UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE UPV INGENIEROS DE TELECOMUNICACIÓN

Bachelor Thesis

Saliency Map Optimization

Author: Claudio Fernandez Martín
UPV Tutor: Joaquín Cerdá Boluda
TUM Cotutor: Tamay Aykut, M.Sc.

Bachelor Thesis presented in the Higher Technical School of Engineers in Telecommunication from the Polytechnic University of Valencia, to obtain the Bachelor's Degree in Telecommunication Technologies and Services Engineering.
Year 2019
Valencia, September 10, 2019

Acknowledgements

From these lines I would like to thank all those people who have shared with me the several stages of this beautiful and, sometimes arduous, journey. In the first place, to my mother for still giving me an education that goes beyond the academic field and that has shaped my way of being, my personality and my values in life. Secondly to the large amount of people who have collaborated in making my studies in the Polytechnic University of Valencia more pleasant. From one side, side my friends and family from Valencia; and specially, to my colleagues from the ARA group with whom I have shared long nights of study in the library and also extraordinary experiences outside the university. To all the international people that I have met during this last year in Munich and with whom I have created boundaries that will maintain our friendship on time despite the distance. Also, a special mention to Jesús Alonso from the International Office for his dedication and care to the students in mobility programs. Finally, I would like to thank my supervisors Ximo Cerdà and Tamay Aykut for giving me the opportunity of working in such stimulating and novel topic in two of the principal European universities.

Abstract

This Bachelor Thesis is divided in two main parts, both related with human visual attention. The first of them consists in the creation of new dataset which contains eye-tracking and head-position data of the users visualizing 360° immersive videos in a virtual reality environment. The second part comprises the design, development, training and testing of a model based on deep learning for the prediction of saliency maps in real time using Convolutional Neural Networks.

Resumen

Este Trabajo Fin de Grado (TFG) está seccionado en dos partes principales, ambas relacionadas con la atención visual humana. La primera de ellas consiste en la creación de un nuevo set de datos el cual contiene los valores de rastreo ocular y los movimientos de la cabeza de los usuarios al visualizar videos inmersivos de 360° en un entorno de realidad virtual. La segunda parte, comprende el diseño, desarrollo, entrenamiento y evaluación de un modelo basado en aprendizaje profundo para la predicción de mapas de prominencia en tiempo real utilizando redes neuronales convolucionales.

Resum

Aquest Treball Fi de Grau (TFG) està seccionat en dos parts principals, ambdues relacionades amb l'atenció visual humana. La primera d'elles consisteix en la creació d'un nou set de dades el qual conté els valors de rastreig ocular i els moviments del cap dels usuaris al visualitzar vídeos inmersius de 360° en un entorn de realitat virtual. La segona part, comprén el disseny, desenvolupament, entrenament i avaluació d'un model basat en aprenentatge profund per a la predicció de mapes de prominència en temps real utilitzant xarxes neuronals convolucionals.

Contents

Chapter 1

Introduction

After thousands of years of evolution, the task of recognizing objects and scenes has become a trivial and automated process for the human brain. In our case, we are rarely aware of the amount of time and energy that we spent on developing such neurological structures that allow us to make such a complex process possible to the point that it takes place inadvertently.

However, this process has been approached by the computer vision community during the last years. In practice, the main tool used for trying to recreate this cognitive process of our brain is deep learning, a sub-field inside the machine learning scope.

The building blocks of deep learning are the so-called Convolutional Neural Networks (CNN) which architecture and functioning are inspired on the biological processes of the human brain. This kind networks are composed by layers of multiple neurons. Each neuron performs an operation for a given input, and the output will be determined by the input and a set of specific parameters called *weights*.

These weights are determined during training which represents the process of how the CNN *learns*. The weights are periodically updated according to the training data, which is correlated to the task for which the network has been designed for.

The human visual system is able to identify current optical inputs considering previously learnt features, structures and patterns. CNNs work in the same way, starting with high level parameters, which will be shaped into lower levels of abstraction after training. These features, for instance, correspond to the detection of borders or edges in the scenes, which are part of higher level structures. Thereby, when the model has faced and learnt from experience the representation of these features at low level, it will be able to recognize them on higher levels.

One of the main characteristics of the human brain, is that it is not only capable of performing all the above mentioned tasks, but it is able to do it in real time. For a given scene or environment, the human brain is constantly processing the visual information, and

therefore, it is then able to react to unexpected situations. One of the aspects involved in this detection process is the recognition of the relevant elements of a scene, which are normally related to certain features, structures or patterns. The combination of these characteristics determines the regions that stand out from the image. Hence, our eyes are the interface between our brain and the information from our environment and for this reason it is vital to know where they are looking.

The salient sections of a scene generate a focus point to our attention which results in our eyes being systematically fixated on them. For this reason, it is vital to obtain relevant attention data from our own visual system behaviour.

The method to obtain the data where our eyes are fixated on, is the use of eye or gaze trackers. In this process, the subjects are exposed to a series of images and the eye-tracking devices record the exact point of the images where their eyes tend to fixate. As a result, certain probability is assigned to each pixel corresponding to the human eye inclination to look at it. Thereby, after normalizing the pixel probabilities, we are able to represent them using a range where zero is depicted as black and one as white. This process generates what are called saliency maps.

1.1 Motivation

The intention of this Bachelor Thesis is to go one step further in terms of detecting saliency in images and scenes. During the last years, saliency detection in two-dimensional images has been widely studied by the computer vision community. Therefore, there already exist reliable saliency datasets which have been carefully gathered from subjects observing common images.

Nowadays, the prompt development of technology is bringing new devices, and thus, new media interfaces with it. This is the case of Virtual Reality (VR), which allows its users to simulate a physical presence in a virtual environment. In this sense, VR is providing us with 360° scenes and experiences which resemble the environment around us. For this reason, it might be relevant to study the human attention in VR environments.

However, we only visualize and process the information from a small region of or environment (Figure 4.1) which corresponds to where we are looking at, our field of view (FoV). Thus, as we aim to recreate the human's brain cognitive process of determining the relevant or salient regions of a scene, the second part of this thesis will address the idea of computing saliency maps from FoV images extracted from 360° immersive videos. Furthermore, in order to make it as similar as possible to our visual attention system, our objective will be to do it in real time just like the human brain does. This approach might be useful for fields that require both real-time and computer vision features such as teleoperation, telepresence or autonomous driving.

1.2 Objectives

The main goal of this work is to study the human visual system in virtual environments and aim to recreate the detection of relevant regions on them using deep learning techniques.

Bearing in mind the ideas mentioned in the previous section the specific objectives of this thesis are synthesized as follows:

- Gather human visual attention data for a novel dataset which includes eyes and head positional data in 360° immersive videos.
- Build a deep learning saliency prediction model which is capable of generating saliency maps in a real-time basis.

1.3 Document Structure

Chapter 1 introduces the theoretical concepts that will be used in the thesis along with the motivation and the goals of this study.

In Chapter 2, the background regarding visual saliency and attention is provided, along with the state-of-the-art of deep learning techniques in saliency prediction. Additionally, we provide an overview of the most used saliency datasets, benchmarks and metrics.

The core of the practical part of this thesis is comprised between the third and the fourth chapters. Firstly, in Chapter 3 the parameters and methodology for the dataset collection are explained. In Chapter 4 the architecture and the training parameters of our deep learning model for saliency prediction are described.

To finalize the document, in Chapter 5 we evaluate the results obtained from the developed model in the previous chapter. Finally, an overview of the contributions done in this thesis, as well as some conclusions which lead to other questions and future work, are briefly described in Chapter 6.

Therefore in Chapter 3 we proceed to gather the head and eyes positions of fifty subjects exposing them to 360° VR immersive videos. This way we contribute in the creation of a novel dataset based on the people related to the Chair of Media Technology of the Technical University of Munich which will be extended this year to the Stanford University.

Chapter 2

State of the Art and Related Work

2.1 Visual Attention and Saliency

Our physical environment provides us a vast source of sensory information, which is that large that not even the human brain is able to process simultaneously. For this reason, and in order to quickly react and respond to sudden changes in the environment, one of our cognitive processes should be in charge of selecting the most relevant stimuli from the physical world and filtering the less relevant information. This process of information selection and filtering is referred to as attention. In literature, the terms visual attention, gaze and saliency are often confused by readers, thus it is crucial to start by differentiating these three terms:

- **Attention:** is the general concept that includes all the factors that influence the human visual selection mechanisms. These factors can be divided in Bottom-Up (scene-driven) and Top-Down (expectation-driven) and will be explained in more detail in Subsection 2.1.1.
- **Saliency:** is the characterization of specific parts (regions or objects) of the images or scenes, that from an observer point of view stand out compared to their neighbour ones. It is a concept related to bottom-up approaches [LIN98] as it will be shown later.
- **Gaze:** is the long time visual fixation, from which depend the attention, action and the engage of vision [HB05].

Moreover, the human visual system has the particular ability of rapidly choose the most visually relevant regions in an individual's field of view. This action is classified as one of our cognitive processes, which are a number of tasks that the human brain is continuously performing. These procedures are in charge of processing all the external information that we receive from the environment and thanks to them, cognition exists and allows us to

explore the world in real time. If we translate this concept to machines, we encounter the field of computer vision which seeks to automate the tasks of the human visual system. For this reason, the main objective of this project is to develop a deep learning model which is capable of generating saliency maps in real time.

It is necessary to provide the reader with some basic definitions related with saliency that will be helpful for the correct reading of the rest of the document

- **Saliency model:** predicts an eye fixation probability density $p(x, y \mid I)$ for a given image I.
- **Saliency metric:** is a performance measure for a saliency map on ground truth data.
- **Saliency map:** is a metric-specific prediction derived from the model density [MKB17].

Additionally, it is important to make a classification of the visual attention models. In the following subsections, this division will be widely explained according to [BI13].

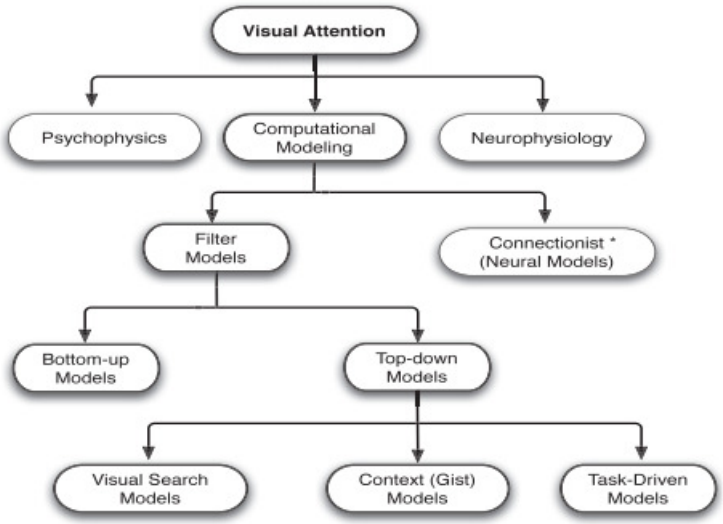

Figure 2.1: Taxonomy of visual attention studies [BI13].

2.1.1 Bottom-Up versus Top-Down Approaches

The concept of attention has been widely categorized in the related studies, however the two most common categories are: Bottom-Up and Top-Down approaches.

- **Bottom-Up (exogenous) Attention:** is stimulus driven. In other words, corresponds to an externally induced process in which the information is selected because

of the highly noticeable features of the stimuli. For this reason, it is reflexive and fast.

- **Top-Down (endogenous) Attention:** is goal driven. It corresponds to an internally induced process in which the information is actively searched in the environment following voluntarily chosen factors, in other words, it is task dependant. Thus, top-down attention is slow and deliberative with variable selection criteria depending on the task.

Moreover, in the field of psychophysics and neuroscience, attention has been extensively studied and has explained that in bottom-up approaches the target stimuli stand out if they differ from their background in terms of features such as color or orientation [JM10]. However, in top-down approaches the target is defined by preconceived factors, and consequently it is identified after an intentional examination of the elements in the field of view (FoV) stimulus by stimulus [WH04]. In spite of this differentiation, bottom-up and top-down factors continually influence each other to orient attention. [JM10] Hence, the bottom-up deployment of attention towards salient locations can be modulated and sometimes superseded by top-down, user-driven factors. [DJ95]

In previous neurophysiological studies of attention, the kind of stimuli and tasks have differed, they could be static (photographs or cartoons) or dynamic stimuli such as videos, movies or interactive videogames. These stimuli have been extensively exploited in previous studies over three types of tasks: free-viewing, visual search and interactive tasks. These types of tasks will be determinant for the selection of the visual stimuli of the LMT Dataset in Chapter 3.

Therefore, and as our goal is to determine the most relevant or salient regions of images (or video frames), it is necessary to specify the unit of the attention. Space-based theories claim that humans consciously attend to locations in the space where a target may appear. This kind of theories are directly related with some fixation bias such as center-bias [TJT09], which determines that humans tend to fixate more in the center of the images. Other kind of approaches base the visual attention essentially in recognized objects giving more importance to higher level features [ESP08]. In contrast, the third possible unit relies on low-level image features that determine the saliency of the scene. Furthermore, and basing their approaches in the above-mentioned units, several state-of-the-art saliency modelling techniques have risen and will be briefly described in the Section 2.2.

2.1.2 Spatial versus Spatio-Temporal Approaches

Visual attention is also context dependent, if a pattern or object repeatedly appears in a set of images they are more quickly recognized. In the real world we constantly experience changes in the visual information due to the natural dynamics of our environment and also due to our own movement. In other words, as we move our field of view (FoV) changes but as we don't live in an static environment, the natural movements in our surroundings

also affect the piece of world that we see and the salient regions inside of it. Therefore, a good saliency detection model should be able to capture regions that are spatial-temporally relevant.

This kind of division is particularly relevant, as one of the goals of this thesis is to be able to determine the saliency in 360° immersive videos. To clarify, there is an important distinction between 360° immersive scenes and 360° videos, as the first ones correspond to static environments while the second ones are dynamic stimuli.

As we have previously discussed, the most of the attention models include a spatial component. Following the same previous division, it is possible to distinguish between two types of temporal attention models:

1. The ones that use the motion channel to capture the fixation, based in the moving stimuli in the images, which clearly corresponds to a bottom-up approach.
2. From a top-down approach some saliency models have as a purpose to find the spatio-temporal aspects of a specific task. Following this idea, in related studies [RS] they study of timing and motion order for humans attending to sequential target stimuli. It is also interesting to mention the approaches that generate static and dynamic saliency maps for dynamic and interacting scenarios like video-games or movies. In this case they limit the temporal information only to the stimulus level and they gather and study the dynamic focus of attention during the task or game [JLG10, SMP07].

Generally, spatio-temporal models take into consideration, not only the information and prediction of the current scene, but also the knowledge extracted from previous frames. For this reason, this kind of models are used in tasks such as video summarizing [SMP07], video cuts placing automation or video thumbnail estimation. This tasks are also addressed in VR scenarios [VS], as it will be pointed out in the next subsection.

Spatio-temporal VR Saliency Prediction

The goal of this subsection is to denote the importance of saliency prediction in VR as well as the impact of spatio-temporal models in this field. This task will be carried out by mentioning, and briefly describing, the main applications of VR saliency.

Some of the most used techniques and disciplines which benefit from saliency in the VR field are:

- **Panorama Video Synopsis:** The task of automatically summarizing videos using saliency has been also applied to VR content. This kind of techniques consist in computing the saliency from the first video frame extracting a first FoV, and the FoVs of the subsequent frames are computed as the most salient FoV which is closest to the center of the one from the previous frame. Some approaches even generate an

FoV camera path that guides the user through the most relevant parts of the scene [YSG16].

- **Panorama Thumbnails:** This technique seeks to extract a representative view-port of a VR scene, which is really relevant, as the panoramas cover the full sphere. It is vital as well for a 2D preview or thumbnail of a VR scene or video, as before wearing an HMD the user can have a better idea of what he will see during the experience. [VS]
- **Automatic alignment of cuts in VR video:** Regarding this kind of application, in [VS] they perform a further study on the investigation of [ASM17], using predicted saliency maps to align the video cuts. Sitzman et al., based their approach in the maximization of the Pearson´s Correlation Coefficient (CC) of the saliency maps.
- **Saliency-aware VR image compression:** VR formats are more demanding in terms of bandwidth and storage than traditional images or videos. For this reason, it is a relevant task to optimize this kind format due to importance of low-latency in VR scenarios. The idea basically consists in computing the saliency maps of the original equirectangular images and keep the highest resolution in the most salient regions of the scene.

2.2 Deep Learning for Saliency Prediction

The purpose of this section is to provide the reader with the theoretical background behind the basic principles of the thesis for a better comprehension, specifically about the concepts of deep learning and convolutional neural networks (CNN).

As result of the technological advances from the last years, the field of artificial intelligence (AI) has become really popular within and out the scientific community. This rise on its popularity has made it easy to forget that the AI field has been around for decades. During this time, its popularity and the confidence in this kind of technology has experienced several fluctuations [his]. One of the main problems in the past of the field, was related with the low computational capacity of GPUs which made the training of the models, with large amount of data, an arduous task. However, today the field has become more accessible, allowing one of its disciplines, machine learning (ML), to be applied to almost every aspect in science.

With the purpose of a better comprehension from the reader, it is important to differentiate the terms of artificial intelligence, machine learning and deep learning (Figure 2.2). The field of AI is, in essence, when machines (robots or computers) are able to perform tasks that typically require human intelligence. Machine learning is the application of AI that provides systems the capacity to automatically learn and improve their from experience, similarly to the way human beings learn. The fundamental component of machine learning are

artificial neural networks (ANN or simply NN), which consist in computational algorithms inspired by the functioning of the neurons in the human brain, that are capable to learn from large amounts of data. On the other hand, deep learning is a branch of ML which uses algorithms with multiple layers to progressively extract higher level features from the raw input. The main idea is to implement multiple layers of features so the model will increasingly learn certain features with a high abstraction level. Furthermore, each of these high-level features are defined in terms of their relationship with simpler features. In other words, the algorithms learn complicated concepts from simpler ones, creating a hierarchical structure that creates large concatenations (*deep*) of features, which provide the name to these kind of techniques. The building blocks used for that purpose are Convolutional Neural Networks (CNN), which will described in the first subsection.

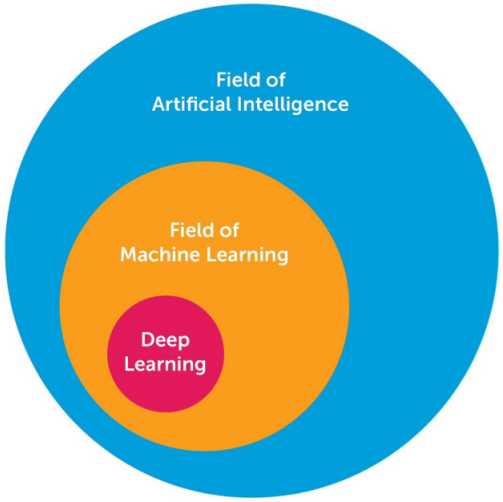

Figure 2.2: Representation of the scope of AI, ML and DL.

So, in essence, the characteristics that differentiate deep learning from other models are as follows:

- Large number of neurons.
- Complex ways of connecting layers.
- Large computational power needed to train.
- Automatic feature extraction.

The late development in the field and state of the art has eased the appearance of several applications in the field of deep learning, such as self-driving cars, automatic translation, automatic text generation, health-care applications like in [KKG17], etc. Furthermore, in the field of our interest, which is computer vision, some examples are face recognition like DeepFace [YTW14], video scene classification and object tracking [NH16]. However, the

purpose of the thesis will focus our research towards saliency prediction using a pre-trained model for an image classification task [JDF09].

2.2.1 Convolutional Neural Networks

Convolutional Neural Networks (CNN or ConvNets) are the hierarchical structures used in deep learning for the most of the image related problems from computer vision, as they are inspired by the structure of the human visual system. They are arguably the most popular architecture of deep learning due to their effectiveness. However, its usage is not restricted only to images, and they are also applied in other areas such as recommender systems or natural language processing.

The interest in CNN has gone through a renaissance phase with the appearance of GPGPUs (General-purpose computing on graphics processing units). It started with AlexNet in 2012 and it has grown exponentially ever since, up to the poiny where CNN have outstanded humans in some tasks such as image classification [JDF09].

As we have seen, we are dealing with a very powerful and efficient model, which will be synthesized and explained in the following subsections. We will start with the building blocks of the CNN (neurons and layers), followed by the description of the architecture of some of the most used models and finally the training or learning process of these architectures will be addressed.

2.2.2 Architecture

Neurons

It is really common to explain the concept of neuron with the biological analogy of the neurons from the human brain. In fact, as we have previously pointed out, neural networks have been inspired and tried to recreate the human brain functioning. The same way as neurons are the basic computational unit of the brain, neurons are the constituent blocks of neural networks. In Figure 2.3 we can observe the structural and functional similitude between the biological and the mathematical model. Biologically, a neuron consists of a neuronal cell body, where the input to the node represents the dendrites and the outputs represent the axon.

In neural networks, the inputs of the nodes are weighted by the so-called process back-propagation, and the output is usually a function of the weighted sum of the inputs. In other words, every input is multiplied by certain parameters (weights) and connected to the neuron's body. After the pertinent computations in the body, and in order to set a threshold for the output, a non-linearity is applied to the result; this receives the name of activation function.

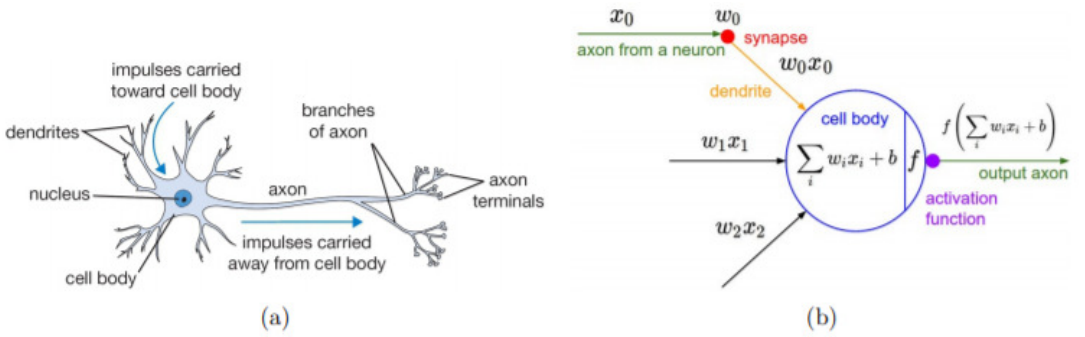

Figure 2.3: Representation of a biological neuron (a) and its mathematical model (b).

Layers

A layer consists in the aggregation of several neurons or nodes. A neural network is constructed by three types of layers:

1. **Input Layer:** initial data for the NN.
2. **Hidden Layers:** intermediate layer between input and output layer and place where the computations are done.
3. **Output Layer:** produce the result for the given inputs.

Each node is connected with each node from the following layer (fully-connected layers), and each of the connections has a particular weight. As we have already described in the previous subsection, weights can be understood as the impact that a node has on the node from the next layer.

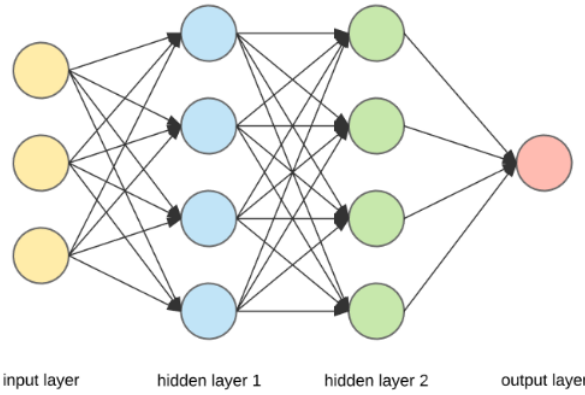

Figure 2.4: Architecture of a Neural Network.

The characteristic that distinguishes deep learning networks from the common single-hidden-layer neural networks is their depth; this means, the number of layers through

which the data must go through in the process of the pattern recognition. For a NN to be considered *deep* it must have more than three layers, including the input and output. Furthermore, when training the network, each layer trains on a different set of features, which are based on the previous layer's output. Thus, the *deeper* you step into a NN, the more complex the features that its nodes can recognize, as they aggregate and recombine features from the previous layers. This phenomenon is known as feature hierarchy, and it is a hierarchy of increasing complexity and abstraction. Moreover, this is the reason that makes deep NN capable of handling datasets of considerable magnitude with billions of parameters.

However, what makes these NN unique, is their capability of discovering patterns, similarities and anomalies in unlabeled and unstructured data, fact which is completely out of the reach of human beings.

There are several types of layers, not only classified by their position in the NN but by their behaviour:

1. **Fully Connected Layers:** is the kind of layer explained before and shown in Figure 2.4. In this case, all the neurons in the layer receive all the values from the neuron of the previous layer or from the input data. It is straight forward that they need a large number of connections, and thus, a considerable amount of memory is required.
2. **Convolutional Layers:** they provide the name to the CNN, because they are their main building blocks. This kind of layer performs a dot product between two matrices, where one matrix is the set of learnable parameters known as kernel or filter, and the other matrix is a portion of the image. The kernel is spatially smaller than an image, but is more in-depth. This means that, if the image is composed of three channels (RGB), the kernel height and width will be spatially small, but its depth extends up to all three channels. During the forward pass, the kernel slides across the height and width of the image producing the image representation of that receptive region. This produces a two-dimensional representation of the image known as an activation map that gives the response of the kernel at each spatial position of the image.

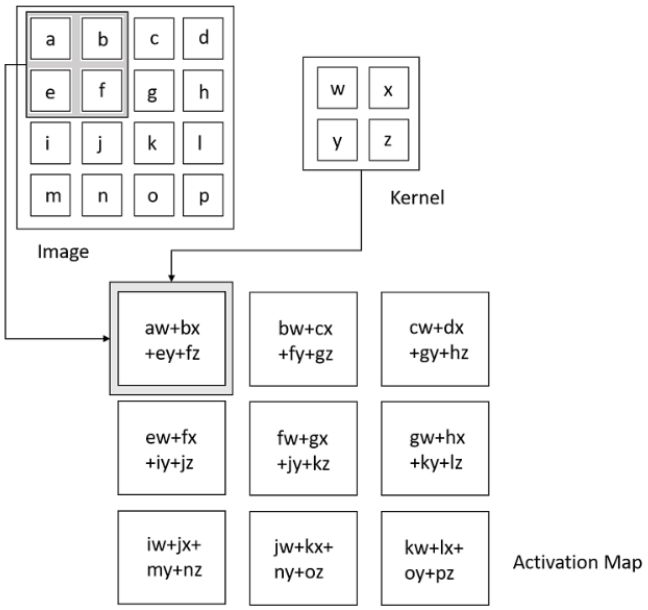

Figure 2.5: Representation of a Convolutional Layer operation.

3. **Transposed Convolutional or Deconvolution Layers:** the transposed convolution operation basically creates the same connectivity as the normal convolution but in the backward direction. Note that despite its name, this does not mean that we take an existing convolution matrix to and use its transposed. It is used instead of the up-sampling layer, in order to increase the size of the feature maps. The difference between both, is that up-sampling use a specific interpolation method for the size expansion, while transposed convolution has learnable parameters. Thus, its functioning can be understood as going backwards of the convolution operation.

4. **Max-pooling or pooling Layers:** the function of the pooling layers is to reduce the size of the image representations. Their operation consists in the selection a portion or window of the image representation's pixels and extract the highest value among the selected pixels. Thus, as it can be appreciated in Figure 2.6, the most significant (highest values) of the activation map are extracted producing a smaller output. Furthermore, pooling layers provide translation invariance which means that a pattern or object could be recognized independently of its position on the frame.

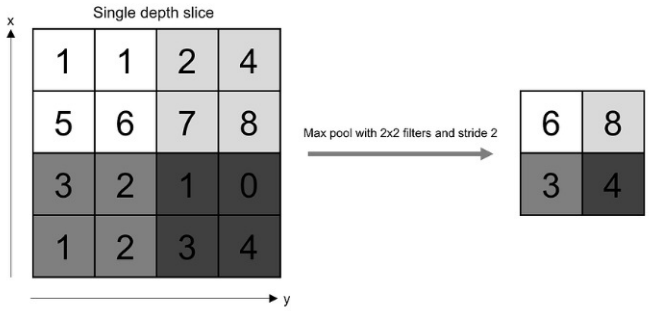

Figure 2.6: Representation of a Pooling Layer operation.

5. **Batch Normalization Layers:** the purpose of this kind of layers is to increase the stability and speed of a neural network by normalizing the output of a previous layer. This is achieved 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.

6. **Dropout Layers:** the objective of this layers is to prevent over-fitting when training a model. Over-fitting happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data. Hence, Dropout works by randomly setting the outgoing edges of hidden units (neurons that make up hidden layers) to 0 at each update of the training phase.

VGG architecture

In this subsection we will briefly describe the VGG architecture because in Chapter 3, its version of 16 layers will be an important component of our MIX-Net approach for saliency prediction.

The VGG network architecture was first introduced in 2014 in [SZ14b] for classification tasks. The main characteristic of this network is its simplicity, making use of 3×3 convolutional layers which are placed on top of each other resulting in a large *depth*. Also, the architecture includes pooling layers after the convolutions for reducing the volume size. After five blocks of convolutions followed by pooling layers, the VGG architecture includes two fully connected layers that are succeeded by a soft-max classifier for the image classification purposes.

The VGG architecture has two variants: VGG16 and VGG19, where 16 and 19 correspond to the number of weight layers in the network. Both structures can be observed in columns D and E of Figure 2.7 [Ros].

In the last years, the VGG architecture was considered *very deep*. However in the last years some models like ResNet [HZRS15] have *deeper* structures.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 2.7: Layer architecture of the VGG networks. Column D corresponds to the VGG16 and E to the VGG19.

However, the VGG networks have two handicaps: they require long times for training and its weights are large in terms of storage. For this reason, smaller versions were developed resulting in the columns A to C from Figure 1.6. This is one of the main reasons, where in our approach explained in Section 3.3 we transfer the learning of the pre-trained weights of the VGG16 to our saliency prediction network and thus, there is no need to re-train it.

In Figure 2.8 a representation of the architecture of the VGG its shown in its 16 layers version, with the objective of providing a better understanding of the model as it will be used for our MIX-Net in Chapter 3.

2.2.3 Training and Testing

Once a model is defined, it is time to train the NN with data so that it learns how to predict an accurate output for a given input. This task is performed making use of datasets, concept that will be described focusing on saliency datasets in Section 2.3.

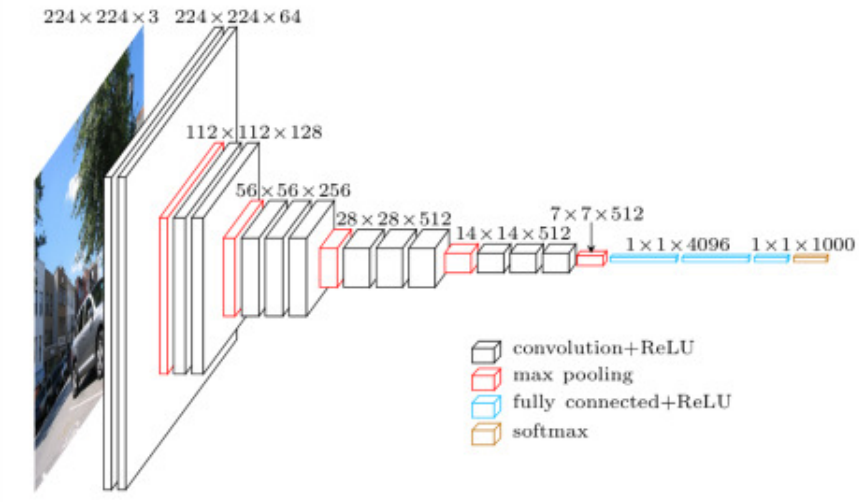

Figure 2.8: Representation of the VGG16 network architecture.

The process of training is performed by providing with the sample data from the datasets consisting in two phases: training and validation, which are performed using different homonymous datasets. Once a model is trained, the testing stage is performed with the objective of evaluating the model. We will define these three phases through the different datasets used in each stage.

- **Training Dataset:** Corresponds to the used data to train the model. The model *sees* and *learns* from this data. In our case, our data is composed of images and their respective fixation maps considered as ground truth for the training.
- **Validation Dataset:** Consists in the sample of data used to provide an unbiased evaluation of a model fit on the training data while tuning model hyperparameters. Thus, the model occasionally *sees* this data, but it never *learns* from it. It is often divided in two parts: in the first one, the model selects the best performing approach (hyperparameters) using the validation data. After that, the accuracy and/or the loss of the selected approach is computed. In the case of classifiers, normally both accuracy and loss are computed, while in our particular case only loss will be computed making use of a loss function as we are implementing a regression task. The loss is computed using a loss or cost function, which is the function which will be minimized during the training process. It is basically a measure of how good a prediction model is in terms of being able to predict the expected outcome. More information about loss functions will be provided in subsection 4.3.3.
- **Testing Dataset:** Is the sample of data used to provide an unbiased evaluation of a final model fit on the training dataset. It is used only once a model is completely trained. Normally, the test set contains carefully sampled data which represents the most of the possibilities in the real world.

In our particular case, we will make use of the SALICON dataset which consists in 10.000 training images (and their corresponding attention maps) and 5.000 in the case of validation and testing sets. Hence, our split would consists in the 50% of the amount of training data for each validation and testing data.

2.2.4 Image Classification

In this subsection we will briefly describe the concept of image classification as some of its features will be used in the design of our deep learning model.

Image classification corresponds to a computer vision process that is able to classify a given image according to its visual content. For instance, an image classification algorithm might be used for determining if an image contains a person or not, or if an image corresponds to a dog or a cat. The task of of detecting an object in an image or scene is a trivial task for the human brain, however it is still a challenge for the computer vision field.

In the recent years, several models, datasets and image classification competitions have risen with competitive results. One of the most famous is the *Dogs vs. Cats Challenge* by Kaggle [Kag] where the task was to outperform the 82.7% of accuracy from [Gol08].

Since many deep learning models have approached the issue of classifying images. For instance, the VGG16 and VGG19 [SZ14b], ResNet50 [HZRS15] or Inception V3 [SLJ+15] models were trained with the popular ImageNet [JDF09] dataset which labels and categorizes images into almost 22.000 object classes for computer vision purposes. The above mentioned models are then trained on $\sim 1.2 million training images with another 50,000 images for validation$

In our model described in Section 3.3, we make use of the VGG16 architecture and weights trained on the ImageNet dataset in order to take advantage of the already learnt object structures. Thereby in our CNN we make use of a model trained for an image classification task for the purpose of predicting saliency maps.

2.3 Saliency Datasets

A fundamental part in the field of machine learning are datasets. The selection of a dataset for training a model, will determine the way a model will learn.

A dataset is basically a collection of data, but this data will be determined by the objective of the task. For instance, the ImageNet dataset [JDF09] is an image database organized according to the WordNet [GAMM93] hierarchy (currently only the nouns), which is used in several tasks but mainly in image classification, providing an average of 1000 images for each word or *synsnet* acting as the image labels.

During the last years, several eye-tracking datasets have been created and shared in the saliency community with the goals of understanding visual attention and building computational saliency models. Saliency datasets, basically include natural images which are the visual stimuli and the corresponding eye movement data recorded using eye-tracking devices as attention maps. A common saliency dataset contains hundreds or thousands of images, which have been viewed by tens of subjects while the location of their gaze in image coordinates is tracked over time.

As one of our objectives its to construct a novel 360° Virtual Reality Videos Dataset in the following subsections we will differentiate between 2D Images datasets and VR Images datasets.

2.3.1 2D Images Datasets

Dozens of new models on fixation prediction, based on deep learning [VS], are published every year and compared on open benchmarks such as MIT300 [BJB+] and LSUN [Zhab]. Usually, every model is trained on one or more saliency datasets, obtaining different weights that in testing can be loaded in order to have not only one, but several possibilities for saliency maps prediction.

The most common datasets are only recorded with the task of "free-viewing" the images [TJT09, MCK09a, BT09, SRC10a]. However, most datasets have their own distinguishing features in the images. For instance, the NUSEF dataset [SRC10b] contains semantically affective objects and scenes, while the FIFA dataset [MCK09b] focuses on faces. Other datasets such as the MIT1003 [TJT09] or SALICON [MJZ15] include more general images, causing an increase of their size, i.e. 1003 images in the first one and 10.000 in the latter one. Recent datasets are focused in specific domains in the field of saliency. For example, EyeCrowd [MJZ14] focuses on saliency in crowd, FiWI [SZ14a] in web page saliency and the MIT Low Resolution dataset [TJT11] in saliency in low resolution images.

2.3.2 Virtual Reality 360° Images Datasets

It is indispensable to make the distinction between saliency in 2D and in VR. It is straightforward that there exist inherent differences between the user experience with VR and the television or PC, hence, visual attention and saliency will also vary between both scenarios.

The desktop viewing conditions are much different from the ones wearing a head mounted display (HMD). In [VS] Vincentr Sitzmann et al. performed an exhaustive study on how people explore virtual scenes. In their work they collected and evaluated gaze data from 122 participants, observing 22 virtual static scenes under three different conditions: the VR condition, which correspond to a standing position using an HMD; the VR seated condition, with the participants seated in an office chair and also in desktop condition. They used

the Oculus DK2 HMD with the addition of the Pupil-Labs eye-tracker, recording at 120Hz for the VR conditions.

After the data collection, they evaluated the results in order to understand the human viewing behaviour in VR. Several conclusions where extracted:

1. **The viewing behaviour is still similar between users** like in the previous studies in desktop conditions [MB13]. Actually, the 70% of all the fixations fell within the 20% more salient regions.

2. **The existence of an equator bias.** As we have previously pointed out, in [TJT09] Judd et al. demonstrated the existence of a strong bias for fixations to be near the center, the so-called center bias. However, in VR, the results showed that the human fixations were concentrated in the latitudes near the equator of the panoramas [VS]. In the study it is noted that their selected stimuli have a clear horizon line and that may have influenced the bias. However, it was later shown that even for images with distributed content along all the latitudes, very few fixations were detected near the poles.

   It is important to point out that this bias was detected during free-viewing tasks on static images and the bias may be vary in other viewing conditions such as gaming.

3. **Different starting points don't influence the final saliency map.** By making use of the Pearson'a Correlation Coefficient (CC), which will be explained in more detail in the next subsection, they obtained a median score of 0.79 which indicates that after 30 seconds, the saliency maps from users starting from different viewports converged and presented a high similarity. In other words, if two persons start their viewing in different positions of the 360° sphere they will end up looking at the same salient parts of the scene.

4. **Relationships between head and gaze movements** First, they showed that the average time for the users to have fully explored the scene was around 19 seconds. They also identified the vestibulo-ocular reflex [LR86] in their data. basically, this reflex consists in the gaze movement in the opposite direction of the head movement, stabilizing the line of sight and improving the quality. Thirdly, they computed the time offset between the gaze and the head, determining an average delay of 58 milliseconds of the head orientation to follow the gaze coordinates. Furthermore, and basing this part of their study in [SG00] where it was shown that gaze speed is different when users fixate and when they do not, they classifies viewing behaviour in two modes: attention and re-orientation. Meaning by attention the situation where users maintain their gaze in a specific salient part of the image, and the movement to new salient regions correspond to re-orientation mode.

2.3.3 Benchmarks and Metrics

A considerable amount of new computational saliency models are published every year. This models, in order to establish a ranking in terms of accuracy, are compared on open benchmarks such as MIT300 and LSUN [BJB+, Zhab]. However, to objectively determine which model provides the best approximation to human eyes fixation is difficult to judge because models are compared using a variety of several metrics. In previous studies, it has been shown that there's no saliency model which performs properly under all the types of metrics. For example, in Figure 2.9 we show the inconsistency in how different metrics rank saliency models. As we can observe for the input image in Figure 2.9.b, the output of 8 different saliency models is shown and, when they are compared to the human fixations ground truth they obtain different scores conforming with different evaluation metrics.

SIM ↑	0.29	0.4	0.49	0.2	0.42	0.41	0.39	0.42
CC ↑	0.04	0.31	0.46	0.14	0.47	0.3	0.25	0.34
NSS ↑	0.13	0.79	1.17	0.43	1.18	0.8	0.68	0.97
AUC ↑	0.52	0.77	0.79	0.59	0.81	0.71	0.68	0.74
sAUC ↑	0.52	0.66	0.53	0.53	0.6	0.64	0.63	0.63
IG ↑	-2.02	-0.03	0.06	-16.71	0.19	-0.39	-0.02	-1.32
KL ↓	2.66	1.30	1.25	13.09	1.12	1.56	1.30	2.33
EMD↓	5.77	4.73	3.35	5.17	4.45	4.63	4.99	4.49

Figure 2.9: Comparison of different saliency metrics which result in different scores [BJO+16].

First of all, we define saliency metrics as functions that take two inputs, ground truth and saliency prediction, that represent eye fixations and provide a number as the output that evaluates the similarity or dissimilarity between them. In this section, the five most commonly used metrics in other evaluations and reported in the MIT Saliency Benchmark [BJB+] will be described. Some of these metrics have been specifically designed for saliency evaluation (sAUC, NNS) but others are adaptations from other fields such as information theory (KL), statistics (CC) or signal detection (variants of AUC). Thus, each metric treats its inputs in a different manner, as random variables, probability distributions, histograms, etc. Hence, for a better understanding of the metrics we will classify them as location-based or distribution-based following [NRD13].

Table 2.1: Saliency Metrics Classification

Metrics	Location-Based	Distribution-Based
Similarity	AUC, NSS	SIM, CC
Dissimilarity		KL

Location-Based Metrics

A) Area under ROC Curve (AUC): AUC is the most common metric for evaluating saliency maps. It is a similarity metric, based in the interpretation of the saliency maps as classifiers of pixels, judging which pixels are fixated or not. In signal detection theory the Receiver Operating Characteristic (ROC) is a metric of the trade-off between true and false positives using several discriminator thresholds. Thus, the area under this ROC is referred as AUC, and based in the saliency maps as binary classifiers of fixations, the ROC curve is determined by the measurements of the true and false positive rates of each binary classifier.

The most employed variants of AUC are the so-called AUC-Judd [BJB$^+$] from Judd et al. [TJT09] and the AUC-Borji by Borji et al. [ABI12] Additionally, it has been shown the false positive computation of AUC-Borji is a discrete approximation of AUC-Judd.

This metrics have the drawback of penalizing the models that implement a center bias [Tat07], which corresponds to the natural tendency of the distribution of fixations to be concentrated near the center of the images. This occurs due to the fact that these types of models will account a least part of the fixations (only the ones in the center) on any image besides its content.

B) Normalized Sanpath Saliency (NSS): NSS is also a similarity metric and it is defined as the average saliency value of fixated pixels in the normalized (zero mean and unit as variance) saliency maps. As the mean value is subtracted during the computation, NSS is invariant to linear transformations.

Distribution-Based Metrics

C) Similarity (SIM): This similarity metric, also called histogram intersection, measures the similarity between two different saliency maps when viewed as probability distributions ($SIM = 1$ means the distributions are identical). It was first introduced as a metric for color-based and content-based images matching, but it has gained usage and recognition as a simple comparison of saliency maps. Given a saliency map and a continuous fixation map, it is computed as the sum of the minimum values at each pixel after normalizing the maps.

D) Pearson´s Correlation Coefficient (CC): This metric corresponds to a statistical

method, widely used in science to measure the correlation of two variables. It is also known as linear correlation coefficient. In this case both maps, saliency map and continuous fixation map, are interpreted as random variables and the result corresponds to the linear relationship between them.

E) Kullback-Leibler divergence (KL): This method is an extensively used measure in information theory, which determines the difference between two probability distributions. It is a non-symmetric measure of the information lost when the saliency map is used to estimate the fixation map. Its formula and a more careful explanation of this metric is provided in subsection 4.3.3 as it is used as the loss function for training our model.

Chapter 3

LMT Dataset Creation

The first part of the performed work and contributions to the field in this thesis correspond to the contribution to the creation of the LMT Dataset. LMT stands for the German translation of Chair of Media Technology from the Technical University of Munich. This dataset consists in the annotation of the fixations of human head and gaze position for 360° short VR videos in a seated condition.

3.1 Motivation for Creating a New Dataset

In this last years, multimedia users have been exposed to new interactive experiences with VR and 360° content, in which they have more freedom to virtually explore the observed scenes. As we have previously mentioned, this change of visual interfaces and content has also an impact on how humans observe what it is shown to them.

Users are no longer passive observers, but they now can actively decide where to look at or how to move within the scene. Both head and gaze movement are fundamental in this exploration process. Thus, the availability of public datasets that contain the visual stimuli and the corresponding head and eye-tracking information is crucial in order to how humans observe and explore 360° content.

In Section 2.3 we have presented some of the state-of-the-art saliency datasets. Once the reader has obtained a better knowledge about them we can point out some facts have motivated us for creating our own new VR dataset.

The principal reason relies on the fact that attention and saliency have been widely explored in 2D images but when coming to VR the amount of available data decreases considerably. Regarding the gaze tracking, previous algorithms were developed with monitor-based experiments (not HMD-based) and visual attention in 2D differs from the immersive experience of VR. Thus, we decided that we would use a VR HMD for showing the stimuli

to the participants. The selection of the HMD and the video stimuli is further described in the following section.

Secondly, the most of the available VR datasets are based on 360° Images and not in 360° immersive videos. Only recent approaches like the dataset collection performed by Erwan J. Daviv et al. in [ED18] include the head and eye-tracking data for dynamic VR scenes. Hence, we aim to provide additional data that might be used in further research where VR dynamic content is present.

Additionally, and one of the facts that differentiates our dataset from the one in [ED18] is that besides the videos and tracking information from a free-viewing task, we also provide the data for a task-driven exploration of the scene. This is the case of the last five videos from our stimuli in which subjects were provided with one or more questions before watching some of the previously showed videos, that will have to be answered after the visualisation. Thereby, we provide novel task-biased information about gaze and head movements in dynamic 360° scenes.

Finally, another differentiating fact is that during the collection of our dataset we also included the audio from some of the videos through stereo speakers. This way we provided the subjects a more immersive and stimulating experience in the scene that they were exploring and we include data that could be relevant for the study of audio saliency and the human head and gaze responses to it.

3.2 Dataset Parameters

3.2.1 Equipment

For the head and gaze data collection we made use of a combination of hardware and software. On one hand the hardware, including Head Mounted Display (HMD) for exploring the virtual scenes, a PC, speakers or earphones for playing the audio from the videos; on the other hand the software, that consisted of programs for connecting the PC with the HMD, for playing the video through the HMD and the algorithm for handling the recordings and storing them to the disk. In the following subsections, the selection procedure for all of the previously mentioned requirements will be described.

Hardware

First of all, for gathering the data, we needed a Head Mounted Display (HMD) which, in our particular case, should be able to also gather the gaze tracking data. Additionally, it should be under the budget and, also, to be available at the earliest.

Once the features that our HMD should implement were clear and after researching [JJMJ18], we found several eligible options:

- **Varjo VR-1:** the main feature of this HMD is its high resolution which is presumed to be the *world's only VR device with human-eye resolution* and it also implements a built-in gaze tracker. The main drawback of this HMD was its cost, which is beyond 5.000 USD, and the waiting time until the reception of the device after the purchase.
- **HTC Vive or Oculus Rift + wearable eye tracker:** in this case, both the HTC Vive and the Oculus Rift devices were available in the Chair of Media Technology. However, they both present the inconvenient of not implementing a built-in gaze tracker. For this reason, we could have to opted for an external solution. There are several options such as the Tobii Pro Glasses 2 [Sli] or for example the binocular add-on from Pupil Labs [MKB14] which would provide the missing gaze tracking to the HMD. The main downside from this option was the combination of two different devices and the small documentation available for the add-on from Pupil Labs, all this on top of the waiting period.
- **HTC Vive Pro:** this device had both of our required features, cost and built-in gaze tracker. Nonetheless, the device was not released at the moment of the experiment, and also the mistrust for being such a new device regarding the documentation and APIs for this first version, contributed for discarding this device.
- **FOVE:** in the case of the FOVE HMD it also accomplished our both requirements, as it implements an integrated 120Hz eye tracker with less than 1 degree of accuracy and 100° FoV. The inconvenient was that it was no longer available to the public. Fortunately, we were aware that the Chair of Human-Machine Communication (LMMK) owned one of this devices, so we decided to ask them if they could lend it to us for two months and we received a positive reply from Ioannis Agtzidis, so we decided to use the FOVE HMD for our experimental set-up.

Figure 3.1: FOVE HMD and its head-tracking camera.

For our study, the used PC holds an Intel Core i7-8700K CPU along with a GTX 1080

GPU. We also needed to add more RAM memory, as in the first tests we were getting less samples per frame as expected.

In addiction, our collection also makes use of the video sound as a stimuli for the user. We do this as in previous studios [ACC12] it has been shown that during the video exploration, gaze is impacted by the related soundtrack. They also shown that when the audio is played alongside with the video, the eye positions of the observers are less dispersed and tend to go more often away from the scene center. This is due to the fact that when the user is immersed in the virtual scene the variation of the sound in one of the areas of the 360^{o} environment can modify and urge the user to look at the scene region corresponding to the unexpected sound. For this reason, the perfect scenario would be to play videos with ambisonic or spatial audio alongside with the use of headphones (so the sounds of the experimental room don't disturb the subjects attention). The inconvenient is that on the internet there aren't enough 360^{o} videos with spatial audio available. Also, the fact that the user is already wearing an HMD makes the option of the headphones unfeasible. Thereby, a straight forward solution would be to use earphones instead, but after leveraging this option we discarded it, as we needed to give some instructions to the users during the experiment. Thus, we decided to place two stereo speakers at both sides of the user's chair in order to reproduce the audio from the videos. The fact that the sound is not ambisonic is relevant but not crucial, as it has also an effect on the gaze movements as shown in [ACC12].

Software

For recording the dataset several software tools were needed. First of all, for connecting the FOVE to the computer we used the owners program FOVE VR Software shown in Figure 3.2.a. This program allowed us to calibrate the integrated eye-tracker from the HMD while it creates the tracking logs from where the eye and head motion data are extracted.

Second, for displaying the video stimuli to the participants in the experiment we used SteamVR and its integrated media player shown in Figure 3.2.b. This program allows four kind of layouts, mono for equirectangular images or videos, left-right and top-bottom for stereoscopic media and anaglyph for 3D content. It also provides four kind of displays: desktop, 180°, 360° and fish-eye. In our case the configuration for the most of the time was mono-360°, except for stereoscopic videos which was top-bottom-360°.

In order to extract and gather the eye and head movements data we used the provided tool from [ASD19]. Which main authour, Ioannis Agtzidis from the Chair of Human-Machine Communication (LMMK) kindly provided us, along with the actual FOVE HMD and position camera. This tool is the *FoveDataToArff* which process the tracking data from the HMD and generates an Attribute-Relation File Format (ARFF) file in which appear the tracked gaze and head positions measured at a 120 FPS from the FOVE. My partner in the dataset creation, Xavier Oliva, and me modified part of the code so the output

ARFF file for each video will also contain the head-position data expressed in quaternions, for further use in his research. Note that the gaze trackings are expressed with an offset provided by a calibration frame which will be further explained in Section 3.4.

Thus, in comparison to previously published datasets, our data is high frequency (240 Hz or 120 FPS) and it provides the raw head-positions and the re-calibrated eye tracking recordings.

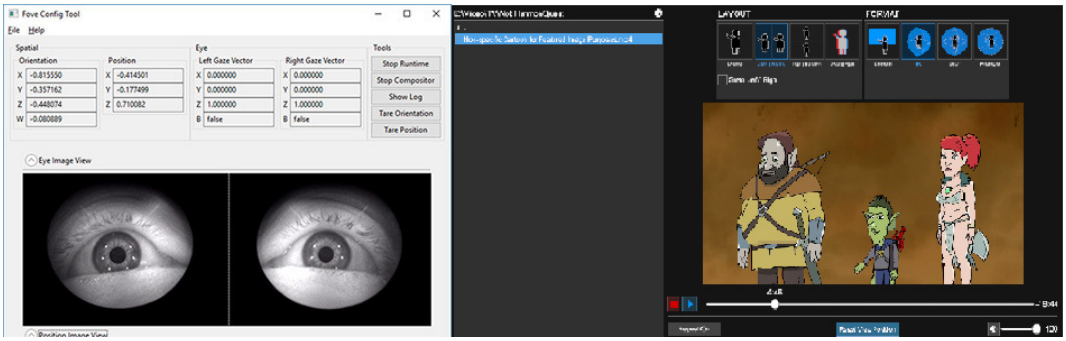

Figure 3.2: Main programs required: (a) FOVE VR Software, used for calibration and interface with the data gathering tool, (b) SteamVR Media Player, used for displaying the 360° videos.

3.2.2 Stimuli

The LMT Dataset is composed of 23 videos, from which five of them are presented again to the subjects for a task-based exploration. All of the videos have been gathered from YouTube at 4K resolution (3840x1920 pixels) and 30 frames per second. The length of the videos is approximately of 60 seconds. Their principal properties are shown in Table 3.1.

The format of the videos also varies, the most of them are in equirectangular or monoscopic format while five of them are stereoscopic. This is done in order to have different information for both formats, as the stereoscopic format provides a feel of *depth* to the human eye, while monoscopic format provides the feeling of being inside of a 360° (dynamic) photograph rather than in a 360° three-dimensional space; this difference can be observed in Figure 3.3.

Following with the technical properties, there are videos where the scene is the same during its whole length and four of them which have scene cuts and where the static location of the camera varies (indicated with the tag "cuts" in Table 3.1). Additionally, we make the differentiation of the videos between static and dynamic (motion). Static videos correspond to the ones where the camera is located in the same place during the whole video, and dynamic videos correspond to the ones where the camera is moving and are indicated with the "Motion" type. The latter ones may cause motion sickness in the users, as there's an incoherence between their static seating position and the moving FoV that they are

observing. These both technical properties are shown in the column type column of Table 3.1. Also, we indicate high-level characteristics of the videos which correspond with their content, for instance if the scene is outdoors or indoors, in an urban or natural environment, if there are people (busy) or not (empty), the type of environment (industrial, hospital, hazardous environment, etc.), or the kind of moving action for the motion videos (car driving, walking, sailing, etc.).

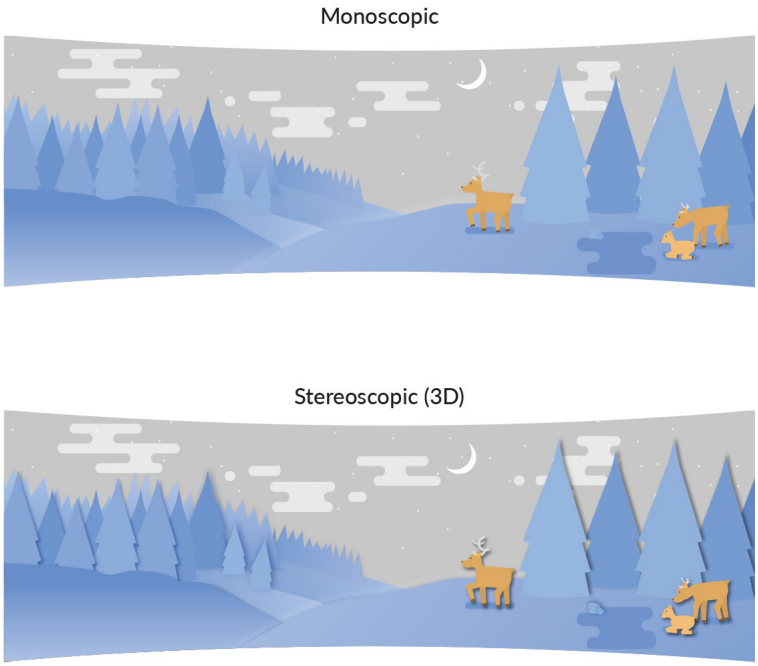

Figure 3.3: Monoscopic versus stereoscopic format.

Task-biased exploration

One of the novel aspects about our dataset, is that we also include data extracted from a task-based viewing task. This means, that in the first twenty-three videos, users are exposed to the already mentioned free-viewing exploration, in which they are able to move freely around the scene and observe it in a bottom-up basis, which leads to a purely stimulus-driven attention. Besides, in our last five videos (from 24 to 28) the experimental team provide the users at the beginning of each video with one or more questions about its content. The videos that were selected for this part, were also part of the first ones, so the user had already visualized them. However, in this case, after watching the video, they would have to give an answer for the question. The questions were basically based on counting objects or people, and thereby influence or bias them in order to keep searching

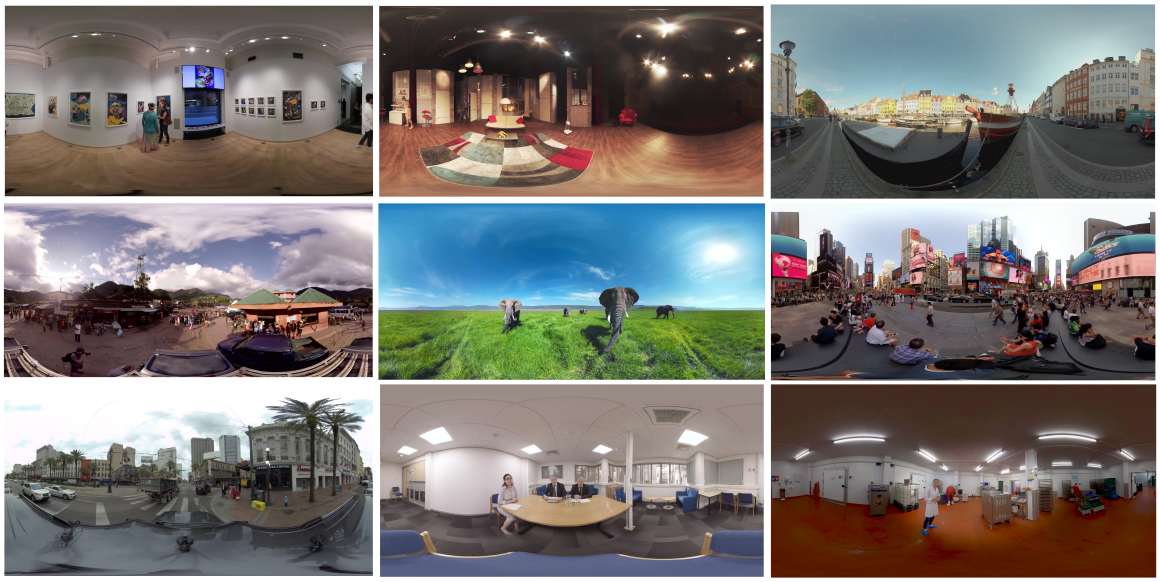

Figure 3.4: Equirectangular frames extracted from some of the video stimuli from Table 2.1. Specifically, frames from the first row correspond to videos 1, 2 and 4; second row to 8, 9 and 13; third row to 14, 16 and 23.

through different parts of the scene which they might have not checked during their free-viewing task.

The questions for the videos 24 to 28 were the following:

- **24 nurse challenge** How many people are there in the scene?
- **25 theatre scene challenge** How many red objects, or objects containing the color red, are there on the stage?
- **26 new orleans drive challenge** How many flags are there in the scene?
- **27 times square challenge** How many screens are there in the scene?
- **28 dance experience challenge** How many hula hoops are there in the scene? How many man and how many woman? How many people is wearing a white shirt?

It is straight forward that by asking these questions the given answer was not really important. The objective was to motivate the subjects to keep exploring the scene even when they had already looked at the most relevant regions.This way, we provide novel eye-position and head-motion data for further studies on task-biased saliency, which clearly corresponds to a top-down approach.

Table 3.1: Video Stimuli.

Video Name	Type	Categories	Length
01_art_gallery	Static	indoors, busy	1:02
02_theatre_scene	Static	indoors, empty	1:02
03_lions	Static, cuts	nature, empty	1:02
04_copenhagen_harbour	Static, cuts	urban, nature, busy	1:02
05_florida_yacht	Motion	sailing, nature, empty	0:45
06_dance_experience	Static, stereoscopic	urban, busy	1:02
07_factory_robots	Static	industrial, indoors, busy	1:00
08_madagascar	Static, cuts	nature, busy	0:53
09_elephants	Static	nature, busy	1:02
10_cruise_festival_skate	Static, cuts, stereoscopic	outdoors, busy	1:02
11_lodhi_garden_india	Static, cuts	nature, empty	1:02
12_gym_workout	Static	indoors, busy	1:02
13_times_square	Static	urban, outdoors, busy	1:02
14_new_orleans_drive	Motion	car driving, urban, outdoors, busy	1:00
15_interrogation	Static, stereoscopic	indoors, busy	1:02
16_interview	Static	indoors, empty	1:02
17_weather_forecast	Static	indoors, empty	1:02
18_bomb_trapped	Static	hazardous environment, indoors, empty	1:00
19_nurse	Motion, stereoscopic	walking, hospital, indoors, busy	1:00
20_car_fix	Static	indoors, empty	1:04
21_operation_room	Static	hospital, indoors, empty	1:04
22_surgical_checklist	Static, stereoscopic	indoors, busy	1:02
23_dog_food_factory	Motion	walking, industrial, indoors, busy	1:00
24_nurse_challenge	Motion, stereoscopic	walking, hospital, indoors, busy	1:00
25_theatre_scene_challenge	Static	indoors, empty	1:02
26_new_orleans_drive_challenge	Motion	car driving, urban, outdoors, busy	1:00
27_times_square_challenge	Static	urban, outdoors, busy	1:02
28_dance_experience_challenge	Static, stereoscopic	urban, busy	1:02

3.3 Experimental Setup

In this section we will explain which devices were involved in the experiment and their distribution in the laboratory. The set-up for the experiment was simple. It consisted in the following elements, which can be seen in Figure 3.3:

- **Computer:** The computer was used basically as the working station where the rest of the elements were connected. It had two main functions that acted together: play the videos and gather the tracking data from the FOVE. As it has been mentioned above, we used the media player of SteamVR for displaying the videos to the HMD and in the moment when we played the videos the algorithm by Ioannis Agtzidis et al. [ASD19] would start the recording of the gaze and head movements. It also provided a connection to the loudspeakers.
- **Office chair** It was placed in front around one and a half meters from the PC in order to permit the 360° movements of the users. It kept the users in the same location during the whole experiment (except on breaks) and allowed them to freely rotate in order to explore all the parts of the scene.
- **FOVE + head-tracking camera** In this case, we placed the camera in front of the chair directly pointing to the user so nothing could interfere in the head tracking. The FOVE HMD was used by the subjects for visualizing the 360° immersive videos and by the experimenters for gathering the head and gaze data through the camera and through the built-in gaze tracker, respectively.
- **Stereo loudspeakers** They were placed one on each side of the computer, facing the user and reproducing the sound of the videos,

3.4 Experimental Procedure

In this section we proceed to the description of the whole experimental procedure. First of all, we selected fifty users which were related to the Chair of Media Technology of TUM: professors, students and staff. We scheduled the fifty participants in one week ending with ten participants per day, on average, divided in sessions of around forty-five minutes. The length of the session depended on the play of the twenty-eight one minute videos, plus the breaks that some of the participants took in order to rest from the immersive experience.

Each of the subjects had to fill up an Informed Consent Form for research involving human subjects. In this document, we informed about the experiment characteristics and the participant's exposure. We gathered personal information about age, gender, nationality and additionally some specific information for the study, such as if the user had visual impairment, if the user was wearing glasses or contact lenses, if they had a surgery in order to correct their vision and question regarding their previous experience with VR if existent.

Figure 3.5: Experimental set-up.

One of the most important questions relied on the motion sickness, because in the case of the motion videos there is an spatio-temporal incoherence between what the user's is watching (positional movement) and his actual static sitting position. This fact, affected some of the participants specially with the video nineteen and twenty-four, in which the movement of the camera is not stabilized.

After the signature of the Informed Consent Form, the participants were seated on the chair and had time to adjust it to their height as the experiment was demanding in terms of time. Note that the participants were offered a break to rest after half of the videos were played and they could also ask for one in any moment during the experiment if they were feeling unwell. Then, they had to wear the FOVE HMD and the experimenters helped them to tight up its three straps so they were fixed but confortable. Once they are wearing the HMD we asked them to do several rotations in the chair to check that the HMD cables were hanging behind the participan's head allowing the 360° movements. At this point we faced the participants with the FoV of the head-tracking camera.

Once the hardware was ready, we started the FOVE VR interface and proceed to a cali-

bration of the eye-tracker. Bear in mind that if the participants took a break, they must undergo through the calibration process before keep visualizing the following videos. The calibration is provided by the FOVE software and consisted in following a moving green dot with the gaze. After the user was calibrated, we played a virtual test scene in Unity which would highlight the fixated objects to check if the calibration was successful. Then, we started Steam and Steam VR. We made use of the media player from SteamVR to reproduce the videos in the FOVE setting the layout options to mono for equirectangular and top-down for the stereoscopic videos. Once the player was ready, we played two short clips for the participants to familiarize with the process. After that, we had to run the script that executed the *FoveDataToArff* tool from [ASD19] which would start recording the data once we started playing the video, until the moment that we stopped. This tool recorded the gaze by setting an offset from the starting and ending viewing point, for this reason at the initial and final seconds of the video a calibration point, were participants had to fixate at, was played in order to establish this offset.

Chapter 4

Saliency Optimization

The content of this chapter corresponds to the description of the temporal requirements of the real-time video saliency and also of the developed deep learning saliency model during the thesis, the MIX-Net. First of all, we will evaluate the model requirements for being able to predict saliency on video frames in real time. After that, we will proceed with an introduction of the concept of Frame FoV Saliency (FFVS) which will allow us to bound saliency not only to the general 360° scene, but for the current FoV of a given user. In the second section, we will describe the MIX-Net model for saliency prediction which is basically a CNN based on the classic VGG-16 network and influenced by two of the highest ranked approaches (with available code) in the MIT Saliency Benchmark [BJB+]: ML-Net [CBSC16] and MSI-Net [AK19]. Finally we will specify how the model was trained with the SALICON dataset [MJZ15] for its future test in the SALICON Saliency Prediction Challenge (LSUN 2017) [Zhab].

4.1 Temporal Saliency Requirements

In this thesis we aim to optimize the saliency prediction in the time domain. It is crucial to clarify that our main goal is to be able to predict video saliency by predicting the saliency map of each frame, specifically the maps of the FoV frames of a video. For this reason, we will use the equirectangular videos from our novel LMT Saliency Dataset described in the previous chapter. In this case the videos are, on average, one minute long and 30 FPS (frames per second). This information is what is going to be our boundary for considering the saliency prediction real-time.

Defining real time, it is a level of computer responsiveness that a user senses as sufficiently immediate, or that enables the computer to keep up with some external process. In our particular case, the user sensation of immediacy would be given by the hypothetical visualization of the saliency video at the same time as the original video. For this reason, our

objective is to build a model which is able to compute saliency maps of given images (or frames) in real-time for the videos of our dataset. Thus, as the videos are at 30 FPS that means that each frame is played at $0.0\widehat{3}$ seconds, or $33.\widehat{3}$ milliseconds.

Consequently, our goal will be to build a model which is capable of predicting saliency maps of video frames in less than $33.\widehat{3}$ milliseconds. This way we will be able to simultaneously view the original video and another one with the saliency maps of each of its frames, generating what we call a saliency video. It is important to note, that the saliency prediction speed will also depend on the size (resolution) of the input images. In other words, if the input image has a higher resolution it will take more time for the model to downsize it in order to make a prediction.

In the previous paragraph we have introduced the impact of resolution in the saliency prediction. Normally saliency models first reduce the resolution of the input images and then pass them through the NN. Once the prediction has been made, they have to re-scale it to the image original size. This fact incorporates two more stages for our aim to make it real-time: image preprocessing and postprocessing. This is one of the reasons why we introduce, additionally to saliency prediction on common images, the concept of Frame FoV Saliency (FFVS) which will be further described in the following section.

4.2 Frame FoV Saliency (FFVS)

In this section we present the novel approach of computing the saliency maps, not only for the whole equirectangular image frame but for the current field of view (FoV) of a given user. The idea is that commonly, the visual content in 360° static videos, which don't include any scene cuts, doesn't usually suffer any drastic variation that would represent a change in the salient points of the scene. This is the reason why in [VS] Vincent Sitzmann et al. compute the saliency map of the whole equirectangular scene, because in their case they use static scenes, not videos.

However, by being aware of where a person is fixating at in his current FoV may also be helpful for other tasks required in the field such as head-movement prediction. As discovered in [LR86] the head follows the gaze direction in what they named the vestibulo-ocular reflex. Hence, if we know the saliency map of a given FoV frame we may also be able to determine to which position is more probable that the user's head will move in the following frames of the video.

Furthermore, the saliency prediction of the actual FoV instead of the whole equirectangular scene may be advantageous in the reduction of the saliency prediction time. This is because, as we pointed out before, a smaller image input will be faster to produce and by cropping the current video frame following the individual's FoV we produce a much smaller input for the network. In the case of the FOVE HMD, which was used in our experiment, it implements a FoV of 100°. Thus, we recreate the specific FoV content of the scene that the

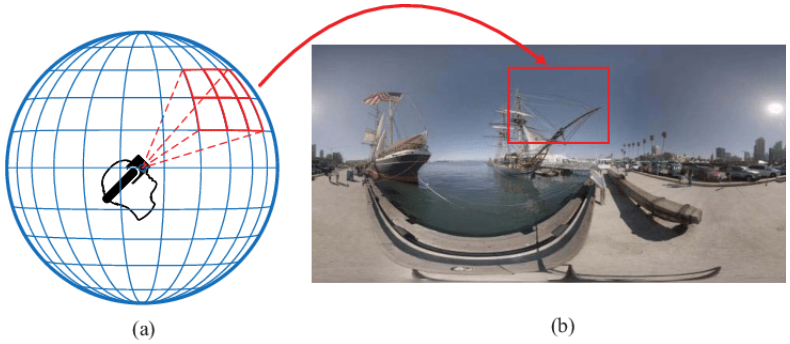

Figure 4.1: The spherical viewing space of 360° video and the equirectangular projection (ERP) plane of the video. (a) shows how observers can only see a part of the entire 360-degree visual scene at a single point of time. (b) illustrates the corresponding FoV in the ERP plane.

user was watching making use of the GitHub code[1] to compute the current FoV of each frame by using the θ and Φ angles from our dataset, which map the user's head position in a specific frame. The modification of the code was necessary as in the original script the author used a variation of the original spherical coordinates, but we finally managed to extract the original FoV from FOVE and thereby reducing the input image size from the original 4K (3840x2160) to 1080x720. Hence, we reach a reduction of about the 90% of the original frame size. This image size reduction can be appreciated in Figure 4.1, where we show the equirectangular projection of the 360° sphere and the corresponding FoV in a specific instant.

4.3 MIX-Net

This section provides a description of the motivation, inspiration, architecture and functioning of the developed deep learning model for saliency prediction, the MIX-Net. MIX-Net is a convolutional neural network for saliency prediction based on the popular VGG16 [SZ14b] network, which is widely used in the machine learning community for its simplicity and elegance.

4.3.1 Inspiration

When proceeding to develop a deep learning model several things must be taken into account. First of all, we need to decide if we want to use any state-of-the art saliency prediction models or if we prefer to develop a new model from scratch. The case of our

[1] https://github.com/fuenwang/Equirec2Perspec

MIX-Net corresponds to a combination of both: the backbone of the CNN is based on the well-known VGG16 [SZ14b] network developed by Karen Simonyan and Andrew Zisserman from the Oxford University, but with some modifications which have been inspired by two state-of-the-art networks, the ML-Net [AK19] and the MSI-Net [KSDG19].

In prior studies [DJV+13], it has been demonstrated that fratures exctracted from CNNs trained for other deep learning tasks, such as object recognition, generalize efficiently to another visual tasks like saliency prediction. This is why our network takes advantage of the VGG16 network, pre-trained on the ImageNet dataset [JDF09] for image classification purposes, in this case for object classification. The VGG16 has been precisely trained for three weeks in object recognition, therefore it is able to extract high-level quality features like faces or car-like structures and we intend to use these contextual features for our saliency map prediction. This idea has also been implemented in the MSI-Net [KSDG19] achieving competitive results on two public saliency benchmarks [BJB+, BI15].

Besides, we also introduce the idea of extracting the high level features from the last convolutional layer of the VGG16 architecture by removing the last pooling layer from the ML-Net [AK19].

The second component from our CNN which was motivated and adapted from [AK19] is the multi-level feature extraction network. This part basically consists in the the concatenation of the output three relevant layers from the VGG16, which benefits from the high, medium and low level features providing a valuable impact on the final result.

Therefore, the architecture of our network benefits from the already trained VGG16 on the ImageNet database and from the findings of two of the state-of-the-art models, which code is available, in the MIT Saliency Benchmark [BJB+]. Due to the fact that it is built combining multiple models, we decided to name it MIX-Net.

4.3.2 Architecture

The detailed architecture of our MX-Net is illustrated in Figure 4.2 and 4.3. As we can observe, it consists on thirteen convolutional layers, five pooling layers and three transposed convolutional layers after the encoding structure.

The functioning of the network consists in: given an input image, the VGG16 net, without the last pooling layer, extracts low, medium an high level features which are contextual based as we are using the pre-trained weights for the object classification task. Then, these multi-level features are concatenated and the output goes into the last part, which up-samples the output and it further has to learn which of all the filters learnt in the VGG16 network attract human attention. Thus, our network consists in two main parts: a feature extraction network (VGG) and an encoding network.

However, just like any other standard CNN architecture, we have the disadvantage of the size reduction of the feature maps at higher levels with respect to the original input size. As

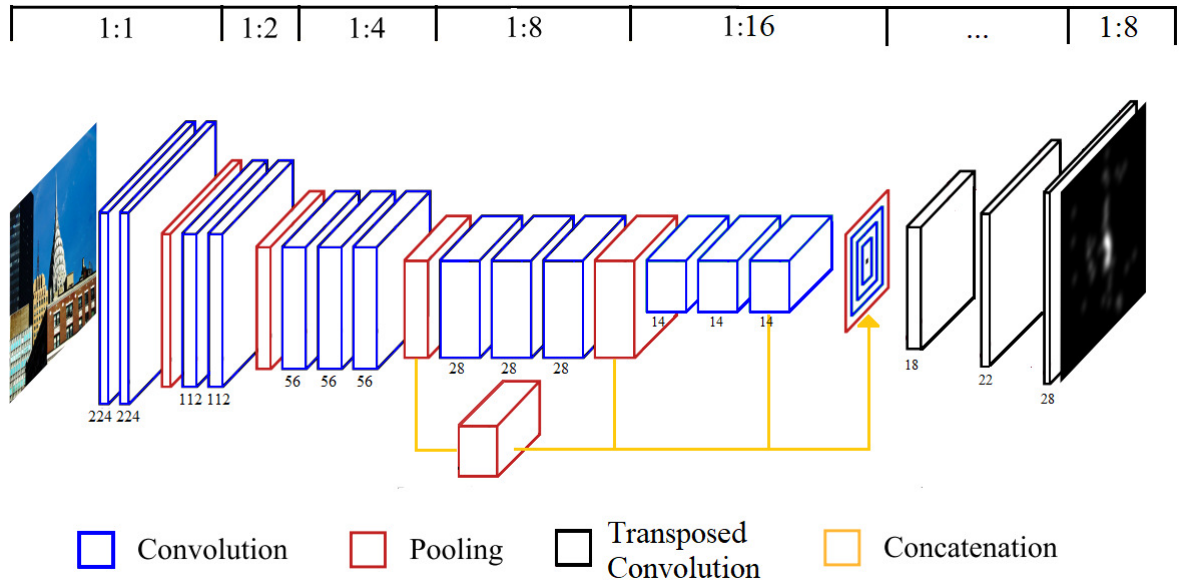

Figure 4.2: MIX-Net Architecture.

we have discussed in subsection 2.2.2, this is due to the presence of pooling layers which have a stride greater than one and hence, they reduce the input size. In our case, this reduction corresponds to half of the original image input size, as the kernel size and stride are equal to two. Thus, given an image with size $W \times H$, the output feature map will be downsized to $\left\lfloor \frac{H}{2^n} \right\rfloor \times \left\lfloor \frac{W}{2^n} \right\rfloor$ where n is the number of pooling layers in the network.

In the original VGG16 [SZ14b] the input images are 224×224 and it implements five pooling layers, thus the output feature maps from the last pooling layer are 7×7. In the case of the MIX-Net feature extraction network, the input images are also 224×224, but the difference is that, as we said before we emulate [AK19] and we avoid the last pooling layer of the VGG16 structure. Thus, the output feature maps will be 14×14.

Once we know how the VGG structure works, we proceed to describe the multi-level feature extraction or encoding network. As it can be seen in Figure 4.2, and inspired by both [AK19, KSDG19] we decide to concatenate the feature maps from three different parts of the network. Specifically from the output of the third pooling layer (which consists of 256 feature maps), the fourth pooling layer (512 feature maps) and from the last convolutional layer (512 feature maps). These maps will be referred as `block3pool`, `block4pool` and `block5conv3`, respectively, due to their position and function type in the network.

Both `block4pool` and `block5conv3` share the same spatial size (14×14), but in order to concatenate the three of them we need to reduce the size of `block3pool` from 28×28 to 14×14 and that is the reason for the late pooling in the model. Now that they are all the same size, they are concatenated and create a tensor of 1280 channels ($256 + 512 + 512$).

This tensor is then fed to three transposed convolutional layers which are in charge of learning which feature maps are relevant, among all the features that the VGG network can extract, and additionally they up-sample the output map until it has size 28×28. The first two transposed convolutional layers, learn 32 saliency-specific feature maps each, and finally the last 1×1 layer learns to weight the significance of each saliency feature map in order to generate the final predicted feature map. Note that all the layers from our CNN use the Rectified Linear Unit (ReLu) as activation function.

As it is shown in Figures 4.2 and 4.3, the input image size of our network is 224×224 and its output (saliency map) is 28×28. Hence, as usually images are not in this specific resolution we must establish a pre-processing mechanism for reducing the size of the input images. Also, as we often want a predicted saliency map which has the same resolution as the original image, we will also need to, first store the original image size and once our network has generated an 28×28 saliency map we need to post-process it and change its size to the original measures. In Figure 4.2, we also show in the top of the image the feature maps ratio with respect with the input image size in each of the stages, and the actual size of the maps below each layer.

4.3.3 Training

Once we have explained the architecture of our model, it is time to make it *learn* how to produce saliency maps. For this reason we will use one of the most used saliency datasets in literature, SALICON [MJZ15] by Ming Jiang et al.

Dataset Preparation

For any deep learning model that deals with large datasets, as it is the case of SALICON, it is convenient, and almost essential, to prepare the data before feeding it to the model and thereby, to optimize computing times and resources.

As we have explained in subsection 2.3.1, the SALICON dataset includes 10.000 images and their respective fixation maps as training set and 5.000 for the validation set. In our case, the input of our model is 224×224 so our first step was to re-scale all the images from the dataset to this specific resolution. We also normalized the training fixation maps such that all the values from a map were non-negative and with unit sum. Once we had our training dataset ready, we had to establish the network training parameters before starting the learning process.

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 224, 224, 3)	0	
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792	input_1[0][0]
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928	block1_conv1[0][0]
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0	block1_conv2[0][0]
block2_conv1 (Conv2D)	(None, 112, 112, 128	73856	block1_pool[0][0]
block2_conv2 (Conv2D)	(None, 112, 112, 128	147584	block2_conv1[0][0]
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0	block2_conv2[0][0]
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168	block2_pool[0][0]
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080	block3_conv1[0][0]
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080	block3_conv2[0][0]
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0	block3_conv3[0][0]
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160	block3_pool[0][0]
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808	block4_conv1[0][0]
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808	block4_conv2[0][0]
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0	block4_conv3[0][0]
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808	block4_pool[0][0]
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808	block5_conv1[0][0]
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 256)	0	block3_pool[0][0]
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808	block5_conv2[0][0]
merge_1 (Merge)	(None, 14, 14, 1280)	0	max_pooling2d_1[0][0] block4_pool[0][0] block5_conv3[0][0]
conv2d_transpose_1 (Conv2DTrans	(None, 18, 18, 32)	1024032	merge_1[0][0]
conv2d_transpose_2 (Conv2DTrans	(None, 22, 22, 32)	25632	conv2d_transpose_1[0][0]
conv2d_transpose_3 (Conv2DTrans	(None, 28, 28, 1)	1569	conv2d_transpose_2[0][0]

Total params: 15,765,921
Trainable params: 3,411,041
Non-trainable params: 12,354,880

Figure 4.3: MIX-Net Layer Summary

Loss function

The first aspect to determine is the loss function. The output of this function basically describes how similar or dissimilar is the result of a network from the expected result. In other words, it indicates the magnitude of the error that a model generated on its prediction. As saliency prediction is a regression task, it describes the difference between the values that the model is predicting and the actual values from the training and validation set. After each epoch, this differences or errors are backpropagated through the model to adjust its weights and make them closer to the actual ones in the next rounds. Backpropagation is basically the process that makes a NN *learn*.

In our case we tried different loss functions until deciding the definitive one: mean squared error, mean absolute error, poisson, etc. [Kera]. As we have previously seen, the saliency map estimation can be understood as a probability distribution prediction task as described

by Jetley et al. in [JMV18], and as we have normalized our training maps to be non-negative and to sum one, they can be seen as the probability distribution of fixations in an image. Hence, we decided to implement the *Kullback-Leibler* (KL) divergence as our loss function. As explained in subsection 2.3.3 it corresponds to a widely used measure in information theory for computing the statistical distance D between two distributions, and is given by the following formula:

$$D_{KL}(P \parallel Q) = \sum_i Q_i \ln \left(\epsilon + \frac{Q_i}{\epsilon + P_i} \right) \tag{4.1}$$

Where Q represents the ground truth distribution, P its estimate, i the index of each pixel and ϵ corresponds to a regularization constant. So, for our training we decided to make use of equation (4.1) as the loss function for our training, which would be gradually minimized through the *Adam* optimizer [KB14]. In this case we did not set any specific learning rate, and we used 1×10^{-3}, which corresponds to the advised value from the Keras documentation [Kerb].

Training Parameters

Once the loss function is defined, we must set some parameters for training our model in the task of saliency map prediction. These parameters are summarized in Table 4.1. As the SALICON dataset includes 10.000 images for training, we feed our network in batches of 100 images and thereby the model was trained for 100 epochs, with 100 steps per epoch.

Table 4.1: Model Training Parameters

Parameters	**Values**
Number of Images	10.000
Image Sizes	224 x 224
Loss Function	KLD
Optimizer	Adam
Learning Rate	0.001
Batch Size	100
Epochs	100
Steps Per Epoch	100

For training the model, as we wanted to take advantage of the learnt weights of the VGG16 for the image classification task, we proceed to freeze its layers. By freezing these layers, we prevent their weights to be modified. This approach is known as *transfer learning* which basically is focused on storing knowledge gained while solving one problem (image classification) and applying it to a different but related problem (saliency prediction).

Additionally, freezing layers during training has a positive impact on the training speed. As we do not intend to modify the weights of the VGG layers, the back-propagation through them can be completely avoided, resulting in a significant speed boost. Thereby, our model was trained for 100 epochs in approximately four hours.

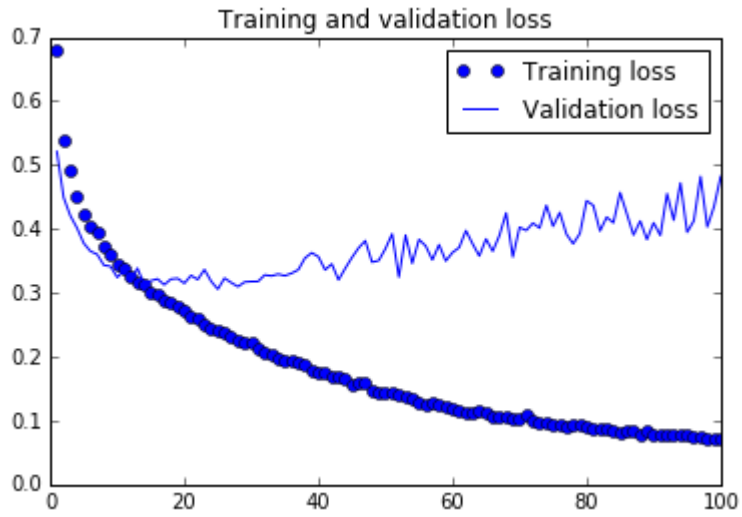

Figure 4.4: Training and validation loss of the MIX-Net trained for 100 epochs on the SALICON dataset.

In Figure 4.4 it can be appreciated the evolution of the training and validation losses during the training time. We can appreciate that our training loss in less than ten epochs is already less than 0.5 and in the last epoch its value was 0.0704. We can appreciate the problem of over-fitting in the validation loss, as it improves at fitting the data that the model sees (training data) while getting worse at fitting the data that it does not see (validation data). This issue, may be due to the used learning rate and the number of epochs. However, as we will see in the next chapter, despite the over-fitting the MIX-Net still managed to get competitive results in the MIT Saliency Benchmark [BJB+].

Chapter 5

Results and Evaluation

In this chapter the evaluation and test of the MIX-Net training will be described. First, we will show the accuracy results obtained by our approach as well as some predicted saliency maps with the ground truth from SALICON dataset [MJZ15] where we can observe the quality of our predictions. After that, we will show the results for the field of view saliency (FFVS) generated by the MIX-Net. Finally, in the third section of this chapter, we will evaluate the accomplishment of the saliency temporal requirements described in the first section of Chapter 4, and determine weather our model is capable of predicting saliency maps in a real-time basis.

5.1 Test of MIX-Net in SALICON Challenge

After conducting the training of our MIX-Net approach for 100 epochs on the SALICON dataset [MJZ15] in this section we will evaluate the results obtained in the LSUN 17 Saliency Prediction Challenge [Zhab] by the SALICON creators.

The SALICON dataset provides 10.000 training images and 5,000 validation images with their fixation maps as ground-truth. The test set with 5,000 images is released without ground-truth. The reason is that for testing a model, one should compute the saliency maps of the test set with a trained model and upload the resulting predictions to the LSUN 17 Saliency Prediction Challenge [Zhab].

In the LSUN Challenge, our deep learning model for saliency prediction, MIX-Net, obtained an overall score of 0.701 which basically corresponds to the score of s-AUC metric. In Table 5.1 we show the results obtained in the SALICON challenge compared to the best performing models in the MIT Saliency Benchmark [BJB$^+$] and the first row of infinite humans is shown as a reference. Note that the shown models in Table 5.1 were tested on [BJB$^+$], not on SALICON.

Table 5.1: Representative accuracy metrics comparison between state-of-the-art saliency prediction models and MIX-Net.

Models	AUC	sAUC	SIM	KLD	CC	NSS
Baseline: infinite humans	0.92	0.81	1	0	1	3.29
Deep Gaze 2	0.88	0.72	0.46	0.96	0.52	1.29
SALICON	0.87	0.70	0.68	0.54	0.74	2.12
MSI-Net	0.87	0.72	0.68	0.66	0.79	2.27
ML-Net	0.85	0.70	0.59	1.10	0.67	2.05
MIX-Net (our approach)	**0.823**	**0.701**	**0.67**	**1.04**	**0.735**	**1.53**

Regarding to the presented models in Table 5.1, DeepGaze 2 [KSAWB16] is shown in the table because it corresponds to the highest scoring model in [BJB$^+$], SALICON [MJZ15] is shown as it is the third best performing model, and MSI-Net and ML-Net [KSDG19, AK19] are also shown in order to compare our approach with those which inspired its architecture.

Furthermore, in the LSUN Challenge, the MIX-Net reached the position thirteen when sorting the models by AUC. Hence, we can affirm that the developed model in Chapter 3 obtained competitive results in one of the most popular saliency benchmarks. The ranking can be accessed through [Zhaa].

In Figure 5.1 we show the resulting saliency maps for some of the provided images in the validation set from the SALICON dataset. In the two first columns we show the original image and the given fixation maps as the ground truth; in the second column we show the predictions generated by the MIX-Net alongside the last column in which we overlap the original image and our prediction (as a heat-map) in order to provide a better visualization of the obtained results.

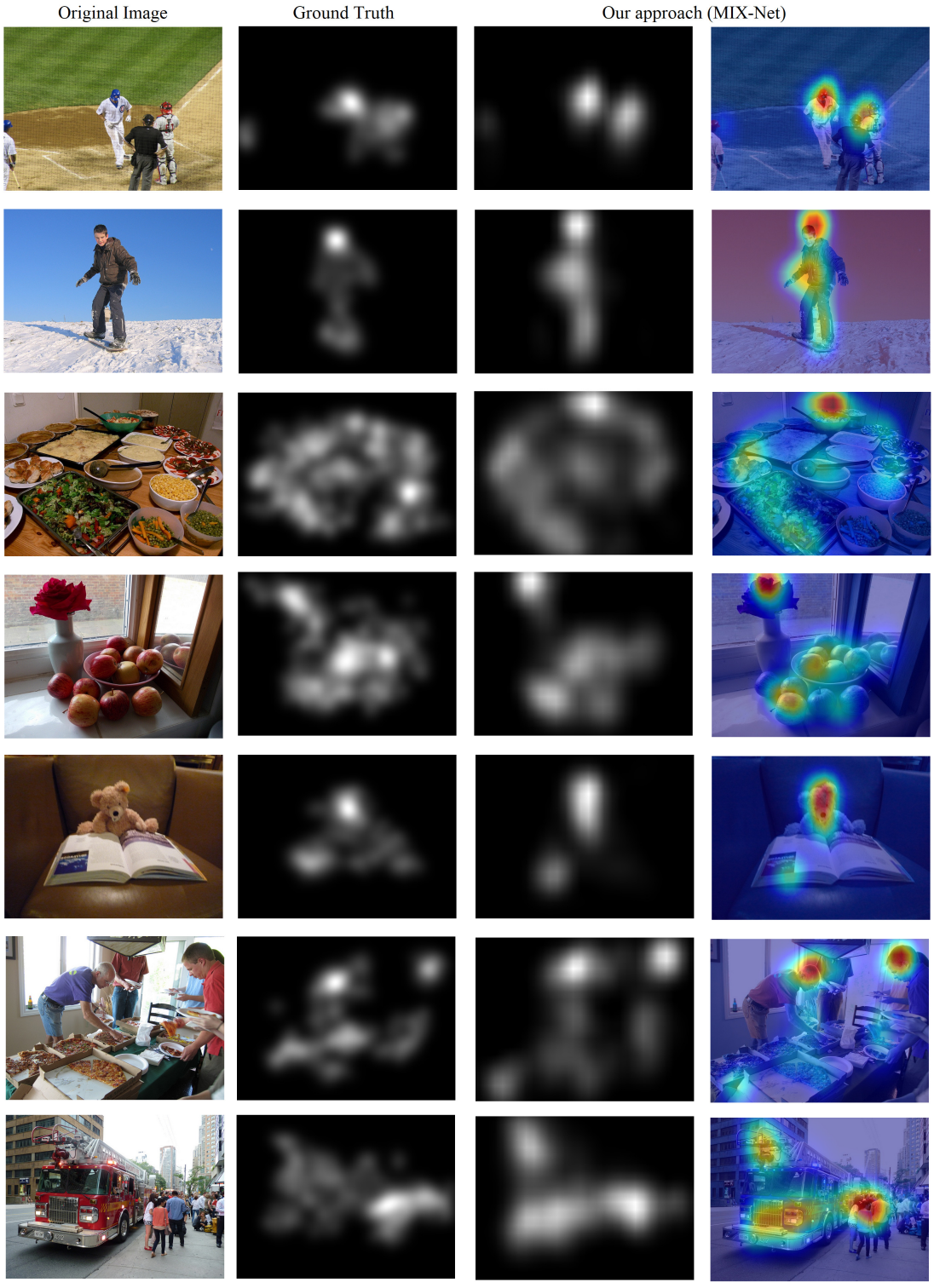

Figure 5.1: Comparison between the fixation maps (ground truth) provided by SALICON and the predicted saliency maps from MIX-Net. The first column corresponds to the original images extracted from the validation set of SALICON dataset, the second are the fixation maps generated after the SALICON visual attention study and the third and fourth column correspond to the saliency predictions generated by the MIX-Net.

5.2 Evaluation of MIX-Net on FFVS

In Section 4.2 the concept of Frame FoV Saliency (FFVS) was firstly introduced. In the following lines we will explain the process of the FoV frame extraction and the resulting saliency maps prediction.

After recording the experiments we had all the gaze and head-positions tracking data in the resulting ARFF files. Thus, we built an algorithm capable of extracting the head-position in every frame from the mentioned file given a certain subject and video number. After that, we used the modified algorithm mentioned in Section 3.2 to extract all the FoV frames from a user in one video. This way we recreated the path that the user followed during the experiment in his scene exploration. All the shown FoV frames were extracted from my own recording data in order to not compromise any participant's data.

Furthermore, we predicted the saliency maps for all the extracted FoV frames of the videos generating the FFVS. In Figure 5.2 we show one arbitrary FoV frame from eight of the twenty-three videos and its predicted saliency map by MIX-Net represented as a heat-map. Hence we can observe that MIX-Net also performs adequately for the given FoV frames. In the pair columns of Figure 5.2, it is noticeable how the MIX-Net net predictions are accurate in awarding the visually relevant pixels of the parts of the original images with a higher attention probability represented with the warm tones in the heat-maps.

Figure 5.2: FoV frames extracted from the videos from Table 3.1 next to their saliency map prediction generated by MIX-Net.The frames from the first column correspond to videos 1, 2, 4 and 9, while the ones from the second column correspond to the videos 13, 14, 16 and 23 from Table 3.1. The FoV shown corresponds to my own head-position recordings.

5.3 Temporal Evaluation of MIX-Net

One of the main purposes of this thesis was to build a saliency prediction deep learning model capable of generate saliency maps in real time. In Section 4.1, we determined an upper bound for the required time for the saliency maps generation. The required time for predicting saliency maps at 30 FPS was $33.\widehat{3}$ milliseconds. Hence, in our case we set a timer during the prediction of the FoV frames of a whole video obtaining an average prediction time per FoV frame of $23.\widehat{3}$ milliseconds. Furthermore, MIX-Net is capable of computing the FFVS of a whole video in less than 45 seconds, allowing a hypothetical delay in the prediction time and also the pre-processing adnd post processing delays.

Furthermore, it is necessary to reiterate that the lower the dimensions of the input image, the lower pre-processing time and thus the faster the prediction. For instance, as the validation images from SALICON are smaller than our extracted FoV frames (640×480) the average prediction time for an image during the generation of the test saliency maps was of $15.\widehat{7}$ milliseconds.

Thereby, we can conclude that our saliency prediction model, the MIX-Net, is capable of generating saliency maps in a real-time basis, accomplishing with one of the main objectives of the thesis.

Chapter 6

Conclusions and Future Work

In this chapter we will perform an evaluation of the obtained results in the developed work of this thesis and thereby determine if the objectives stated in Section 1.2 have been achieved.

Our first objective consisted in gathering the eye and head movement data from a considerable number of subjects, while wearing VR headset and visualizing 360° immersive videos, in order to create a VR saliency data for dynamic media content. As it is described during Chapter 3, we first created a VR data collection set-up in the laboratory of the Chair of Media Technology from the Technical University of Munich (TUM).

During the experiment we studied and gathered the gaze and head positions of fifty users related to the chair while watching twenty-eight short 360° immersive videos. One of the contributions of this task to the field of computer vision and visual attention is that, apart from the usual *free-viewing* approach used in previous datasets, the visualization of the last five videos was biased by the experimenters through the assignment of some challenges to the users. By performing this modification, the fixation data from the users in this last five videos is biased in a top-down basis. This information might be useful for the study of content-biased exploration of virtual scenes such as VR gaming, where the eyes and head positions of the users may vary depending on the task or challenge assigned to them.

Respecting the further work to be done regarding our first objective it must be noted that the dataset is not completed yet. First of all, because the number of participants should be increased in order to provide reliable samples for the future applications of this dataset. Following this guideline, the supervisor of this thesis in TUM, M.Sc. Tamay Aykut, has the intention of keep recording this data during his future stay in the Stanford University in the United States. Thereby, the idea is to create a joint dataset between TUM and Stanford with the the visual stimuli and the procedure designed during the first part of this Bachelor Thesis.

Furthermore, as for the moment we only provide the head positions and fixation data

extracted from the FOVE, it will be also necessary to derive the fixation maps from this data. This way, and with the further collection of data in the University of Stanford we will made our contributions in the publication of a large, novel and quality datasets for dynamic VR immersive content.

However, the second part of this work was to develop a deep learning saliency model which is able to predict saliency maps in real time. In the introduction of the thesis it was emphasized the importance of this predictions to be generated in real time, in order to recreate the cognitive process of attention of the human brain.

During Chapter 4 we described the inspirations and the architecture of our saliency prediction model, the MIX-Net. Which corresponds to a deep learning model based on the popular VGG [SZ14b] architecture. Once the structure of the model was described, we show the parameters and results of the training process of the MIX-Net on the SALICON dataset [MJZ15]. Also, we derived the temporal performance requirements for the prediction of the saliency maps in real time and introduced the concept of FFVS which consisted in generating the saliency prediction for the FoV frames of a 360° immersive video.

Moreover, in Chapter 5 we show the value of the performance metrics obtained by the MIX-Net in the LSUN 17 Saliency Prediction Challenge [Zhab]. In this ranking, our model obtained competitive results when tested on the most used metrics for saliency prediction evaluation, achieving the thirteenth position in the ranking when sorted by the AUC metric.

In addition, we fulfilled the temporal requirements described in Section 4.1 for achieving a real-time performance. Specifically, the MIX-Net took less than $33.\widehat{3}$ milliseconds in average for generating the saliency maps of the FoV frames of any of the videos from the developed datasets with the visual data from the author of this work. This margin, might also allow to the real-time post-processing of this frames in order to be played simultaneously with the original video. Furthermore, the novel concept of FFVS might be convenient for certain state-of-the-art technological applications such as telepresence, teleoperation and autonomous driving.

The future work to be done regarding the second part of this study is broad. As described in the Subsection 4.3.3 we encountered the problem of over-fitting with the validation data from SALICON. For this reason, it would be appropriate to modify the training steps of the MIX-Net, for instance by increasing the batch size and reducing the number of training epochs and learning rate. After performing this modifications it will also be necessary to test the model again in the LSUN Challenge [Zhab] in order to know if we outperform the obtained results by the MIX-Net shown in this thesis.

Also, following the idea of evaluating the performance of our developed model it would be convenient to also test the model in the MIT Saliency Benchmark [BJB$^+$] in order to observe how the MIX-Net model performs compared to the state-of-the-art saliency prediction approaches. Additionally, it might be insightful to train the model in other datasets like MIT300 or CAT2000 [BJB$^+$, BI15] and test its performance again on the

already mentioned saliency benchmarks.

In order to improve the temporal performance of the MIX-Net, it might be pertinent to apply state-of-the-art approaches for boosting its speed like the novel pruning techniques or fine-tuning. This way we might make our model smaller in size, more memory-efficient, more power-efficient and faster. Finally, after applying these techniques, it might be interesting to perform a comparison between the accuracy and the temporal performance of the model, which are estimated to be inversely proportional.

List of Figures

List of Tables

Bibliography

[ABI12] D. N. Sihite A. Borji and L. Itti. Quantitative analysis of human-model agreement in visual saliency modeling: a comparative study. pages 55–69, 2012.

[ACC12] Gelu Ionescu Antoine Coutrot, Nathalie Guyader and Alice Caplier. Influence of soundtrack on eye movements during video exploration. *Journal of Eye Movement Research*, 2012.

[AK19] Kurt Driessens Rainer Goebel Alexander Kroner, Mario Senden. Contextual Encoder-Decoder Network for Visual Saliency Prediction. 2019.

[ASD19] Ioannis Agtzidis, Mikhail Startsev, and Michael Dorr. A ground-truth data set and a classification algorithm for eye movements in 360-degree videos. 03 2019.

[ASM17] J. Ruiz-Borau G. Wetzstein D. Gutierrez A. Serrano, V. Sitzmann and B. Masia. Movie editing and cognitive event segmentation in virtual reality video. 2017.

[BI13] Ali Borji and Laurent Itti. State-of-the-Art in Visual Attention Modeling. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 35(1), 2013.

[BI15] Ali Borji and Laurent Itti. Cat2000: A large scale fixation dataset for boosting saliency research. 05 2015.

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

[BJO+16] Zoya Bylinskii, Tilke Judd, Aude Oliva, Antonio Torralba, and Frédo Durand. What do different evaluation metrics tell us about saliency models? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP, 04 2016.

[BT09] N. D. B. Bruce and J. K. Tsotsos. Saliency, attention, and visual search: an information theoretic approach. 2009.

[CBSC16] Marcella Cornia, Lorenzo Baraldi, Giuseppe Serra, and Rita Cucchiara. A Deep Multi-Level Network for Saliency Prediction. In *International Conference on Pattern Recognition (ICPR)*, 2016.

[DJ95] Humphreys GW.d Duncan J. Neural mechanisms of selective visual attention. *Review of Neuroscience*, 1995.

[DJV+13] 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. *arXiv preprint*, 32, 10 2013.

[ED18] Antoine Coutrot-Matthieu Perreira da Silva Patrick Le Callet Erwan David, Jesus Gutierrez. A Dataset of Head and Eye Movements for 360° Videos. *ACM Multimedia Systems Conference*, 2018.

[ESP08] Wolfgang Einhäuser, Merrielle Spain, and Pietro Perona. Objects predict fixations better than early saliency. *Journal of vision*, 8:18.1–26, 02 2008.

[GAMM93] Christiane Fellbaum Derek Gross George A. Miller, Richard Beckwith and Katherine Miller. Introduction to WordNet: An On-line Lexical Database. 1993.

[Gol08] Philippe Golle. Machine Learning Attacks Against the Asirra CAPTCHA. 2008.

[HB05] M. Hayhoe and D. Ballard. Eye Movements in Natural Behavior. *Trends in Cognitive Sciences*, 9, 2005.

[his] History of artificial intelligence, url = https://en.wikipedia.org/wiki/History-of-artificial-intelligence.

[HZRS15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. 7, 12 2015.

[JDF09] R. Socher L.-J. Li K. Li J. Deng, W. Dong and L. FeiFei. ImageNet: A large-scale hierarchical image database. *CVPR*, 2009.

[JJMJ18] John Pearson Jeff J. MacInnes, Shariq Iqbal and Elizabeth N. Johnson. Wearable Eye-tracking for Research: Automated dynamic gaze mapping and accuracy/precision comparisons across devices . 2018.

[JLG10] T. Huang J. Li, Y. Tian and W. Gao. Probabilistic multi-task learning for visual saliency estimation in video. pages 150–165, 2010.

[JM10] Wolfe JM. Visual search. *Curr Biol*, 2010.

[JMV18] Saumya Jetley, Naila Murray, and Eleonora Vig. End-to-end saliency mapping via probability distribution prediction. 04 2018.

[Kag] Kaggle. Dogs vs. cats, url = https://www.kaggle.com/c/dogs-vs-cats.

[KB14] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[Kera] Keras. Keras loss functions, url = https://keras.io/losses/.

[Kerb] Keras. Keras optimizers, url = https://keras.io/optimizers/.

[KKG17] Virginia FJ Newcombe-Joanna P Simpson Andrew D Kane David K Menon Daniel Rueckert Konstantinos Kamnitsas, Christian Ledig and Ben Glocker. Efficient multi-scale 3d cnn with fully connected crf for accurate brain lesion segmentation. 2017.

[KSAWB16] Matthias Kümmerer, Thomas S. A. Wallis, and Matthias Bethge. Deepgaze ii: Reading fixations from deep features trained on object recognition. 10 2016.

[KSDG19] Alexander Kroner, Mario Senden, Kurt Driessens, and Rainer Goebel. Contextual encoder-decoder network for visual saliency prediction. 02 2019.

[LIN98] C. Koch L. Itti and E. Niebu. A Model of Saliency-Based Visual Attention for Rapid Scene Analysis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20(11), 1998.

[LR86] V. Laurutis and D. Robinson. The vestibulo-ocular reflex during human saccadic eye movements. *The Journal of Physiology*, 373, 1986.

[MB13] O. Le Meur and Baccino. Methods for comparing scanpaths and saliency maps: strengths and weaknesses. 2013.

[MCK09a] E. P. Frady M. Cerf and C. Koch. Faces and text attract gaze independent of the task: Experimental data and computer model. 2009.

[MCK09b] E. P. Frady M. Cerf and C. Koch. Faces and text attract gaze independent of the task: Experimental data and computer model. 2009.

[MJZ14] J. Xu M. Jiang and Q. Zhao. Saliency in crowd. 2014.

[MJZ15] J. Duan M. Jiang, S. Huang and Q. Zhao. SALICON: Saliency in Context. *CVPR*, 2015.

[MKB14] William Patera Moritz Kassner and Andreas Bulling. Pupil: An Open Source Platform for Pervasive Eye Tracking and Mobile Gaze-based Interaction. 2014.

[MKB17] Thomas S.A. Wallis Matthias Kummerer and Matthias Bethge. Saliency benchmarking made easy: Separating models, maps and metrics. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 2017.

[NH16] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. page 4293–4302, 2016.

[NRD13] M. Mancas B. Gosselin N. Riche, M. Duvinage and T. Dutoit. Saliency and human fixations: State-of-the-art and study of comparison metrics. *ICCV*, 2013.

[Ros] Adrian Rosebrock. Imagenet: Vggnet, resnet, inception, and xception with keras, url = http://cs231n.github.io/neural-networks-1/.

[RS] A. Reeves and G. Sperling. Attention gating in short-term visual memory. pages 180–206, 04.

[SG00] D. Salvucci and J. Goldberg. Identifying fixations and saccades in eyetracking protocols. 2000.

[Sli] Peter Slijkhuis. TobiiPro Glasses 2. *Department of Geo-Information Processing Faculty ITC and Department of Cognitive Psychology and Ergonomics Faculty of Behavioural, Management and Social Sciences University of Twente.*

[SLJ+15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 06 2015.

[SMP07] M. Guironnet S. Marat and D. Pellerin. Video summarization using a visual attention model. 2007.

[SRC10a] N. Sebe M. Kankanhalli S. Ramanathan, H. Katti and T. S. Chua. An eye fixation database for saliency detection in images. *ECCV*, (30), 2010.

[SRC10b] N. Sebe M. Kankanhalli S. Ramanathan, H. Katti and T. S. Chua. An eye fixation database for saliency detection in images. pages 30–43, 2010.

[SZ14a] C. Shen and Q. Zhao. Webpage saliency. 2014.

[SZ14b] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv 1409.1556*, 09 2014.

[Tat07] B. W. Tatler. The central fixation bias in scene viewing: Selecting an optimal viewing position independently of motor biases and image feature distributions. *Journal of Vision*, 2007.

[TJT09] F. Durand T. Judd, K. Ehinger and A. Torralba. Learning to predict where humans look. *ICCV*, 2009.

[TJT11] F. Durand T. Judd and A. Torralba. Fixations on lowresolution images. pages 1–20, 2011.

[VS] Amy Pavel ManeeshAgrawala Diego Gutierrez Belen Masia Gordon Wetzstein Vincent Sitzmann, Ana Serrano. How do people explore virtual environments?

[WH04] Jeremy Wolfe and Todd Horowitz. What attributes guide the deployment of visual attention and how do they do it? *Nature reviews. Neuroscience*, 5:495–501, 07 2004.

[YSG16] D. Jayaraman Y. Su and K. Grauman. Automatic cinematography for watching 360° videos. 2016.

[YTW14] Marc'Aurelio Ranzato Yaniv Taigman, Ming Yang and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. page 1701–1708, 2014.

[Zhaa] Catherine Qi Zhao. Ranking of salicon saliency prediction challenge (lsun 2017), url = https://competitions.codalab.org/competitions/17136results.

[Zhab] Catherine Qi Zhao. Salicon saliency prediction challenge (lsun 2017), url = http://salicon.net/challenge-2017/.