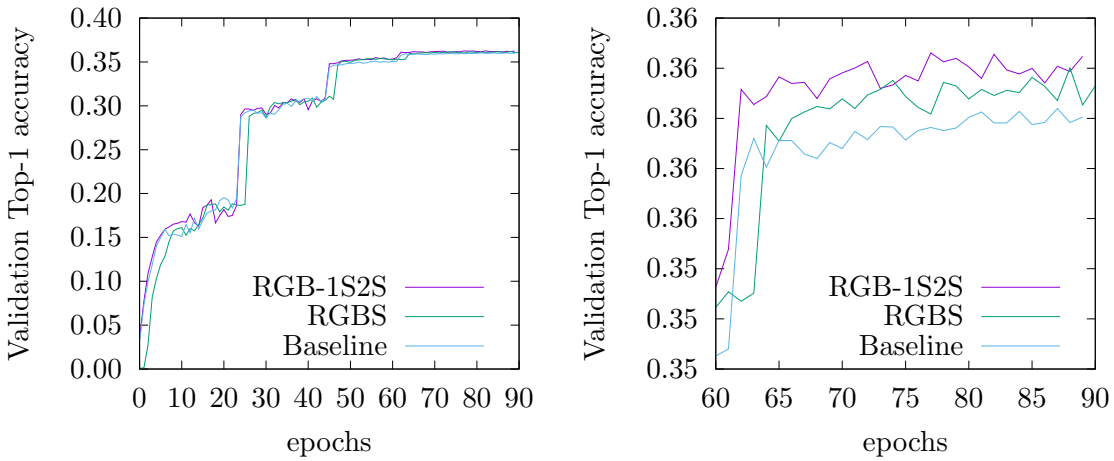


Figure 4.5: Comparison of the Top-1 accuracy obtained in the experiments *RGBS* and *RGB-1S-2S*

Even though a minor improvement in accuracy can be noted in the Top-1 plots from the figure 4.5, the results are rather disappointing. A first conclusion might be that concatenating

an extra channel to the 96 feature maps from the first convolutional layer and another to the 256 feature maps from the second convolutional layer is too small a contribution. There might not be enough information to provide a meaningful increase of the accuracy. In *RGBS* the saliency maps are introduced in the input concatenated to the other three channels, in this case the contribution is more noticeable. These results are further discussed in the Chapter 5.

The evident need of another strategy to introduce the saliency maps motivated us to approach the problem with a completely different methodology introduced in the section 3.6, and which results are showed in the following section.

4.4 Fan-in network

4.4.1 Merging before conv-3

The results shown in this section correspond to the experiments described in the section 3.6.1. Here the network has two identical input branches, one for the saliency maps and another for the images. The plots from the figure 4.6 show the training and validation curves obtained while training the network. With this architecture an increase of 1.72% is obtained in the Top-1 accuracy.

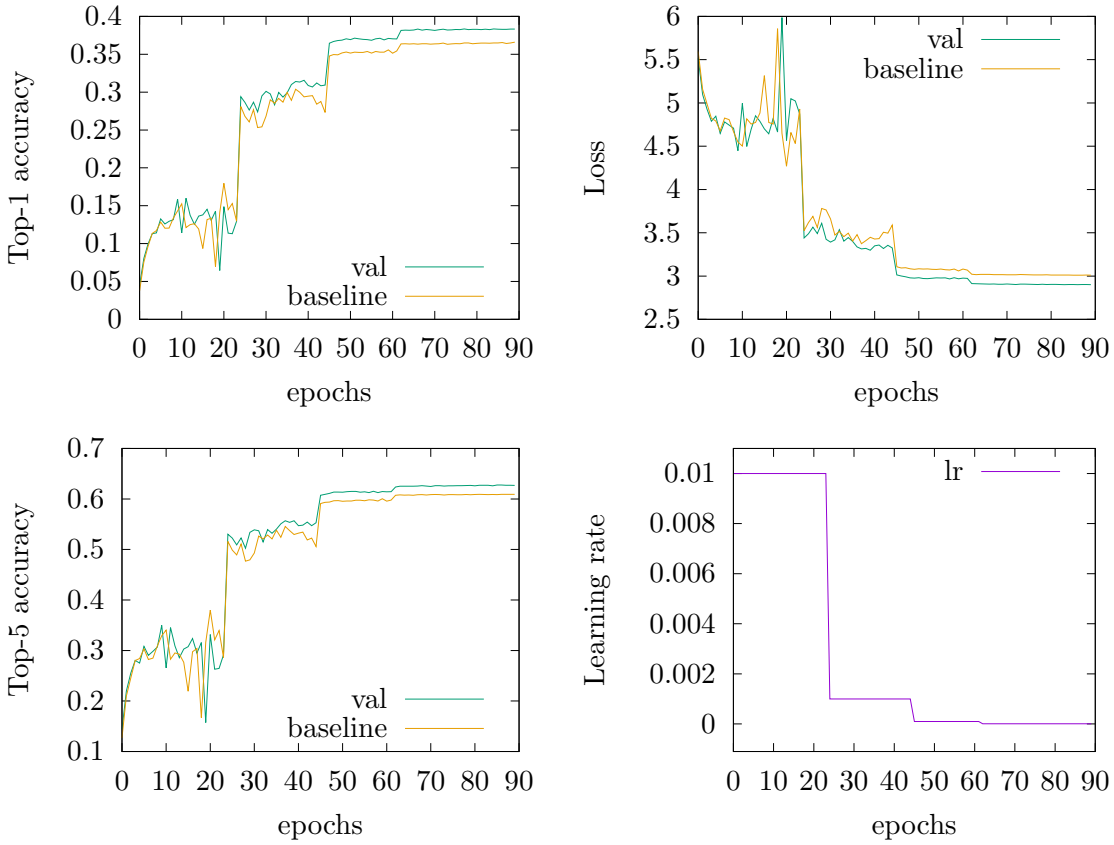


Figure 4.6: Training and validation curves for the first two input branches architecture (also referred as *Fan-in1*) where both the saliency maps and the images pass through two convolutional layers as explained in 3.6.1.

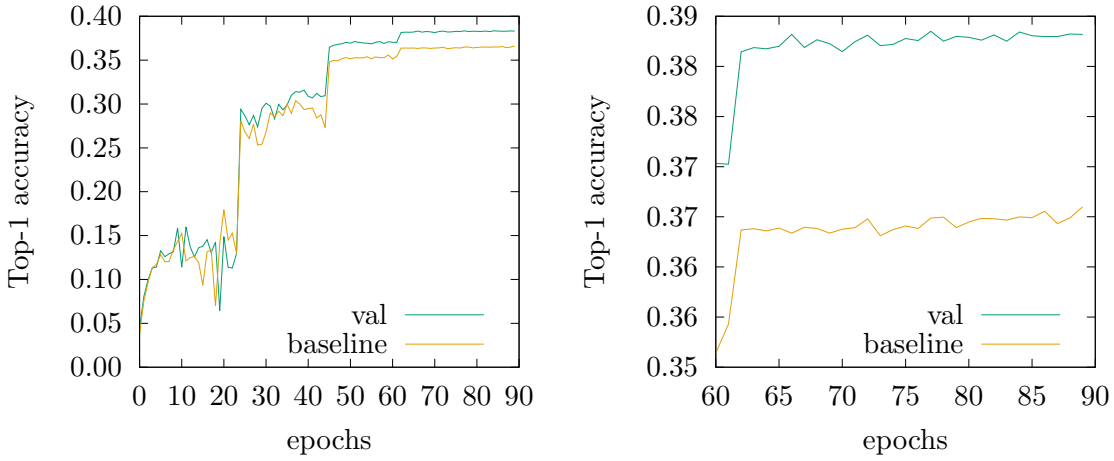


Figure 4.7: Top-1 accuracy of the first implementation of the CNN architecture with two input branches (*Fan-in1*).

These results lead to suggest that providing an input branch for the images and one for the saliency maps is a good solution for the problem stated in the thesis. Consequently, numerous

possibilities are opened to exploit the information from the saliency maps. The first of them is addressed in the following section, where the importance of the size of the feature maps when the branches are merged is discussed.

Due to the improvement on the accuracy showed by this approach the experiment has been reproduced with the larger images and the results are discussed in the section 4.6.

4.4.2 Merging before conv-2

As it was mentioned in the section 3.6 this approach introduces the saliency maps after those passing through a convolutional layer. The results corresponding to this architecture are plotted in 4.8. This approach aims to take advantage of the bigger sizes of the saliency maps when the branches are merged in more shallow layers.

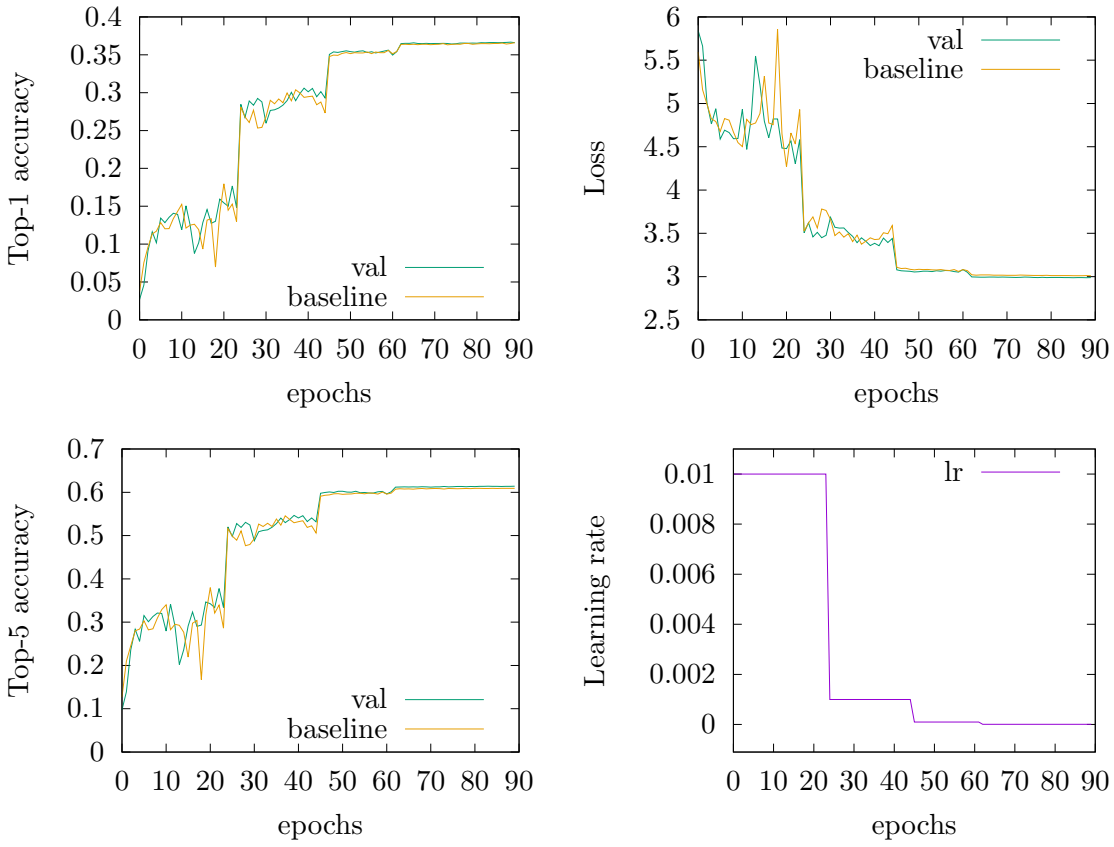


Figure 4.8: Training and validation curves for the two input branches architecture (also referred as *Fan-in2*) where both the saliency maps and the images pass through only one convolutional layers as explained in 3.6.2.

The slight improvement that can be appreciated in figure 4.9 is far from be meaningful. In fact, throughout the validation curves there is no remarkable difference between this experiment and the baseline, what indicate a low contribution from the saliency maps during training.

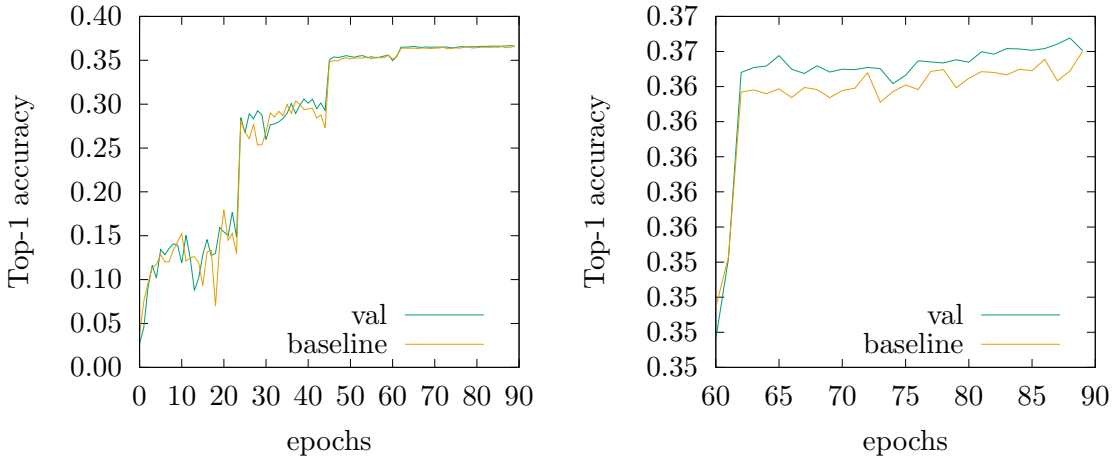


Figure 4.9: Top-1 accuracy of the second implementation of the CNN architecture with two input branches (*Fan-in2*).

4.4.3 Skipped conv-2 layer

The results showed in this section correspond to a network which main novelty is that one convolutional layer, one batch normalization and one maxpooling layer have been removed. Consequently the CNN have less parameters than the original AlexNet and probably, due to the information introduced by the saliency maps, its performance is higher in terms of accuracy. Even if the results presented correspond to the approach that needs less parameters, in the image 4.10 it can be seen that there is a 12.4% of improvement of the accuracy comparing with the baseline. However, in the Chapter 6 it is proposed to study the behavior of this architecture without saliency maps.

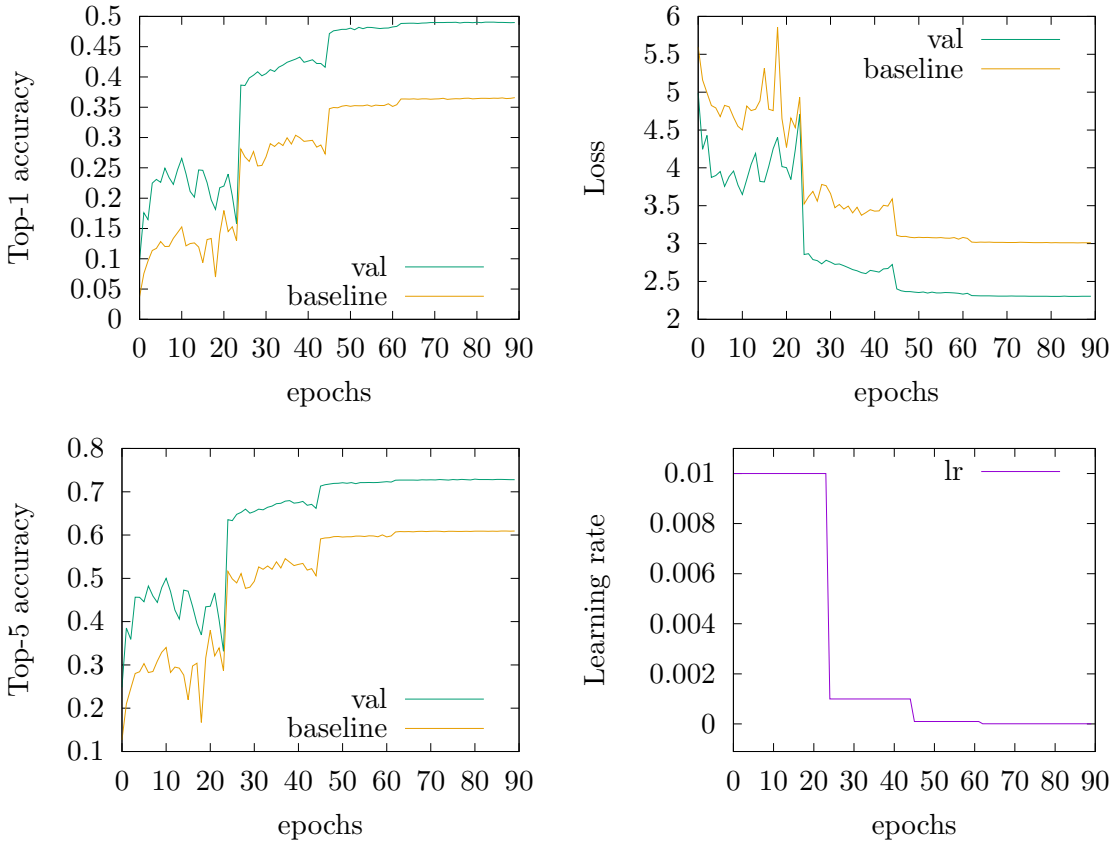


Figure 4.10: Training and validation curves for the two input branches architecture (also referred as *Fan-in3*) where several layers had been removed as explained in 3.6.3.

The structure of the CNN for this experiment is described in the section 3.6.3. The first hypothesis for the surprisingly high improvement in the accuracy was attributed to the fact that the saliency information and the images are mixed when the feature maps still have a reasonable size to present visual coherence. However, the results from the previous section (4.4.2) suggest that this increase of accuracy might be due to the removal of several layers.

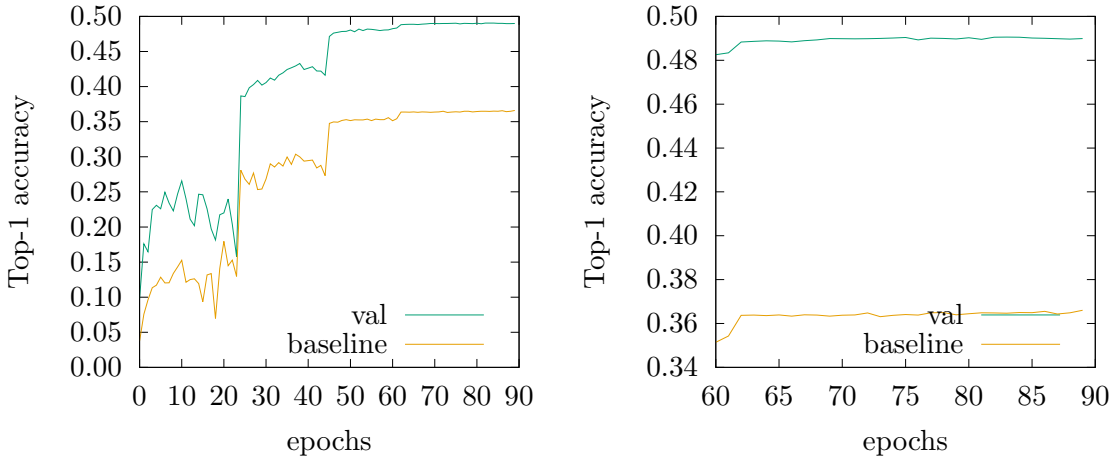


Figure 4.11: Top-1 accuracy of the third implementation of the CNN architecture with two input branches (*Fan-in3*).

4.4.4 Comparing the strategies

Comparing the results obtained with the baseline and the previous strategies, a significant increase of the accuracy can be noticed when the structure used is the one explained in section 3.6.3. A highly interesting step after this results would be to run this experiment with full size images in order to see whether the improvement is as scalable as expected. In the section 4.6, though, the results for the first approach trained with the larger images are shown.

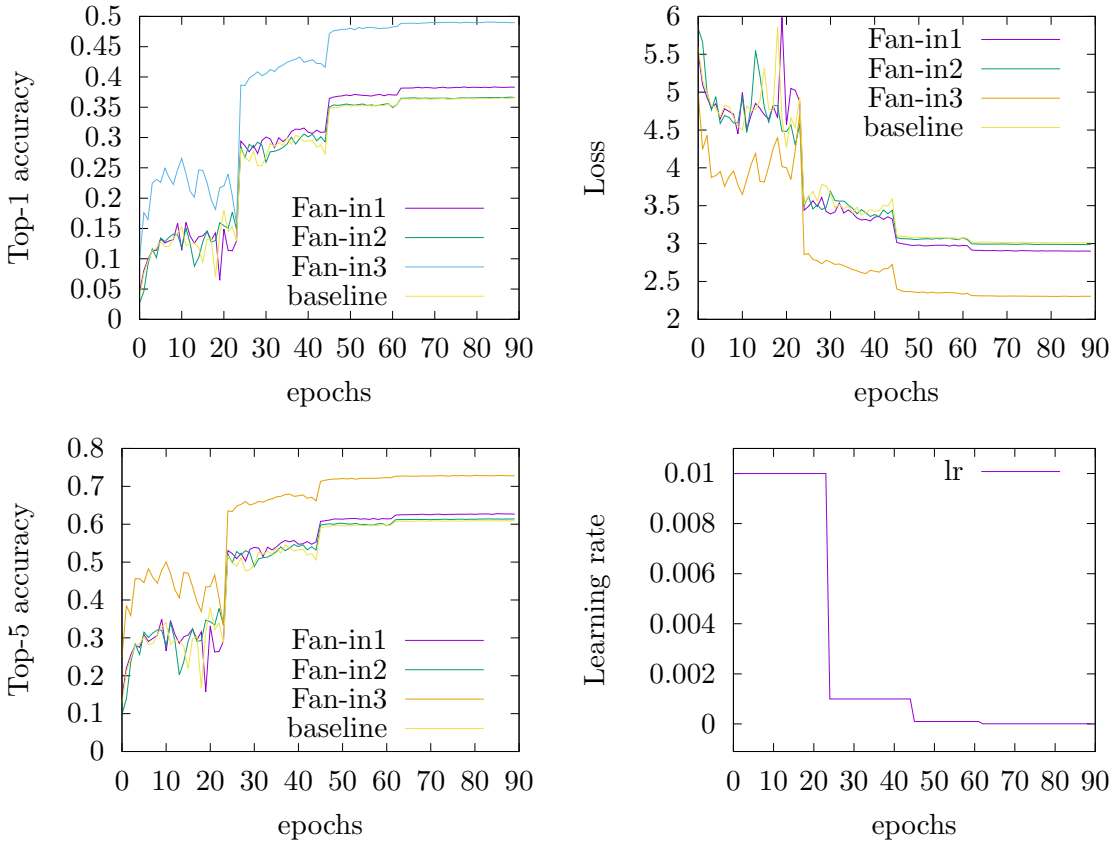


Figure 4.12: Comparison of the results for the three proposals with two input branches.

A first conclusion was that allowing the network to learn how to represent the saliency maps before adding them to the original CNN is a successful approach, even if the global number of parameters of the network are reduced (see results *Fan-in3* from figure 4.13). However, the results plotted in the curve *Fan-in2* from the figure 4.13 show a poor improvement of the accuracy. The highly remarkable difference between the results from this two approaches suggest that this increase of accuracy is not entirely due to the addition of the branch for saliency, but for the removal of the second convolutional layer of the original CNN architecture.

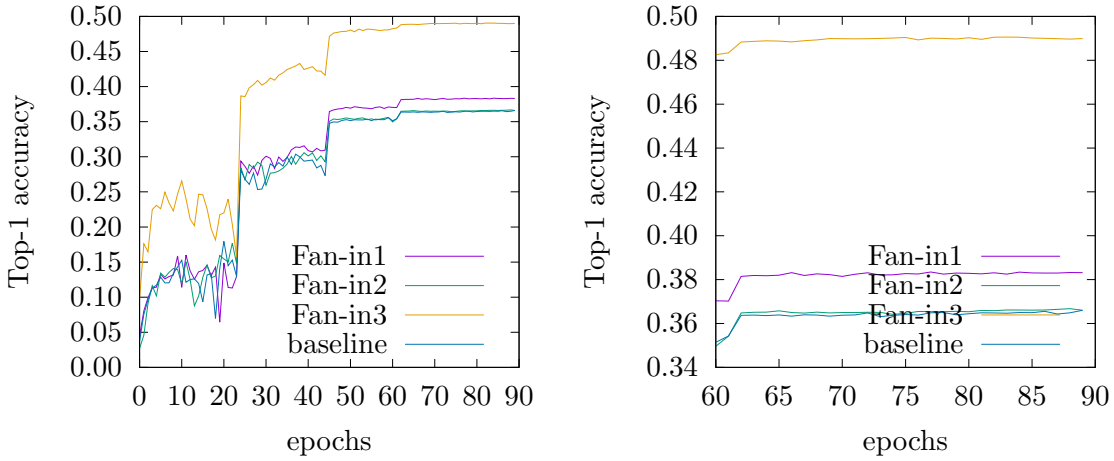


Figure 4.13: Top-1 accuracy of the three experiments with the two branch implementation

To go deeper into these strategies some other experiments are proposed in chapter 6. Table 4.1 provides a numerical overview of the experiments done with the networks trained all from scratch. Something considered in the chapter 6 (future work) is that two of the three structures with two input branches outperform the strategies were the saliency maps were directly added to the network. Setting this way an interesting path to keep exploring the contribution of the saliency predictions.

Table 4.1: Summary of the results from the different strategies used to add the saliency maps. Note that RGB is the network trained subtracting the mean from the images and the saliency maps. While in RGB* the mean is no subtracted.

Structure	Top-1	Top-1 improvement	Top-5	Top-5 improvement
Saliency concatenated to existing layers				
RGB	36.00%	0.00%	60.21%	0.00%
RGBS	36.60%	0.60%	60.35%	0.14%
RGB_1S2S	36.25%	0.25%	60.62%	0.41%
Saliency merged from a new branch				
RGB*	36.60	0.00%	60.91%	0.00%
Fan-in1	38.32	1.72%	62.69%	1.78%
Fan-in2	36.60%	0.00%	61.40%	0.49%
Fan-inbr3	48.99%	12.39%	72.80%	11.89%

4.5 Mean subtraction

The results exposed in this section correspond to additional experiments and its purpose is to study how does affect to the final result the subtraction of the mean of the inputs. In the corresponding cases, the mean has been subtracted from the saliency maps and from each of the three channels from the image. In the figure 4.14 the results for different strategies are showed with and without mean subtraction.

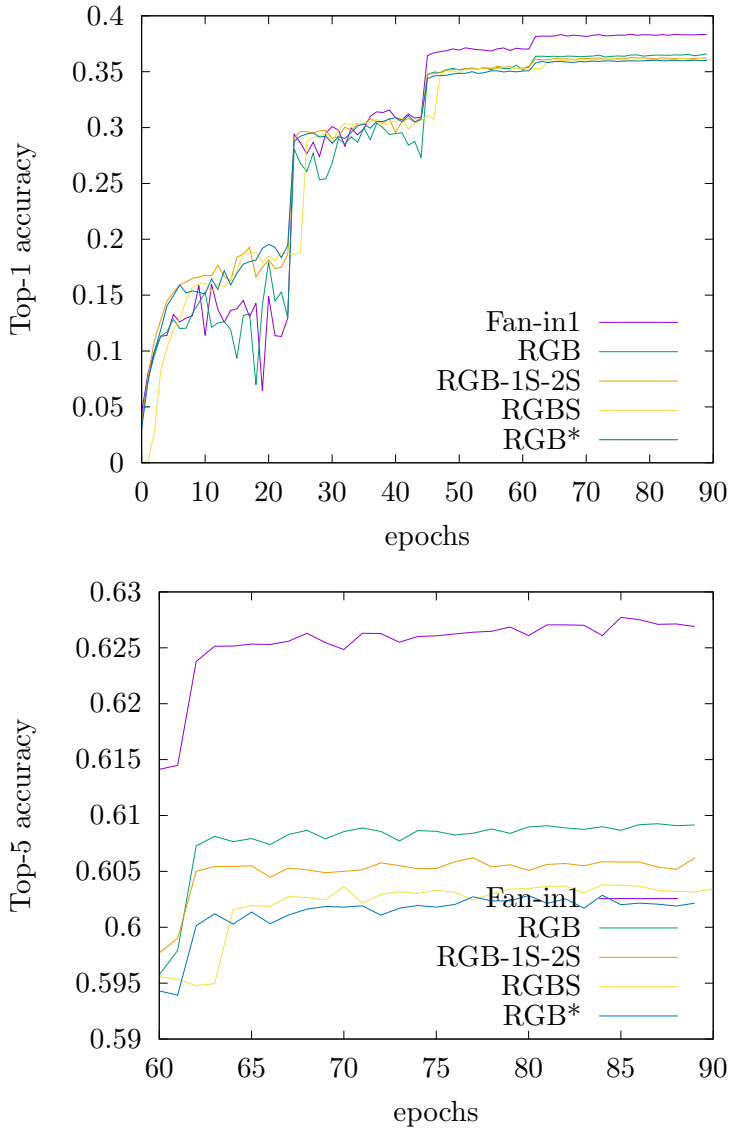


Figure 4.14: The experiments where the meas was subtracted from the images and the saliency maps are: *RGB*, *RGB-1S-2S* and *RGBS*. The experiments were the mean was not subtracted nor from the images nor from the saliency maps are: *Fan-in1* and *RGB**.

From the results plotted in Figure 4.14 it can be seen that the experiments where the mean has not been subtracted from the input images nor from the maps provide higher accuracies

than those where it has been subtracted. The curve *RGB* shows a slight increase in the accuracy compared with *RGB**, which is the same experiment with mean subtracted. In the cases where the saliency maps are introduced the explanation is more evident. When the saliency values are mean subtracted they can take negative values. Then, when these are introduced in the middle of the network, they are concatenated with the outputs of a ReLU, which only take positive values. This shifts the distribution of the inputs to the subsequent layers, making learning more difficult.However, the reason because the reduction of accuracy in *RGB** is lower than in *RGB* is not clear.

Since the network is using Batch Normalization layers, the images and the maps are already being normalized, maintaining the mean close to zero and the standard deviation close to one, reducing this way the covariance shift as mentioned in section 2.1.1. Consequently, subtracting the mean of the images when a Batch Normalization is being used is not necessary, in fact, in our case it is reducing the accuracy. However, in the experiments done with the classical CNN structure, the mean was subtracted from both, the images and the saliency maps.

4.6 Full size images

The results exposed in this section correspond to two of the successful approaches applied to full size images (227×227): when the saliency maps are concatenated o the input as a fourth channel (*RGBS* 3.5), and the two input branches approach where the maps and the images pass by two convolutional layer (i.e. *Fan-in1* 3.6.1). In figure 4.16 can be seen an increase in the accuracy from both approaches, which confirms the statement that the saliency maps (and consequently the models trained to predict visual saliency) can be applied to improve the performance of CNN in the classifications task. Specifically in the approach *Fan-in1*, the improvement of the performance is much higher, increasing the accuracy performance in 1.35%.

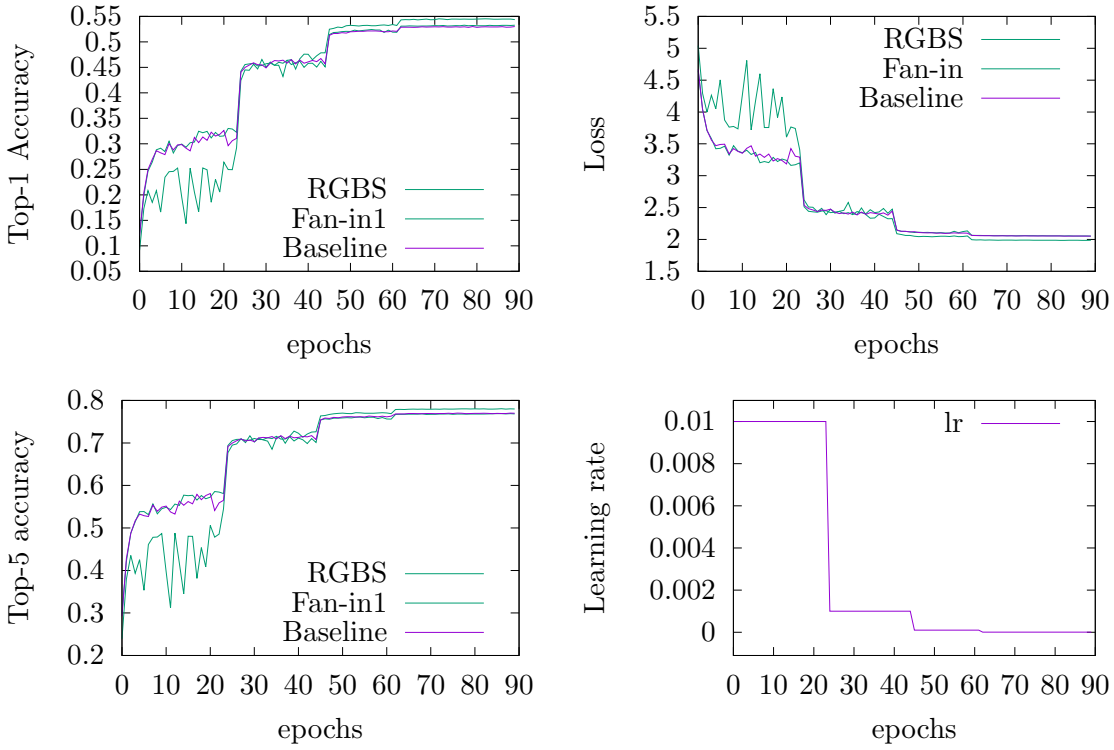


Figure 4.15: Training curves for the experiments done with 227 × 227 images.

These results provide an evidence of how the improvements of accuracy obtained with smaller images result in improvements with the larger images. Consequently, a first deduction leads us to suggest that the approach explained in 3.6.3 that provided 12.39% increase of the accuracy, might provide a higher increase than the experiments currently done in this section, being also in 227 × 227 images the best option to add saliency maps to the image classification task. Consequently, in the chapter 6 it is mentioned as one of the proposals for future lines.

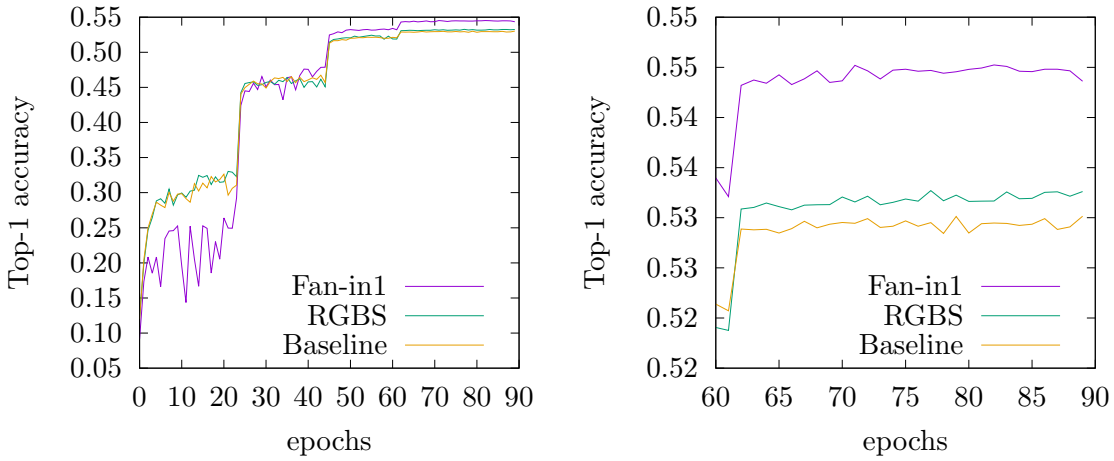


Figure 4.16: Training curves for the Top-1 accuracy for experiments done with 227 × 227 images.

4.7 Analysis of per-class improvements

The purpose of this section is to carry out a deeper study of the models trained during the development of the thesis. The objective is to evaluate those models that did not improved its performance with the introduction of the saliency maps and see if there has been some improvement at class level. For that, the three classes which have had the major improvement in accuracy after the introduction of the saliency maps, are displayed in the table 4.2.

Table 4.2: Three classes per strategy that most improved when the saliency maps are introduced

Structure	Increase in accuracy	Label
	128×128 images	
RGBS	26%	French bulldog
	25%	Spatula
	23%	Crane
RGB_1S2S	100%	Spatula
	23%	Shield, buckler
	21%	Crane
Fan-in1	36%	Black-and-tan coonhound
	33%	Spatula
	25%	Streetcar, tram, tramcar, trolley, trolley car
Fan-in2	30%	Can opener, tin opener
	22%	Hair spray
	22%	Gar, garfish, garpike, billfish, Lepisosteus osseus
Fan-in3	46%	Cradle
	43%	French bulldog
	43%	Space shuttle
	227×227 images	
RGBS	22%	Chiffonier, commode
	20%	Chain
	19%	Thimble
Fan-in1	25%	Ox
	23%	Head cabbage
	21%	Red-backed sandpiper, dunlin, Erolia alpina

The table shows for each model the index of the category that has improved the most, the increase of accuracy experienced by each class and the label corresponding to that class. Something surprising is that each model has several classes whose accuracies have improved. If

the global accuracy of the model has not improved, means that the accuracy for other classes must have decreased.

At the same time worth mentioning that those models that had improved the most are also those that are showing classes with higher rates of improvement. A reasonable justification for that, might be that the saliency maps are teaching the model to learn some classes more than the others. This might be also influenced by the dataset used to train the model that generated the saliency maps. In the Chapter 6 there are some proposals to continue this study, since it can bring valuable information about saliency predictions and possible applications.

Chapter 5

Conclusions

The main goal of the project was to study how visual predictions affect the performance of a CNN in the task of image classification. In the introduction of the thesis (in Chapter 1), as well as in th state of the art section (in Chapter 2), it was remarked the importance of the visual attention on how the humans recognize objects and scenes. This statement strongly suggested the possibility that the information of visual saliency might be also used for CNN in the task of image classification.

The most meaningful conclusion of this chapter is that CNNs trained for saliency prediction can be used to improve the accuracy of other CNNs in the classification task. From the last experiment with large images it can be seen how the model that improved the accuracy in the previous experiments with reduced images is also providing higher accuracy when the images have the original dimensions. Showing thus, how the experiments done to understand how the networks are able learn from the saliency maps, are scalable to the larger images.

The conclusion from the practical point of view refers to the strategy to add the saliency maps, which is not trivial. A good strategy is giving to the network freedom to learn the features that best represent the maps in order to add them to the images. Significant improvements have been achieved from the experiments with a fan-in architecture.

A first hypothesis coming from this results is that the contribution in terms of accuracy do not come from the global number of parameters of the network and how are they distributed. Since significant results has been achieved training (using small images) a structure with less parameters than the original AlexNet.

From the experiments in the section 4.2 a first conclusion was that the best way to add the saliency maps was concatenating them to the feature maps instead of multiplying them. That made sense, since the multiplication would cancel regions from the image with possible contextual information. However, in the section 4.3 the results show that the improvement between methods is not as significant as expected based on the results from 4.2. The conclusion

from here is that this methodology is providing information of how the addition of saliency to an already trained model affects. The accuracy of this models after 30 epochs depend more on the quantity of weights that has been manipulated than on the structure being tested.

Another contribution of this work is the proposal of a fan-in architecture that combines saliency maps and the raw images reducing the number of parameters used in AlexNet and providing higher performance in terms of accuracy. This fact shows how significant the contribution of saliency might be. The network was only tested with the small images, consequently remains as future work (discussed in Chapter 6) training this proposal with the original size images.

During the first stages of the project many drawbacks were faced, specially due to the lack of experience dealing with CNN and large datasets. After building the first model and training with smaller datasets we moved to the challenging ImageNet, which made the evolution of the project slower but at the same time much more interesting and engaging.

Having in mind the process followed and the results showed during this report, the following chapter is dedicated to provide some suggestions or future lines to keep exploring the possibilities of the salient visual information as well as techniques to mix knowledge between models.

Chapter 6

Future work

The purpose of this chapter is to describe possible future experiments and propose lines of study that were not approached during the development of this work due to lack of time.

Most of the proposals aim to understand how the last strategy to add the saliency maps can be optimized. Two questions arise from the results obtained. The first one is whether this branch can be added to an already trained network, re-train it for few epochs and still improve the accuracy. If it does, whether the time required is worth the improvement provided or not. The second question is how this solution can be optimized. In this thesis three approaches had been tested. Therefore, there are many experiments that might lead to a better solution. Those proposals can be organized as follows:

- Those experiments that use two identical input branches after different layers of the original network, and those where the same branch is added at different points. This approaches would aim to find a trade-off between number of extra parameters added and the position in the network where to merge the branches.
- Several experiments can be done inspired in the study [45]. They transfer learning from one network to the other, forcing the one that is learning to pay attention to the same features that the other network does. On the one hand, in our case some approaches might follow the line of introducing the saliency maps on a second input and merge both branches several times after several layers. On the other hand, some different strategies might be followed pointing to have a closer solution to the one presented in the study. That might be done, for instance, using the classification network as the network learning the task and the CNN that generates the saliency map as the network that teaches where to pay attention.

At the same time, in order to discard the possibility that the improvement of accuracy in the last experiment comes from removing some layers, it worth studying the CNN's behavior

without saliency maps.

Aside from the proposals aiming to improve the addition of the saliency branch, there are many possibilities of study regarding the new architecture proposed. The starting point would be to do the experiments with the full size images.

Bibliography

[1] Convolutional neural networks for visual recognition. `http://cs231n.github.io/neural-networks-1/`.

[2] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[3] Zoya Bylinskii, Tilke Judd, Ali Borji, Laurent Itti, Frédo Durand, Aude Oliva, and Antonio Torralba. Mit saliency benchmark, 2015.

[4] Charles F Cadieu, Ha Hong, Daniel LK Yamins, Nicolas Pinto, Diego Ardila, Ethan A Solomon, Najib J Majaj, and James J DiCarlo. Deep neural networks rival the representation of primate it cortex for core visual object recognition. *PLoS Comput Biol*, 10(12):e1003963, 2014.

[5] François Chollet. Keras. `https://github.com/fchollet/keras`, 2015.

[6] Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3642–3649. IEEE, 2012.

[7] Philippe Pérez de San Roman, Jenny Benois-Pineau, Jean-Philippe Domenger, Aymar De Rugy, Florent Paclet, and Daniel Cataert. Saliency driven object recognition in egocentric videos with deep cnn. 2016.

[8] Sander Dieleman, Jan Schlüter, Colin Raffel, Eben Olson, Søren Kaae Sønderby, Daniel Nouri, Daniel Maturana, Martin Thoma, Eric Battenberg, Jack Kelly, Jeffrey De Fauw, Michael Heilman, Diogo Moitinho de Almeida, Brian McFee, Hendrik Weideman, Gábor

Takács, Peter de Rivaz, Jon Crall, Gregory Sanders, Kashif Rasul, Cong Liu, Geoffrey French, and Jonas Degrave. Lasagne: First release., August 2015.

[9] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Icml*, volume 32, pages 647–655, 2014.

[10] Emmanouil Giouvanakis and Constantine Kotropoulos. Saliency map driven image retrieval combining the bag-of-words model and plsa. In *Digital Signal Processing (DSP), 2014 19th International Conference on*, pages 280–285. IEEE, 2014.

[11] Jonathan Harel, Christof Koch, Pietro Perona, et al. Graph-based visual saliency. In *NIPS*, volume 1, page 5, 2006.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[13] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[14] Chesti Altaff Hussain, D Venkata Rao, and S Aruna Masthani. Robust pre-processing technique based on saliency detection for content based image retrieval systems. *Procedia Computer Science*, 85:571–580, 2016.

[15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[16] Ming Jiang, Shengsheng Huang, Juanyong Duan, and Qi Zhao. Salicon: Saliency in context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1072–1080, 2015.

[17] Konstantinos Kamnitsas, Christian Ledig, Virginia FJ Newcombe, Joanna P Simpson, Andrew D Kane, David K Menon, Daniel Rueckert, and Ben Glocker. Efficient multi-scale 3d cnn with fully connected crf for accurate brain lesion segmentation. *Medical Image Analysis*, 36:61–78, 2017.

[18] Nour Karessli, Zeynep Akata, Andreas Bulling, and Bernt Schiele. Gaze embeddings for zero-shot image classification. *arXiv preprint arXiv:1611.09309*, 2016.

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

[20] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[22] Srinivas SS Kruthiventi, Vennela Gudisa, Jaley H Dholakiya, and R Venkatesh Babu. Saliency unified: A deep architecture for simultaneous eye fixation prediction and salient object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5781–5790, 2016.

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

[24] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*, pages 609–616. ACM, 2009.

[25] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014.

[26] Kevin McGuinness. Compaction code, 2016.

[27] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[28] Dmytro Mishkin, Nikolay Sergievskiy, and Jiri Matas. Systematic evaluation of cnn advances on the imagenet. *arXiv preprint arXiv:1606.02228*, 2016.

[29] Eva Mohedano, Kevin McGuinness, Noel E O'Connor, Amaia Salvador, Ferran Marqués, and Xavier Giró-i Nieto. Bags of local convolutional features for scalable instance search. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*, pages 327–331. ACM, 2016.

[30] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4293–4302, 2016.

[31] Junting Pan, Elisa Sayrol, Xavier Giro-i Nieto, Kevin McGuinness, and Noel E. O'Connor. Shallow and deep convolutional networks for saliency prediction. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[32] Junting Pan, Elisa Sayrol, Xavier Giro-i Nieto, Kevin McGuinness, and Noel E. O'Connor. Shallow and deep convolutional networks for saliency prediction. June 2016. `https://github.com/imatge-upc/saliency-2016-cvpr`.

[33] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[34] Cristian Reyes, Eva Mohedano, Kevin McGuinness, Noel E O'Connor, and Xavier Giro-i Nieto. Where is my phone?: Personal object retrieval from egocentric images. In *Proceedings of the first Workshop on Lifelogging Tools and Applications*, pages 55–62. ACM, 2016.

[35] Nicolas Riche, Matthieu Duvinage, Matei Mancas, Bernard Gosselin, and Thierry Dutoit. Saliency and human fixations: State-of-the-art and study of comparison metrics. In *Proceedings of the IEEE international conference on computer vision*, pages 1153–1160, 2013.

[36] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[37] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astoundiaseline for recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 806–813, 2014.

[38] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[39] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[40] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014.

[41] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016.

[42] Lijun Wang, Wanli Ouyang, Xiaogang Wang, and Huchuan Lu. Visual tracking with fully convolutional networks. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[43] Wikipedia. History of artificial intelligence — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/wiki/History_of_artificial_intelligence`.

[44] Wikipedia. List of datasets for machine learning research — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/wiki/List_of_datasets_for_machine_learning_research`.

[45] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016.

[46] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.