Making Machines More Human: Neural Models for Visual Question Answering with Human Attention

Gonçalo Migueis de Matos Afonso Correia

Master of Science
Artificial Intelligence
School of Informatics
University of Edinburgh
2017

Abstract

A lot of recent work in NLP and computer vision has focused on visual question answering (VQA), the task of automatically generating an answer to a question about an image. Existing VQA models are not designed to mimic human attention and do not have access to human attention data at training time. Using an existing dataset of human attention in the context of VQA, we are able to build a new VQA model that is able to learn the VQA task while attending to images the way humans do, using Multitask Learning. The cost function and attention mechanism used to learn both of these tasks at the same time can be easily integrated into any other model that uses soft attention. We will show that our model is able to obtain a human rank correlation of 0.59, which is very close to human agreement correlation, all the while improving the accuracy of our VQA baseline model by $+0.9\%$.

Acknowledgements

I want to thank my supervisor, Frank Keller, for all the guidance provided throughout this project.

I also want to thank my family for their love and support, not only during this dissertation but through all my life.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Gonçalo Migueis de Matos Afonso Correia*)

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The task of Visual Question Answering (VQA) [1] consists of obtaining an answer to a question related to an image. The most recent attempts at the task of VQA have found successful results by using attention-based neural network models. These models focus on a particular section of the image in order to answer the question. This is clearly inspired by the way humans do the same task: after reading the question, they search the image for a region that contains the answer and verbalize it.

However, these attention-based models have been found to not attend to regions of images in the same way that humans do. Das *et al.* [2] have found that there is little correlation between the attention maps generated by these mathematical models and the attention maps generated by humans when presented with a VQA task. This result is somewhat expected since the attention mechanism these models use is unsupervised - they are only generating a spatial distribution aiming to optimize the cross-entropy error of getting the right answer for the image-question pair.

There have been a handful of research that uses human data to aid a machine learning model in a specific task, but none, to our knowledge, in VQA.

1.1 Motivation

Along with their findings of little correlation between the attention maps of VQA models and the attention maps of humans, Das *et al.* made their data set of human attention publicly available. This work aims to use this new data set by extending a VQA model to learn to attend to images the same way humans do while doing this task.

Our hypothesis is that, using Multitask Learning [3] and these human attention maps, in a neural network model that uses an attention mechanism, we will be able to

develop a model that is able to attend to different regions of the images more closely to the way humans do, thus creating a model that is able to mimic human behavior in this task.

1.2 Contributions

Our contributions in this dissertation are two-fold:

- Present a seamless Multitask Learning setting that is able to both learn the VQA task and the human attention task and which can easily be incorporated in any model that uses attention.
- Present an attention mechanism that is able to attend to images more closely to the way humans do when performing the VQA task. This attention mechanism can also be seamlessly integrated into any neural network model that uses soft attention.

To the best of our knowledge, this is the first work to jointly learn to perform the VQA task while learning to mimic human attention.

1.3 Structure and Overview

This dissertation has the following structure:

- **Chapter 2** presents an overview of the most important background knowledge of Machine Learning, Deep Learning, and tasks presented in this work that are crucial to the understanding of this dissertation.
- **Chapter 3** describes the most relevant related work of this dissertation, explaining the model details of all our baselines.
- **Chapter 4** presents the methodology of the work performed in this dissertation. It describes the data sets used, the models implemented and how they are going to be evaluated.
- **Chapter 5** shows the results of the models presented in Chapter 4. The experimental details are described, the models are evaluated and a discussion and analysis of the results are followed.

- **Chapter 6** gives the concluding remarks of this work, along with suggestions for future work in this area.

Chapter 2

Background

In this chapter, we will review some concepts of Machine Learning, Deep Learning, and tasks that are actively researched in the field. These concepts are essential to understand this work.

2.1 The Visual Question Answering Task

The term "AI-complete" is used informally in the field of Artificial Intelligence to describe a problem of this field that solves, or comes close to solving, intelligence [4]. The problem of Visual Question Answering (VQA) was first introduced in Antol *et al.* [1] with the goal of improving the "AI-*completeness*" of the task of image captioning. Image captioning did not imply a complete scene-understanding of the image as was initially thought - a general description of the scene may not include several elements and details of it.

The task of VQA is the task of taking an image and a question about it as an input and producing an answer to this question. The questions available in the VQA data set [1] [1] are *free-form* and *open-ended*, that is, the questions do not limit respondents to a set of limited answers and the question may be open to interpretation. Therefore, to answer such questions, an intelligent system needs a multi-modal knowledge of several domains, from object detection knowledge to common-sense reasoning (see Figure 2.2).

Furthermore, when compared to the task of image captioning, VQA has a much better-defined evaluation metric. Image captioning resorts to evaluation metrics like BLEU [6] - a metric commonly used to evaluate Machine Translation systems - that is

[1] `http://www.visualqa.org/`

Figure 2.1: Image and respective caption, generated by the neural network model NIC [5]. The caption correctly describes the image but fails to capture details of it (e.g. one can not know from the caption that the boy on the center has a red shirt and blue shorts).

Q: Is the bed neatly made? A: yes

Figure 2.2: Example of an image and one of its possible questions and respective answer, available in the VQA data set. To correctly answer the question, a system needs to understand not only the concept of the object “bed”, but also needs to grasp the concept of what it means for a bed to be made.

not suited to evaluate how good a description of the image is, since several different correct captions could be written for the same image. However, in a VQA setting, to correctly answer a question about the image, one only needs to produce a mostly unique answer of one to three words. This way, the evaluation of a VQA system can be easily done by checking the percentage of correct answers it produces.

2.2 Human Attention and Saliency Prediction

Every day, we are surrounded by ever-present, multiple and simultaneous stimuli, however, our brain guides our eye gaze to the most salient regions of our visual field at that time. The retina is incapable of gathering all the details of the visual field since it is only capable of having high resolution in just a few degrees of visual angles [7] - this requires the brain to have an attention mechanism that is able to know where to look, at the right moment.

This attention mechanism can be triggered by cognitive factors (*top-down*), where knowledge from previous experience, expectations and the current task take the main role in influencing where to look, or *bottom-up* factors, where the information flow comes directly from sensory stimulation [8]. The former is thus the mechanism that controls our attention when we are, for instance, looking for a specific object in our room, while the latter is largely the reason why STOP signs are red - the bright color greatly stimulates our sensory input and leads our brain to look at it when it enters our field of vision.

The prediction of this mechanism, especially the one caused by bottom-up factors, has a large body of research and is usually named *saliency prediction*. The ability to correctly predict human saliency has several applications, from video compression to video surveillance systems [9, 10]. The data sets used for saliency prediction typically consist of gaze data - the locations of the eye gaze over time when looking freely at an image - and fixation (or saliency) maps - a gray scale heat map of the main locations of fixation of the eye gaze in the image.

Most of these data sets have been gathered by collecting eye tracking data by recording the eye movements of a subject through a camera and infrared sensors. This type of data collection is very expensive and thus large-scale data collection of eye tracking is financially implausible. In fact, one of these data sets, POET, is the largest data set of gaze tracking data and consists of $6,270$ images collected solely from 5 subjects [11]. However, it has been found that mouse tracking data is quite correlated

with human gaze and thus, using Amazon Mechanical Turk to obtain gaze data and saliency maps, the data set SALICON was developed, obtaining saliency prediction data for $10,000$ images from several subjects [12].

Inspired by the procedure for data collection for SALICON, the data set VQA-HAT [2] was also gathered through mouse tracking data. In this data set, the subjects were not only presented with an image but also with a question about it, in order to obtain human saliency information for the VQA data set explained in the previous section. The presence of the question along with the image implies that the subjects looked at the image with cognitive information, leading to a data set of top-down attention instead of bottom-up, which is in great contrast to the SALICON data set.

2.3 Neural Networks

As a result of the advent of the backpropagation algorithm [13] and, much later, the computing capability of Graphics Processing Units (GPUs) [14] that led to the breakthrough of building very deep architectures, neural networks are one of the most used algorithms in the field of Machine Learning. The success of this algorithm when compared to others largely comes from the flexible function approximation capacity of the model. In fact, given enough hidden units, it has been shown that a neural network can approximate any function - this is often called the Universal Approximation Theorem [15].

Subsequently, the most relevant strength of a neural network is its capacity to learn feature representations from such high dimensional data such as images or text. Whereas in other algorithms the data scientist may need domain-specific knowledge about the problem to come up with meaningful features that work with the chosen algorithm, a neural network does the feature engineering by itself. It does this by hierarchically learning representations of the data - while lower layers try and capture finer details of the data, higher layers capture instead more abstract features that are built upon the features that come from the lower level layers. This means that neural networks excel at Representation Learning - they have the capability to learn how to build representations of the data that make it easier for the top layers (the classifiers) to extract the most useful information to correctly predict the outcome of an example [16].

Hence, it must not come as a surprise that the most successful attempts at tackling the Visual Question Answering task use neural networks [17, 18, 19, 20, 21, 22, 23, 24,

25]. As mentioned, this task demands the model to grasp simple and abstract concepts about the images and the questions given as input to it and using neural networks is currently the best solution to solve this problem. In the following sections, we will describe some deep learning concepts that will be useful to understand the state of the art of VQA models.

2.3.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) [26] are a variety of neural networks that take advantage of the spatial structure of the data, if the data has a grid like structure. This property of data is commonly associated with images and thus using CNNs led to remarkable results in a variety of vision tasks, for example, image classification [14]. However, CNNs can also be used for time-series or text data given the correct data representation, as will be explained in the next section.

Instead of a simple matrix multiplication of the features with the network's weights, CNNs do the convolution operation between the data and the weights, often named *kernels*. The kernels consist of a defined number of $k \times k$ grids of weights with a bias for each grid. A convolution ($*$) between input I and a kernel K can be calculated in the following way (see Figure 2.3 for a visualization) [27]:

$$(I * K)(i,j) = \sum_m \sum_n I(i-m, j-n) K(m,n) \tag{2.1}$$

The output of this computation is often called a *feature map*. The number of feature maps after a convolutional layer is determined by the number of kernels chosen. It is desirable to choose several kernels in a convolutional layer since it is then possible to extract different features from the same input. As in feedforward neural networks, after the linear activation of the input with the kernels, the feature maps are run through a nonlinear activation (e.g. rectified linear unit [28]) and learning can be made through the usual backpropagation algorithm [13].

Convolutional layers have several useful properties that help to improve the performance of a model [27]:

- **Sparse interactions** - in a feedforward neural network each input unit is connected to each output unit through a weight. By using kernels smaller than the input, a convolutional layer forces a sparse connectivity between input and output units. Sparse connectivity or sparse interactions have been shown to aid generalization in supervised classification tasks [30].

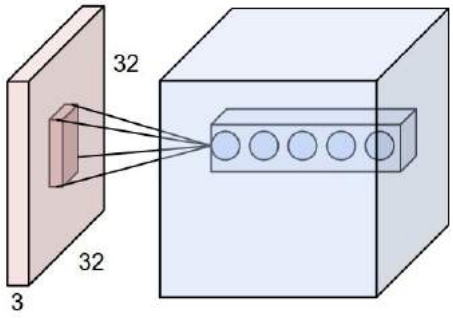


Figure 2.3: Sketch of a convolutional layer, from the CS231 course website [29]. An input of $3 \times 32 \times 32$ is transformed in 5 feature maps through a convolution with 5 kernels.

- **Parameter sharing** - the same kernel is applied to each section of the input. This greatly reduces the parameters making the model more efficient and less prone to overfitting when compared to feedforward networks.

Besides convolutional layers, CNNs usually have in their architecture *pooling layers*. Pooling layers are used to reduce the dimensionality of the input by making a summary of nearby input units. The most used example of pooling is max pooling [31], where only the maximum of each $k \times k$ region of the input passes through to the output unit (see Figure 2.4).

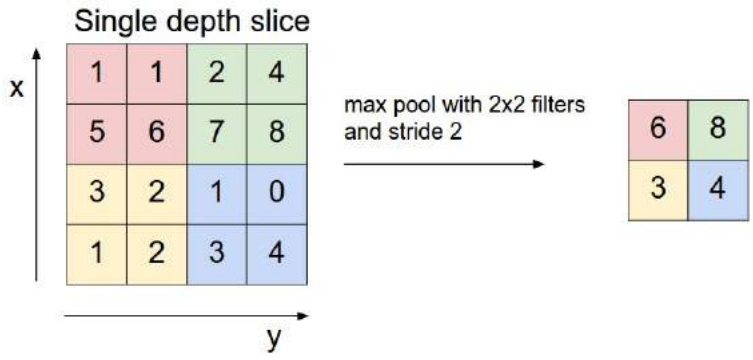


Figure 2.4: Sketch of a max pooling layer, from the CS231 course website [32]. An input of 4×4 is transformed in an output of 2×2 by taking the maximum of each 2×2 region with a stride of 2.

Pooling is especially useful to turn the feature representation output of a convolutional layer invariant to small translations of the input, that is, pooling helps to turn the model invariant to where a feature of the input is, making it instead more sensible to the existence of the feature. This is desirable, for instance, in detecting if an ob-

ject is present in an image or not - it does not matter *where* the object is, only if it is *somewhere* in the image.

All of the above properties of CNNs are helpful in the construction of a VQA system. The output of an architecture of CNNs specialized in image classification has a desirable representation of features useful to later correctly answer the image related question.

2.3.1.1 Convolution Layers in Natural Language Understanding

In order to obtain a good feature representation of a word, much research has been done in word vector representations through neural language models [33, 34]. To obtain a vector representation of a word, also called an *embedding*, a fixed vocabulary of words, represented in a one-of-V encoding (V being the size of the vocabulary), is projected to a lower dimensional vector space through a hidden layer. These representations encode semantic features of the words and it can be observed that the vector distance - e.g. cosine distance - between two similar or related words is lower than the distance between the embeddings of two unrelated words.

These embeddings can then be concatenated one after another to obtain a $2D$ feature representation of a sentence and run through a CNN [35]. If the word vector representation chosen has k dimensions and a sentence has n words, a kernel K of size $h \times k$ can be used to perform a convolution with a window of size h to the sentence representation S:

$$\mathbf{c} = f\left(K * S + b\right) \ , \ \ K \in \mathbb{R}^{h \times k}, \ S \in \mathbb{R}^{n \times k} \tag{2.2}$$

where f is the nonlinear activation of the layer and $\mathbf{c}$ is the corresponding output feature map of the sentence embedding. As in image CNNs, several kernels can be used to extract different features of the training sentences. Different sizes for h can also be chosen, creating a conceptually similar feature extracting process to the usual *N-gram* features commonly used in Natural Language Processing [36]. In order to obtain a feature vector with a fixed length despite the variable n sentence size, a max-pooling can be performed for each vector $\mathbf{c}$, which results in a feature vector of the same size as the number of kernels used. The final feature vector can then go through feedforward layers to obtain a classification of the sentence. Figure 2.5 shows a schematic of this process.

The most useful property of CNNs in a natural language context is the temporal

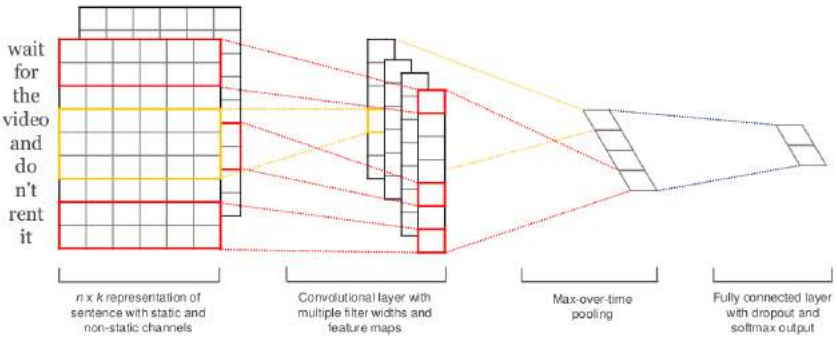


Figure 2.5: Sketch of a CNN in a natural language context, from [35]. In red, a kernel extracts bigram features of the sentence, while in yellow another kernel extracts trigram features. 4 kernels result in 4 vectors $\mathbf{c}$, which are then run through a max pooling layer, resulting in a feature vector of size 4. This feature vector can then be processed through a feedforward layer to classify the sentence.

invariance of the convolution since the same kernel parameters are used throughout the sentence due to parameter sharing and the max pooling extracts the most relevant feature over time for each feature map. While this property is not ideal for a task that requires temporal knowledge of the sentence - where an LSTM network architecture [37] is more suitable - it can be useful in a classification task where the position of the N-grams in the sentence is irrelevant but rather the occurrence of a desirable feature is pursued.

2.3.2 Visual Attention Models

A typical neural network model applied to vision (e.g. a CNN model) needs to process every pixel in the image, leading to possibly useless information from irrelevant pixels being carried to the higher layers for the classification task, which in turn may corrupt the useful information of the representations that reach the higher layers and worsen the performance of the model. This is in great contrast to the human visual system - we are able to focus our attention on the most relevant regions of our visual field and simplify our representation in order to excel at the current task (see section 2.2).

Attention based models try to bring this concept to neural networks and have been a great focus of deep learning research in recent years, with several results showing that this mechanism is essential to obtain more meaningful representations that are capable of improving the performance of neural networks in several tasks. Some of the most

noteworthy results associated with attention models include improved performance in machine translation [38], image captioning [39] and object recognition [40, 41].

There are several types of attention used in the literature, but we will restrict our explanation to the only one relevant to this dissertation - *soft attention*.

In a soft attention model, also called *deterministic attention*, we aim to compute vectors $\mathbf{z}_i$ that are the weighted representations of the feature vectors $\mathbf{a}_i$, for $i = 1, ..., L$, extracted from L regions of an image. For each of these regions, the model computes a weight α_i that corresponds to a relative importance to give to the feature vector $\mathbf{a}_i$. This can be implemented by a feedforward neural network layer with weights $\mathbf{w} \in \mathbb{R}^L$:

$$\mathbf{e} = f\left(\sum_i w_i \mathbf{a}_i + b\right) \tag{2.3}$$

with f being a non-linear activation function. With the vector $\mathbf{e}$ we can then compute the region weights α_i by taking the softmax:

$$\alpha_i = \frac{\exp(e_i)}{\sum_j \exp(e_j)} \tag{2.4}$$

which creates relative weights α_i that sum to 1.

Finally, we can obtain the weighted representation of the feature vectors $\mathbf{a}_i$ by multiplying them with their respective weights:

$$\mathbf{z} = \sum_i^L \alpha_i \mathbf{a}_i \tag{2.5}$$

This model is smooth and differentiable and thus can be trivially learned using backpropagation.

2.4 Multitask Learning

Multitask Learning [3] (MTL) is a way to improve the learning of one task by learning another task in parallel, through shared representations. The hidden layers that the tasks share will hopefully have better-generalized representations due to a greater pressure on the parameters since the shared section of the model will be constrained to reach better values for both tasks.

The most common approach to MTL is to simply create a neural network where the lower layers are shared for both tasks and, on top of those shared representations, add task-specific layers that learn useful representations for each task. In order to train

a network, one only needs to backpropagate the errors of each task through the task specific parameters and sum the gradients for each task in the shared layers.

One of the most interesting properties of MTL models is the ability of a task to *eavesdrop* on another task. Consider that Task 1, T_1, and Task 2, T_2, both require a shared representation h. If T_1 has trouble in its learning algorithm to obtain good parameters h, T_1 can *eavesdrop* on T_2 through MTL. This way, T_1 improves its results thanks to a better representation of h.

Through this property and others, MTL models are capable of achieving better *generalization* - due to an improved statistical strength of the shared parameters thanks to the extra information of each task - and better *regularization*, since the shared parameters won't be as able to overfit to the statistics of a specific task.

Particularly useful to this dissertation is the fact that MTL does not constrain the size of the data sets used in each task. While the input may be the same, each task may have a different number of labels available or even none at all (unsupervised learning) - there is no constraint in the cost function used for each task.

This way, we will be able to use VQA-HAT in our MTL model, which uses only a subset of the whole VQA data set. Furthermore, the cost function will be different since the model will learn a distribution in the VQA-HAT task and a regular classification task with the VQA data set.

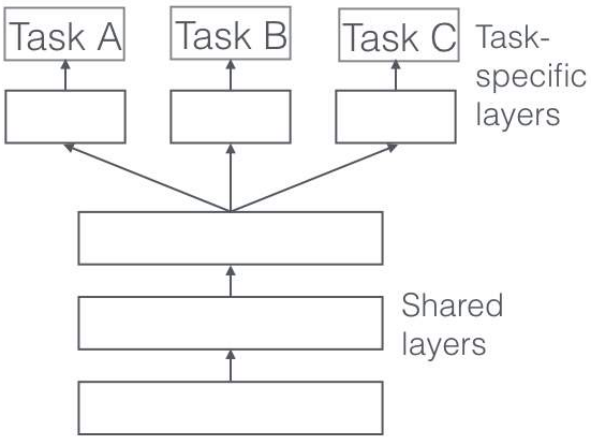


Figure 2.6: Diagram of a simple MTL model. The lower layers are shared across all tasks and the higher layers have parameters specific to their respective tasks.

Chapter 3

Related Work

In this section, we will review the most relevant models for this dissertation. We will especially focus on two state of the art deep learning methods: Stacked Attention Networks - for VQA prediction - and DeepFix - for saliency attention maps prediction. We will also go over some models that use human saliency to improve task performance.

3.1 Stacked Attention Networks

Stacked Attention Networks (SAN) [25] is a deep learning model for the VQA task. The model uses two layers of the visual attention mechanism (explained in the previous chapter) to allow the network to do multi-step reasoning.

The architecture of SAN is split into three main components:

1. The image feature extractor that obtains the vector representation v_I of the image through a CNN.
2. The question model that obtains a semantic representation v_Q of the question.
3. The stacked attention model that obtains a multi-step reasoning filtered representation of the question-image pair in order to obtain an answer.

A schematic of the overall model can be found in Figure 3.1. In the following sections, we will explain each of these three key components.

3.1.1 Image Model

To obtain the image features, the authors use VGGNet [42]. The features f_I are extracted by doing a forward pass on the pre-trained VGG-16 network up until the last

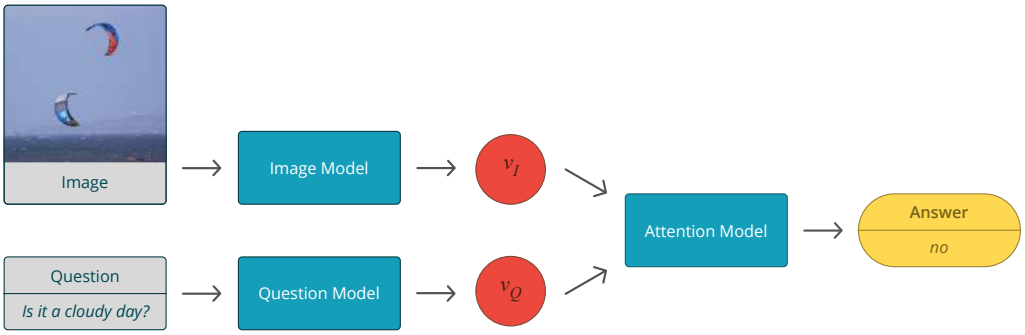


Figure 3.1: Schematic of overall architecture of the Stacked Attention Network.

pooling layer of the model (see Figure 3.2), obtaining 512 feature maps of 14×14 shape. Therefore, we are obtaining 512 features for each of the 196 (14×14) regions of the image. These region features correspond to 32×32 pixel regions of the 448×448 input image. The feature vector for each region is denoted by f_i , $i \in \{0, 195\}$.

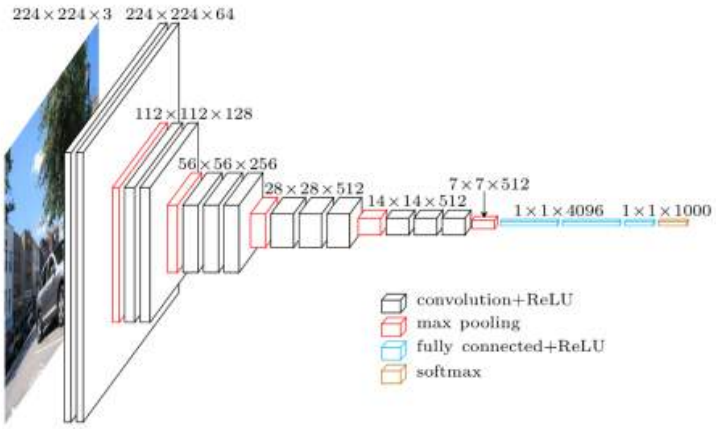


Figure 3.2: Diagram of the VGGNet, taken from [43]. The features extracted for f_I are the ones from the last pooling layer. In the SAN model, the ouput of this layer is instead of size $14 \times 14 \times 512$, since the input image size is $448 \times 448 \times 3$ instead of the $224 \times 224 \times 3$ depicted.

After obtaining these feature vectors, the image representation is brought to the dimensions of the question representation described in the next section, for modeling purposes, since we will sum both representations to obtain a combined representation. This can be done by a multilayer perceptron:

$$v_I = \tanh\left(W_I f_I + b_I\right) \tag{3.1}$$

where $v_I \in \mathbb{R}^{d \times m}$ is the image representation in the question vector dimensions with

v_i denoting the feature vector for each region i and d being the image representation dimension and m the number of regions.

3.1.2 Question Model

Both LSTMs and CNNs are used to obtain a question representation in this model. We will first go over how to extract semantic meaning with LSTMs and then with CNNs.

3.1.2.1 LSTM Question Model

Long Short Term Memory (LSTM) [37] is a neural network model capable of processing a sequence while preserving the hidden representations of each step of the sequence. It uses the hidden representation of the previous steps of the sequence, h_{t-1} and the current input step x_t to output the current hidden representation h_t. This is done by updating at each step a memory cell c_t through gate mechanisms that control how much the previous hidden representation and current input affect h_t.

This gate mechanism is composed of three gates:

- Forget gate f_t - controls how much the previous memory cell c_{t-1} is used to compute h_t.
- Input gate i_t - controls how much of the current input x_t is used to compute the current memory cell.
- Output gate o_t - controls how much of c_t is used to compute h_t.

Each of these gates are controlled by multilayer perceptrons with weights and biases learned through stochastic gradient descent. The non linear function used for the gates is a sigmoid since the output will then be in interval $]0,1[$, thus resembling a gate. However, for the memory cell and the output representation, tanh is used. The equations of the layers used for each block of the LSTM (computed for each step $t \in \{1,...,T\}$) are as follows:

$$i_t = \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + b_i\right) \tag{3.2}$$

$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + b_f\right) \tag{3.3}$$

$$o_t = \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + b_o\right) \tag{3.4}$$

$$c_t = f_t c_{t-1} + i_t \tanh\left(W_{xc} x_t + W_{hc} h_{t-1} + b_c\right) \tag{3.5}$$

$$h_t = o_t \tanh\left(c_t\right) \tag{3.6}$$

To obtain the full vector representation of the question $q = [q_1, ..., q_T]$, each word q_t is encoded as a one-of-V vector and then embedded into a vector space through the following embedding matrix:

$$x_t = W_e q_t \tag{3.7}$$

Then, at each time step $t \in \{1, ..., T\}$, we can compute the hidden representation h_t through an LSTM:

$$h_t = \text{LSTM}\left(x_t\right) \tag{3.8}$$

Thus obtaining the question vector $v_Q \in \mathbb{R}^d$ by taking the last hidden representation h_T.

3.1.2.2 CNN Question Model

Another alternative to obtain a semantic representation of the question is through a CNN similar to the one explained in section 2.3.1.1. As in the LSTM model, the authors obtain an embedding representation of each word in the question through equation 3.7 and then concatenate each word vector x_t to obtain the embedded 2D representation of the question:

$$x_{1:T} = [x_1, x_2, ..., x_T] \tag{3.9}$$

Then, the authors use kernels of size one, two and three (unigrams, bigrams and trigrams, respectively) to perform the convolution operation in the question embedding. That is, for each kernel of height $c \in \{1, 2, 3\}$ and width of the embedding dimension, a feature map h_c is obtained through computing the following multilayer perceptron:

$$h_{c,t} = \tanh\left(W_c x_{t:t+c-1} + b_c\right) \tag{3.10}$$

and concatenating the $h_{c,t}$ together:

$$h_c = [h_{c,1}, ..., h_{c,T-c+1}] \tag{3.11}$$

Note that W_c and b_c are the same for each $h_{c,t}$, maintaining the parameter tying property of CNNs.

Then, we perform a max-pool over all d feature maps h_c, thus getting one value per feature map of the number of unigrams, bigrams and trigrams kernels chosen. To obtain the question representation $v_Q \in \mathbb{R}^d$, we concatenate all of these values:

$$v_Q = \left[\tilde{h}_1, \tilde{h}_2, \tilde{h}_3\right] \tag{3.12}$$

where each $\tilde{h}_c \in \mathbb{R}^{d_c}$ is the output of the max-pooling of the feature maps of the kernels of window size c.

3.1.3 Stacked Attention Model

Finally, the Stacked Attention component of SAN is computed through k attention layers whose input is both the image feature vectors v_I and the question vector v_Q. Through an attention layer, the image features are filtered by weighting down noisy regions that do not help the network to answer the question. The goal of having multiple attention layers is to let the network do multi-step reasoning in order to answer the question - the image representation is gradually filtered in an effort to locate the most probable regions where the answer will most likely be found.

Formally, the image and question representations go through the following multilayer perceptrons before going into the attention layer k:

$$h^k_{A,I} = \tanh\left(W^k_{I,A} v_I + b^k_{I,A}\right) \tag{3.13}$$

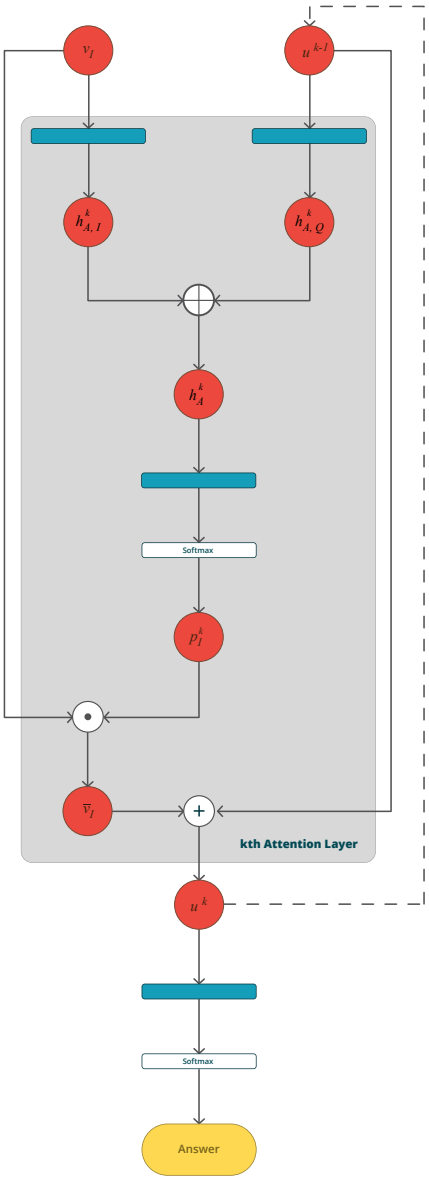


Figure 3.3: Diagram of the attention model of SAN.

$$h_{A,Q}^k = \tanh\left(W_{Q,A}^k u^{k-1} + b_{Q,A}^k\right) \tag{3.14}$$

where $W_{I,A}^k$, $W_{Q,A}^k \in \mathbb{R}^{n\times d}$, with n being the number of attention features used to obtain the kth attention layer representation h_A^k, and u^{k-1} being the query vector computed in equation 3.18 where u^0 is initialized to v_Q. The representations $h_{A,Q}$ and $h_{A,I}$ are summed to obtain h_A^k:

$$h_A^k = h_{A,Q} \oplus h_{A,I} \tag{3.15}$$

This representation can then go through another feedforward layer and through a softmax function to obtain the probability attention vector p_I^k:

$$p_I^k = \text{softmax}\left(W_P^k h_A^k + b_P^k\right) \tag{3.16}$$

where $W_P^k \in \mathbb{R}^{1\times n}$ and $p_I^k \in \mathbb{R}^m$. We are now ready to filter the image vectors by computing the deterministic attention:

$$\tilde{v}_I^k = \sum_i p_i^k v_i \tag{3.17}$$

Finally, we compute the query vector u^k by doing the following:

$$u^k = \tilde{v}_I^k + u^{k-1} \tag{3.18}$$

After repeating the process K times, the answer can be obtained by using the final u^K:

$$p_{\text{ans}} = \text{softmax}\left(W_u u^K + b_u\right) \tag{3.19}$$

The index of the maximal value of p_{ans} is the inferred answer of the model in the one-of-V encoding of possible answers. See Figure 3.3 for a schematic overview of the Stacked Attention Model.

3.2 Using Human Saliency in Models

The incorporation of human gaze in Machine Learning models allows an additional supervisory signal to the model. Gaze information helps the machine by sharing part

of the image understanding processing, since it can make the machine look more efficiently at the most useful sections of the image. Gaze assisted models have been used successfully in object recognition and localization [44, 11].

More closely related to this dissertation, Sugano *et al.* used a gaze assisted model to improve performance in the task of image captioning [45]. The authors used a gaze feature $\mathbf{g} = \{g_1, ..., g_L\}$ represented by a normalized histogram of the L regions of the image. This gaze feature is incorporated in the attention model described in section 2.3.2.

Instead of learning the attention representation through equation 2.3, the gaze feature is added to this equation by learning weights for the fixated and non-fixated regions in the images through the following equation:

$$e_{t,i} = g_i \mathbf{w}_{\text{pos}} \mathbf{p}_{t,i} + (1 - g_i) \mathbf{w}_{\text{neg}} \mathbf{p}_{t,i} + b \tag{3.20}$$

where $\mathbf{p}_{t,i}$ is a nonlinear projection of the image features $\mathbf{a}_i$ of region i and the previous hidden state $\mathbf{h}_{t-1}$ of the LSTM that is used to generate each work of the resulting caption. This way the model can efficiently use the most relevant gaze features while also not losing information about the non-fixated regions of the human gaze that can still be useful.

The authors performed experiments in the above model (named *split attention*) using the SALICON data set [12] for the gaze features. The model was compared with the original and machine-only attention model, with a model where the attention model was simply replaced by the gaze features and another where the non-fixation weights ($\mathbf{w}_{\text{neg}}$) were not used. Only the split attention model was found to be useful to improve the baseline of machine-only attention based image captioning.

Even though the split-attention model improved the image captioning performance of the baseline, the fact that the gaze features from the SALICON data set need to always be used to produce results - even when at test time - is quite lacking from a practical standpoint since human attention will not always be available.

3.3 Predicting Saliency with DeepFix

Since gaze information may not be available at test time, it is crucial to be able to correctly predict human fixation if we want to efficiently integrate it in a model without the need of constant human input. The neural network model DeepFix [46] is a state of

the art model at predicting human attention at the time of the writing of this dissertation and will be explained in this section.

Inspired by the success of deep neural networks in a variety of tasks, Kruthiventi *et al.* created the DeepFix model by slightly modifying the architecture of VGG net and adding layers to it that extract the features needed to correctly predict human fixation. The whole model is composed of 20 convolutional layers and its diagram can be found in Figure 3.4.

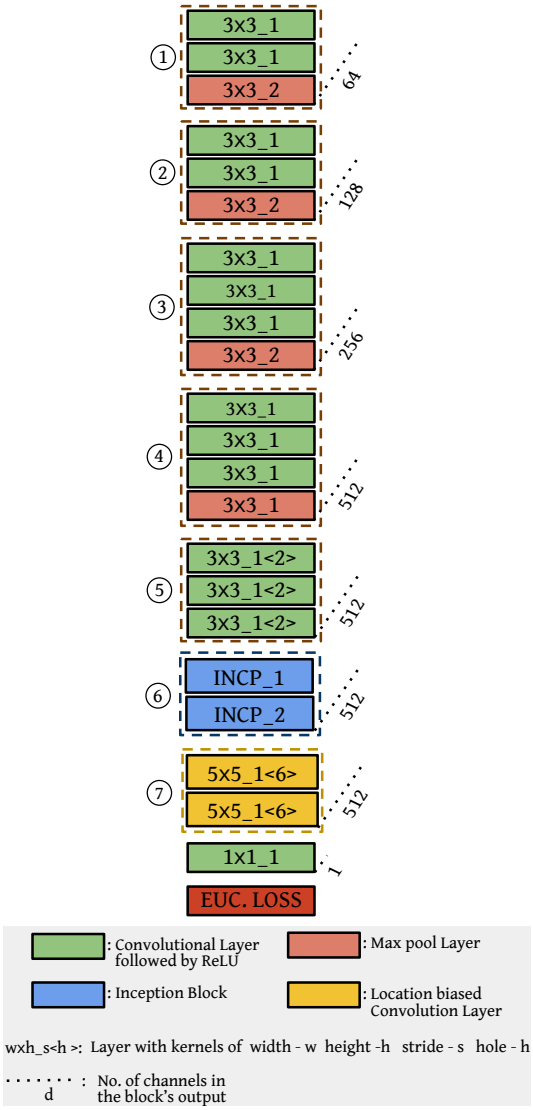


Figure 3.4: Diagram of the architecture of DeepFix, taken from the original paper [46].

The model can be described via 7 blocks. Blocks (1) to (5) are initialized to VGG-16 and resemble its architecture for the most part. The main difference can be found in block (5) where the layers now have *dilated kernels*, making the layer an *atrous*

convolution [47].

In an atrous convolution, the kernels used have *holes* between the parameters, which can be useful in order to have a convolutional layer with a larger kernel but with the same efficiency of a smaller kernel. In the case of block (5), this means the parameters used are from a 3×3 kernel but the actual size of the kernel will be 5×5. The rule to obtain the actual kernel size $\hat{k}$ from the original kernel size k and the number of holes h is:

$$\hat{k} = k + (k-1)(h-1) \tag{3.21}$$

and a visualization of an atrous convolution can be found in Figure 3.5.

In block (6), Kruthiventi *et al.* implemented two inception blocks, first introduced in GoogLeNet [48]. An inception module is used to capture multi-scale semantic features since it is composed of several different convolutional layers of different kernel size parallel to each other whose outputs are concatenated. Similar to the modules found in GoogLeNet, the inception block in DeepFix has 4 parallel paths of computation, using 1×1 and 3×3 convolutions and max-pool layers. However, instead of a 5×5 convolution in one of the paths, DeepFix uses a 3×3 kernel with holes of size 2, simulating the original 5×5 kernel. A diagram of this inception module can be found in Figure 3.6.

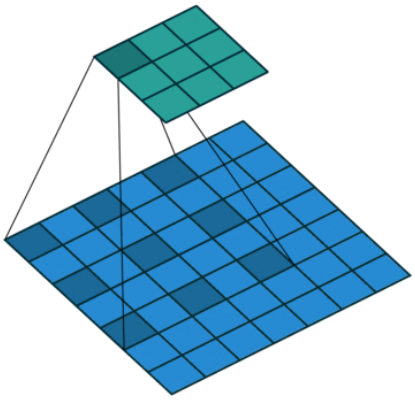


Figure 3.5: Visualization of the atrous convolution, taken from [49].

Finally, in block (7), the authors introduce *Location Biased Convolutions* (LBC). These layers are similar to the regular convolutional layers but add a location specific feature $\mathbf{L}(x,y)$ to the output feature map computation. This location specific input is composed of 16 2D Gaussians with centered mean, each with different values for

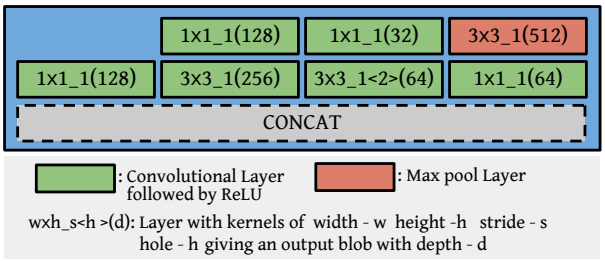


Figure 3.6: Sub-architecture diagram of the inception block in DeepFix, taken from the original paper [46].

vertical and horizontal variance. While in normal convolutions the feature maps R_c are obtained in the following way:

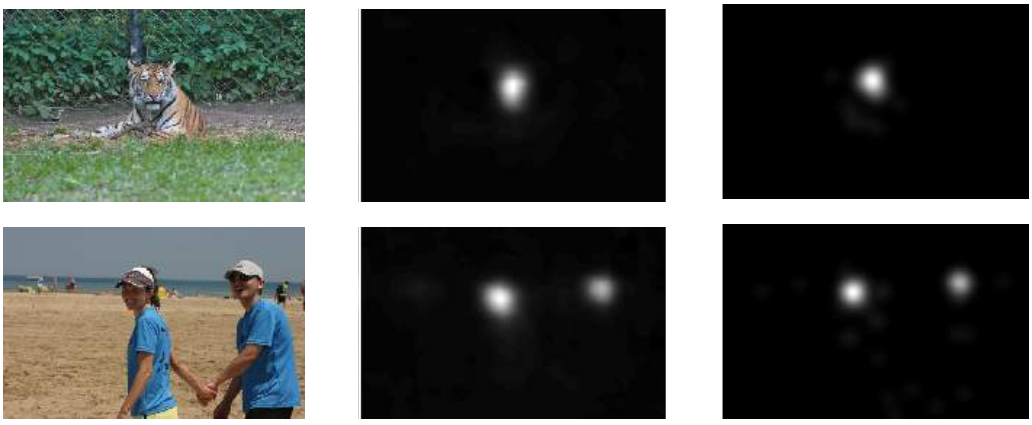
$$R_c(x,y) = f(\mathbf{I} \ast \mathbf{W_c} + b_c) \tag{3.22}$$

In LBC, a second set of weights W'_c is added to the computation, whose role is to learn the center bias of the saliency prediction:

$$R_c(x,y) = f\left(\mathbf{I} \ast \mathbf{W_c} + \mathbf{L} \ast \mathbf{W'_c} + b_c\right) \tag{3.23}$$

Furthermore, the kernels of block (7) have kernels of size 5×5 with holes of size 6, making the actual kernel size 25×25. These kernels have such large size in order to capture features through examination of a large neighbor context, since it has been found that, in humans, the most salient objects are characterized by how unique they look in the context of their nearby objects [50]. After the second LBC layer of block (7), a dropout layer is added with a dropout ratio of 0.5 to avoid overfitting.

After the 7 blocks, the feature maps go through a 1×1 convolutional layer whose output is up-sampled to the original size of the input image and compared to the actual human saliency map. Figure 3.7 shows prediction examples from the model, along with their respective correct human saliency maps.



(a) Original Image (b) DeepFix Prediction (c) Human Attention

Figure 3.7: Examples of attention predictions with DeepFix and their respective ground truths of human attention, taken from the original paper [46].

Chapter 4

Methodology

In this chapter, we will go over the methods used in this dissertation to improve the correlation of the attention mechanism in a VQA model with human attention data and how this can improve the performance of our VQA baseline.

First, we will describe the data sets used in this dissertation and how to preprocess the human attention maps of VQA-HAT. Then, we will explain how to efficiently integrate a cost function for the human attention maps in a visual attention model and how this can be used to train a neural network in a Multitask Learning setting. Subsequently, we will go over how our baseline will be expanded to contain a smaller version of DeepFix in order to improve the performance of saliency prediction of the model. We will also explain a way to combine the human saliency prediction with the unsupervised machine attention. Finally, the evaluation methods of the VQA task and of the saliency prediction task will be described.

4.1 Data Sets

Two data sets were used in this dissertation - the VQA data set and the VQA-HAT data set. They are described below.

4.1.1 VQA Data Set

The VQA data set [1] is composed of images, questions and multiple possible answers to each question. The images of the VQA data set are a subset of the MSCOCO data set [51], containing in total $204,721$ images. Each image has 3 questions and each question 10 possible answers gathered by 10 different subjects, which sums a total of

614,163 questions and 7,984,119 answers. Only the top 1,000 answers are used as possible categories in our model, which represents 82.67% of the possible answers in the data set. An example of this data set can be found in Figure 2.2.

4.1.2 VQA-HAT Data Set

The VQA-HAT data set [2] is composed of grayscale images that are correlated with human attention where white regions correspond to the fixations of the human gaze in the image and the black regions the non-fixations.

The data was gathered by showing subjects of Amazon Mechanical Turk a blurred image of the VQA data set along with one of the possible questions and its most frequent answer. The subjects could then sharpen the blurred image in the regions they thought it would make them answer the question correctly.

This experiment resulted in a total of 58,475 attention maps for the question-image pairs of the VQA training set and 4,122 attention maps for the question-image pairs of the VQA validation set.

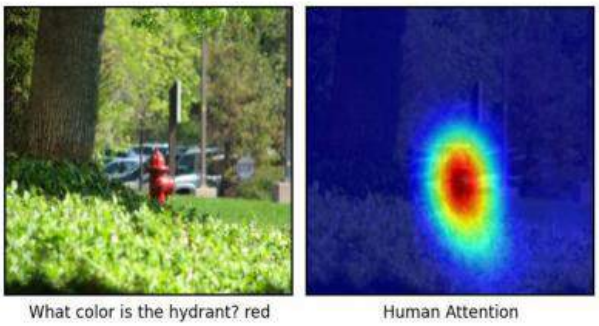


Figure 4.1: Example of the VQA-HAT data set. On the left side, an image, question and answer from the VQA data set and on the right side, a human attention map from VQA-HAT [2].

4.2 Human Attention Preprocessing

The human attention maps of the VQA-HAT data set are grayscale images of different widths and heights. In contrast, the attention representation of our VQA baseline is the output of a softmax of a layer with 196 units.

In order to represent the human attention maps in a way that can be compared to the output of this softmax, the grayscale images are resized to a 448×448 shape and

then down-sampled by local averaging by a factor of 32. This results in a 2D array of shape $14 \times 14 = 196$. Finally, the maps are flattened to 1D and normalized to resemble a distribution like the output of the attention model. In the training set, some image-question pairs had more than one possible human attention map. In these cases, we took the average of the flattened vectors and only then did we normalize the resulting vector. In the validation set, all attention maps were used to evaluate the model.

It is important to note that 125 attention maps were ignored from the training set of VQA-HAT. These maps had the same value for every pixel and thus would bring issues in training and in evaluation metrics that compute standard deviation.

4.3 Using Multitask Learning to improve the SAN model

The main contribution of this dissertation is a method to bring attention based VQA models to a closer resemblance to the way human saliency works. In a different approach to the work mentioned in section 3.2, we argue that the best way to improve the correlation of machine attention and human attention while improving VQA accuracy is through making the model learn to develop representations that improve both tasks at the same time. This model does not assume the availability of human attention data at test time, which is more accurate in a real-world scenario. This hypothesis can be fulfilled by using a Multitask Learning setting in an existing VQA model.

As mentioned, this existing model will be SAN. The model uses k attention layers to obtain attention representations that filter the image in order to obtain a less noisy representation of it to increase the probability of getting the correct answer to the question. In the last attention layer K, the query representation u^K goes through a classification layer that obtains a distribution of probabilities $\bar{\mathbf{y}}$ of dimension V (V being the number of possible answers), where

$$\hat{i} = \arg\max_i(\bar{y}_i) \tag{4.1}$$

is the index to the predicted answer. This probability vector can be compared to the 1-of-V representation of the correct answer $\mathbf{y}$ through a computation of the cross-entropy loss L_{CE}:

$$L_{CE}(\mathbf{y}, \bar{\mathbf{y}}) = -\sum_i y_i \log \bar{y}_i \tag{4.2}$$

which can then be backpropagated through the network.

On the other hand, let's denote the target human attention map distributions by z. These target distributions have the same dimensions as the attention representation of the kth attention layer of SAN. We will assume that the human attention maps of VQA-HAT represent the last step of a multi-step reasoning to single out the most relevant regions of the image and so we will compare the target human attention map distribution $\mathbf{z}$ to the predicted attention map distribution $\bar{\mathbf{z}}$ of the last attention layer K of SAN. The loss between these two distributions can be computed through the forward Kullback–Leibler divergence ℓ_{KL}:

$$\ell_{KL}(\mathbf{z}, \bar{\mathbf{z}}) = \sum_i z_i \log \frac{z_i}{\bar{z}_i} \tag{4.3}$$

The introduction of an extra cost function in an existing layer of the model is in contrast to most MTL literature where there are shared representations between the tasks in lower layers and top layers are task specific. In our MTL setting, the task-specific layers are only available to the answer prediction task while all the representations used in the human saliency prediction task are shared between the tasks. This modeling choice is crucial to our hypothesis since our goal is to bring the machine attention representation closer to the human attention and thus it would not make sense to have task-specific layers for the saliency prediction - the human saliency prediction *itself* is the probability vector used to filter the image representation in attention layer K.

In order to train this MTL model, we choose to use a joint cost function that combines the loss of the answer prediction, L_{CE}, and the loss of the saliency prediction, ℓ_{KL}. However, as stated in previous sections, the VQA-HAT data set is a subset of the VQA data set, which poses a problem in computing a cost function that can account for this imbalance of data. We overcome this issue by training our model with carefully constructed mini-batches of size N where M of those samples are drawn from the subsection of the VQA data set that contains VQA-HAT attention maps. The remaining $N-M$ examples are drawn from the subsection of the data set that only has image-question pairs and answer labels. This way, for each batch the following cost function can be iteratively minimized through gradient descent:

$$\text{Batch Cost} = (1-\lambda)\frac{1}{N}\sum_{i=1}^{N} L(\mathbf{y}_i, \bar{\mathbf{y}}_i) + \lambda \frac{1}{M}\sum_{j=1}^{M} \ell\left(\mathbf{z}_j, \bar{\mathbf{z}}_j\right) \ , \quad 0 < \lambda < 1 \tag{4.4}$$

where the hyperparameter λ can be tuned to change how much the human saliency prediction task affects the loss computation when compared to the answer prediction

task. Note that each of the losses combined in this joint cost is averaged across all the examples available in the batch for the task, which makes this combination of losses a fair one, even if the human saliency task has fewer examples. The number of examples drawn out of VQA-HAT when compared to the size of the whole mini-batch is controlled through the hyperparameter r, where:

$$M = \lfloor r \cdot N \rfloor , \quad 0 < r < 1 \tag{4.5}$$

This cost function can be smoothly integrated into any model that uses a soft attention layer - not only SAN.

4.4 Expanding SAN with DeepFix

In a further effort to bring the attention prediction of our VQA model closer to the way humans look at images, we extend the computation of the attention probability vector of the Kth attention layer of SAN by swapping the deepforward layer used in SAN by a modified version of blocks (6) and (7) of the DeepFix model described in section 3.3.

These blocks correspond to the section of DeepFix after the layers that resemble VGG net. In SAN, we already extracted the features of VGG up until the last max-pooling layer. This has a slight contrast to the section of VGG the DeepFix model uses, since not only DeepFix fine-tunes the pre-trained VGG while SAN just extracts these features, but also DeepFix has modified convolutions in block (5) (see Figure 3.4) and does not down-sample the features with a max-pool layer after this block (see Figure 3.2). However, we do not consider these modifications to have a great impact on the attention representation and we will simply use the extracted VGG features of the last max-pooling layer as the original SAN model does.

DeepFix is a model that excels in bottom-up attention prediction. This is in opposition with attention in the task of VQA - there are cognitive factors in play in this task since the information encoded in the question will affect the region in the image that will be attended. Accordingly, blocks (6) and (7) of DeepFix will be introduced in place of the feedforward layer described in equation 3.16, in the attention layer where $k = K$ - the last attention layer of SAN. This way, instead of using as input to block (6) the output of the VGG features, we are instead using the representation h_A^K, which is a combined representation of the image and query, already filtered by previous attention

layers. We argue that this representation is a better-suited input to our DeepFix attention prediction since it contains the cognitive information of the question combined with the bottom-up factors of the image.

Therefore, it is expected that the representation h_A^K has high-level semantic information that can be used to make the probability distribution of attention concentrate on the best regions of the image. As argued in [46], it has been shown that saliency information is best captured when these high-level semantic features are considered from several scales [52, 53] and thus we will use the inception block (6) of DeepFix almost exactly as described in section 3.3. The main difference between our approach to this block and DeepFix's approach is that, due to time and computing resources, we will just use 32 kernels which result in an output of 32 feature maps.

In block (7) of DeepFix, kernels of 5×5 with holes of size 6 are used, which result in an actual kernel size of 25×25. The input size of this block are feature maps of $512 \times 58 \times 58$. However, the representation that comes out of the inception modules in our model is instead $32 \times 14 \times 14$. These smaller input dimensions make the use of an actual kernel size of 25×25 no longer reasonable since the kernel dimensions would be bigger than the input dimensions. For this reason, we modify the convolutions used in this layer to kernels of 3×3 with holes of size 3, resulting in an actual kernel size of 7×7. This modeling choice was made from the fact that the relation between a kernel of 7×7 to an input of 14×14 was the closest viable option to the relation between a kernel of 25×25 with an input of 58×58, since $\frac{25}{58} \approx \frac{7}{14} \approx 0.5$ [1].

Furthermore, the original block (7) uses LBC layers, where a location bias to the center is learned in order to improve the performance of the model since the most salient objects in a picture are most likely to be present in the center due to photographer's bias [54]. However, we argue that, while this bias exists for bottom-up saliency, top-down saliency may not be as affected by it since the cognitive information of the question can lead the attention mass to regions completely outside the central region of the image in order to answer a question correctly and such location bias could make it difficult for the model to look at these outside regions when needed. For this reason, we remove this bias from out model in block (7), using instead regular convolution layers. This result was observed in our model explorations - adding a location bias to these convolutional layers would only add unnecessary complexity to our model since the evaluation results would stay the same or worsen.

[1] Although a 6×6 kernel would make the kernel-input size ratio closer to the original ratio, the size of the kernel needs to be odd to obtain an integer number for zero-padding so the output size of the layer

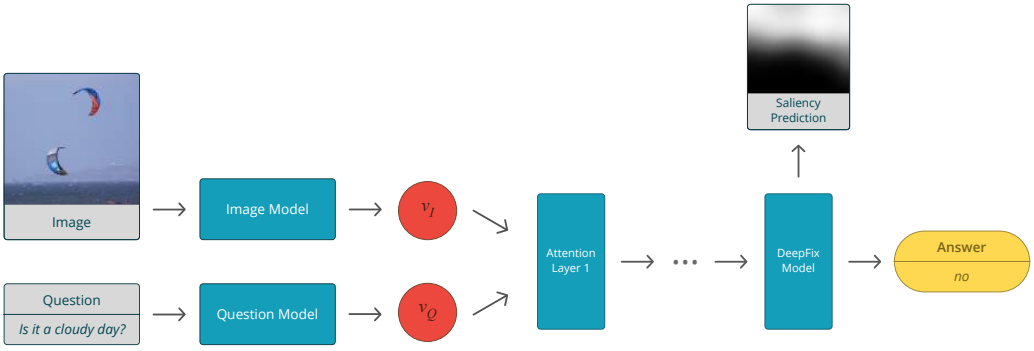


Figure 4.2: Diagram of the MTL SAN+DeepFix model.

As in DeepFix, after block (7), we add a dropout layer to avoid overfitting of the attention maps and then a 1×1 convolution layer is used to obtain the representation that goes through a softmax function to obtain the final probability distribution of attention. This model can also be easily trained through the cost function described in the previous section using gradient descent and can be used in place of any soft attention layer of any model other than SAN. For a diagram of the model used in this section, see Figure 4.2 and Figure 4.3 for the changes in the DeepFix architecture.

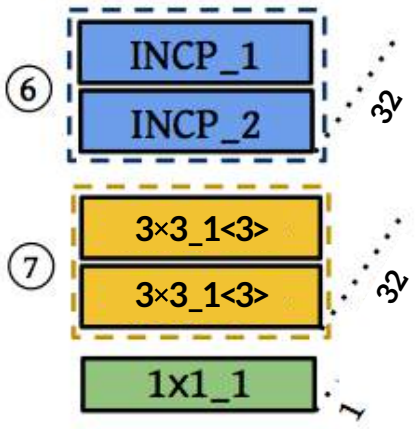


Figure 4.3: Diagram of the section of Deepfix used in our models and its modifications.

4.5 Machine and Human Attention Combination

Additionally, we will also explore the possibility of using task-specific layers for the saliency prediction task and combining the machine attention prediction with the human attention prediction instead of simply replacing the former with the latter.

remains 14×14.

In order to do this, we use the same modified DeepFix layers as described in the last section using $\mathbf{h}_A^K$ as input and compute the human saliency prediction, $\mathbf{g}_I$. However, in this model, this probability distribution does not replace the distribution used to filter the image features for the Kth time - it will instead be used to learn the best way to combine its encoded information with the SAN machine attention representation. We follow [45], and combine both attentions as explained in section 3.2:

$$\mathbf{p}_I^K = \text{softmax}\left(\mathbf{g}_I W_{P,\text{pos}} \mathbf{h}_A^K + (1 - \mathbf{g}_I) W_{P,\text{neg}} \mathbf{h}_A^K + b_P^K\right) \tag{4.6}$$

In contrast to what was presented in section 4.3, this model has task-specific layers for the saliency prediction task. A diagram of the model can be observed in Figure 4.4.

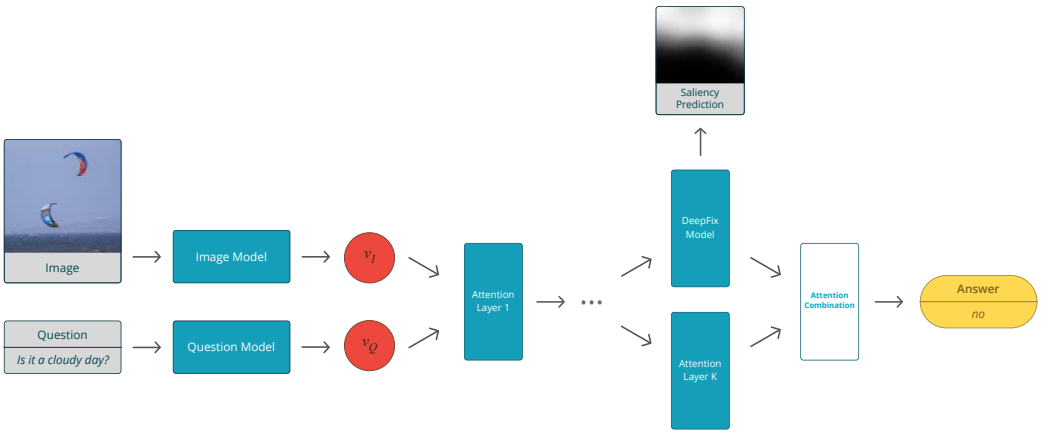


Figure 4.4: Diagram of the MTL Split Attention model.

4.6 Evaluation

In this section, we will describe the evaluation metrics used for the models used in this dissertation.

4.6.1 VQA Accuracy

The evaluation metric of the VQA task is the accuracy of the predicted answer. In the VQA data set [1], there are 10 human answers gathered from distinct subjects and the accuracy of a generated answer is computed by:

$$\text{Answer Accuracy} = \min\left(\frac{\#\text{ human answers equal to provided answer}}{3},\ 1\right) \tag{4.7}$$

This means that if there are 3 humans or more that provided the same answer, the answer is considered 100% correct. The average over all answer accuracies in the test set is then computed to obtain the test set accuracy.

4.6.2 Rank Correlation

On the other hand, the saliency task is evaluated by computing Spearman's Rank Correlation Coefficient. It is a common metric to evaluate saliency which computes the correlation between two variables X and Y by calculating the Pearson's Correlation Coefficient,

$$\rho = \frac{\text{cov}(r_X, r_Y)}{\sigma_{r_X}\sigma_{r_Y}} \tag{4.8}$$

over the ranked variables r_X and r_Y, that is, before computing the coefficient, the values in the arrays for each variable are assigned an integer that corresponds to how that value ranks compared to other values in the variable. These ranked variables are the ranked versions of the array with the pixels values of the attention maps that are being compared - the model's predicted attention maps, X, and the human attention maps, Y.

This results in a nonparametric metric for the correlation between two variables, since a coefficient value of 1 signifies a perfect relation between the variables by *any* monotonic function. This is in contrast with Pearson's Correlation Coefficient, which can only measure the linear correlation between the variables.

Each model's resulting attention maps are compared to the test set human attention maps through this metric and then the average over all coefficients is taken to obtain the test set evaluation of the saliency prediction.

4.6.3 Generalized Variance of a Distribution

Finally, we will evaluate each model's attention maps alone by computing their average generalized variance [55]. The generalized variance (GV) is defined as the determinant of the covariance matrix of a distribution, $|\Sigma|$. The larger GV is, the more dispersed is the distribution, which makes GV a good measure of the 2-dimensional spread of the attention maps of each model. In fact, the determinant of a matrix is the product of its eigenvalues:

$$|\Sigma| = \prod_i \lambda_i \tag{4.9}$$

and since we are computing the determinant of the covariance matrix Σ, the eigenvalues correspond to the variances along the principal components of the distribution and thus GV measures the product of these principal variances. This results in a way to measure the area occupied by the most selected regions of the attention map.

In the specific case of our 2D attention maps, we will assume the maps are in the space $\mathbb{R}^{[0,1]\times[0,1]}$ and compute the square root of GV, $\sqrt{|\Sigma|}$, after computing the respective 2×2 covariance matrix Σ of the attention map distribution. This metric will be used to compare how the GV of the attention maps affect the VQA accuracy, more specifically, if a lower GV improves VQA accuracy, that is, if a more confident attention model improves the answer accuracy of the overall model.

Chapter 5

Experiments and Results

In this section, we will show the results obtained for the models described in the previous chapters. An overview of those models and their baselines will be done, followed by the description of important hyperparameters choices and data splits. The validation and test set results are then showed and a discussion and error analysis of the models ensues.

5.1 Models and Baselines Overview

As explained in the previous chapter, we explore three main models in this dissertation:

- **MTL SAN** - the SAN model with an additional supervision in the last attention layer.
- **MTL SAN+DeepFix** - the SAN model where the last attention layer is replaced with the last layers of DeepFix.
- **MTL Split Attention** - the SAN model where parallel to the last attention layer prediction of machine attention, the last layers of DeepFix also predict a human attention distribution and both are combined.

All three of these models are trained through an MTL setting with a joint cost, explained in section 4.3.

To observe how these models compare to state-of-the-art results, the following baselines are also evaluated:

- **SAN** - the original SAN model as described in section 3.1. This will serve as a baseline for the VQA task.

- **DeepFix** - a baseline for the human saliency prediction. For a better comparison with our models, this baseline does not fine-tune the VGG-16 parameters like the original paper but instead just uses the features extracted from the last pooling layer like in SAN. Furthermore, like explained in section 4.4, blocks (6) and (7) of DeepFix are also modified in the same way in this baseline.

- **Human Agreement** - important upper bound of human attention prediction. It is essential to check how much humans agree with each other on what regions of the image are important to answer a question for the same image-question pair.

- **SAN+DeepFix** - the model of section 4.4 but without human attention supervision, i.e., $\lambda = 0$ in equation 4.4.

- **Uniform Distribution** - a baseline (upper-bound) for the GV metric described in section 4.6.3. It corresponds to the GV of the attention map that selects the entire image as relevant.

5.2 Experimental Details

All models were trained in the same random seed for data shuffles and parameter initialization. The parameter initialization was performed as in [25]. Training was performed with stochastic gradient descent using a learning rate of 0.1 and momentum of 0.9. In contrast to [25] and [46], no L2 regularization was used since we found it would lead to better results in our validation. When using SAN, we chose $K = 2$ as the number of attention layers and only the CNN Question model was used to obtain the question representations. Regarding our MTL models, we choose $r = 0.23$ (approximately the percentage of VQA-HAT image-question pairs when compared to the whole VQA) and $\lambda = 0.2$, the only exception being the baseline SAN+DeepFix where $\lambda = 0$.

Regarding data splits, we follow [25] in using a section of the validation to train as well in order to have comparable results to the original paper. We denote this validation section used in training as `val1` and the other validation section to locally evaluate VQA accuracy as `val2`. However, while the VQA task has a test set besides validation, VQA-HAT only provides training and validation attention maps. For this reason, only the training section of VQA-HAT is used to train the saliency task, while `val1` is used as a validation set and `val2` as a test set.

5.3 Results

The results for the several models are shown in the following tables. Some baselines will only be shown in tables where they are relevant. Table 5.1 shows results for VQA accuracy in the validation set `val2` and Table 5.2 for the test set. We also show accuracies for question categories in these tables. Tables 5.3 and 5.4 show the correlation results for the validation set `val1` and the test set `val2`, respectively. Each of these tables show as well the average generalized variance of each model's attention maps.

Table 5.1: Validation set VQA accuracy results (`val2`).

		Accuracy Per Answer Type (%)		
Model	**Overall Accuracy (%)**	**other**	**number**	**yes/no**
SAN	59.36	49.10	36.52	78.89
MTL SAN	59.51	49.24	37.24	78.87
SAN+DeepFix	60.00	49.94	37.61	79.15
MTL SAN+DeepFix	59.99	50.29	36.56	79.03
MTL Split Attention	59.42	48.98	36.74	79.11

Table 5.2: Test set VQA accuracy results (`test-dev2015`).

		Accuracy Per Answer Type (%)		
Model	**Overall Accuracy (%)**	**other**	**number**	**yes/no**
SAN	56.09	42.51	35.00	77.59
MTL SAN	56.13	42.47	35.12	77.69
SAN+DeepFix	57.09	44.03	35.78	77.81
MTL SAN+DeepFix	56.99	44.21	33.70	77.88
MTL Split Attention	56.15	42.39	33.84	78.18

5.4 Discussion and Error Analysis

From the results presented in the tables, we can observe that our hypothesis and goal - to build a model able to mimic human attention behavior while successfully doing the VQA task - was achieved.

We can observe that the VQA accuracy of our baseline SAN is roughly the same to our first model MTL SAN, where the last attention layer of SAN has the additional

Table 5.3: Validation set (`val1`) human rank correlation (higher is better) and generalized variance (lower is better) results.

Model	Human Rank Correlation	$\sqrt{\lvert\Sigma\rvert}$
SAN	0.164 ± 0.005	0.064 ± 0.001
DeepFix	0.589 ± 0.006	0.059 ± 0.000
MTL SAN	0.381 ± 0.006	0.076 ± 0.001
SAN+DeepFix	0.181 ± 0.005	0.051 ± 0.001
MTL SAN+DeepFix	0.574 ± 0.007	0.039 ± 0.001
MTL Split Attention	0.180 ± 0.004	0.040 ± 0.001
Human Agreement	0.623 ± 0.003	0.071 ± 0.000
Uniform Distribution	–	0.096 ± 0.002

Table 5.4: Test set (`val2`) human rank correlation (higher is better) and generalized variance (lower is better) results.

Model	Human Rank Correlation	$\sqrt{\lvert\Sigma\rvert}$
SAN	0.164 ± 0.005	0.065 ± 0.001
DeepFix	0.598 ± 0.006	0.058 ± 0.002
MTL SAN	0.395 ± 0.006	0.076 ± 0.001
SAN+DeepFix	0.192 ± 0.005	0.048 ± 0.001
MTL SAN+DeepFix	0.590 ± 0.006	0.038 ± 0.001
MTL Split Attention	0.183 ± 0.004	0.041 ± 0.001
Human Agreement	0.623 ± 0.003	0.071 ± 0.000
Uniform Distribution	–	0.096 ± 0.002

supervision of the human attention maps. There is only a slight increase in accuracy of $+0.15\%$ on the validation set and even smaller on the test set ($+0.04\%$) -thus making it unclear if this increase remains over different random seeds and test sets or if it is just noise. Nonetheless, there is some increase in Human Rank Correlation ($+0.231$).

However, we can find a large increase in human correlation along with an increase in VQA accuracy with the model MTL SAN+DeepFix. This model is able to fulfill the hypothesis of having a model that is able to mimic human behavior and learn the task of VQA. We can observe that this model obtains a human correlation of 0.590 in the test set while the human agreement between subjects is 0.623, having only a difference of 0.033 between each other, which leads us to conclude that this model is incredibly close to mimic human saliency in the VQA task.

Regarding the VQA accuracy of this model, we can observe an increase of approximately $+0.9\%$ when compared to the baseline. This result is observed as well in the unsupervised attention model of SAN+DeepFix, which means the increase in accuracy may be due to the layers of DeepFix included in the attention model and not necessarily a product of the increase in human rank correlation. Nevertheless, this unsupervised attention model, that uses DeepFix instead of a regular attention layer, has an increase in human rank correlation when compared to the baseline. This result strengthens our belief that models that aim to mimic human behavior will obtain better performances, even if the model does not necessarily learn from humans - it just needs a model that resembles the neural reasoning of humans doing the task.

An interesting result can be observed in the VQA accuracies per answer type - there is an increase of almost 2% in the *others* category in the MTL SAN+DeepFix and SAN+DeepFix model when compared to the remaining models. A slight increase in MTL SAN+DeepFix of 0.2% to 0.4% when compared to SAN+DeepFix can also be observed. This category typically has questions that require an answer that is referring to something specific in the image (e.g.: "What is the man holding?" requires that the answer is the name of the object the man is supposedly holding). This is likely due to the better attention maps with higher human correlation, making the models look into the correct regions of the image.

It is also worth noticing the results of the MTL Split Attention model. In this model, the supervised DeepFix attention prediction was learned in parallel to the final unsupervised machine attention prediction and then combined through a layer of parameters that would learn the best way to combine both attention maps. The results on the VQA task are very similar to the MTL SAN model, having only a very slight in-

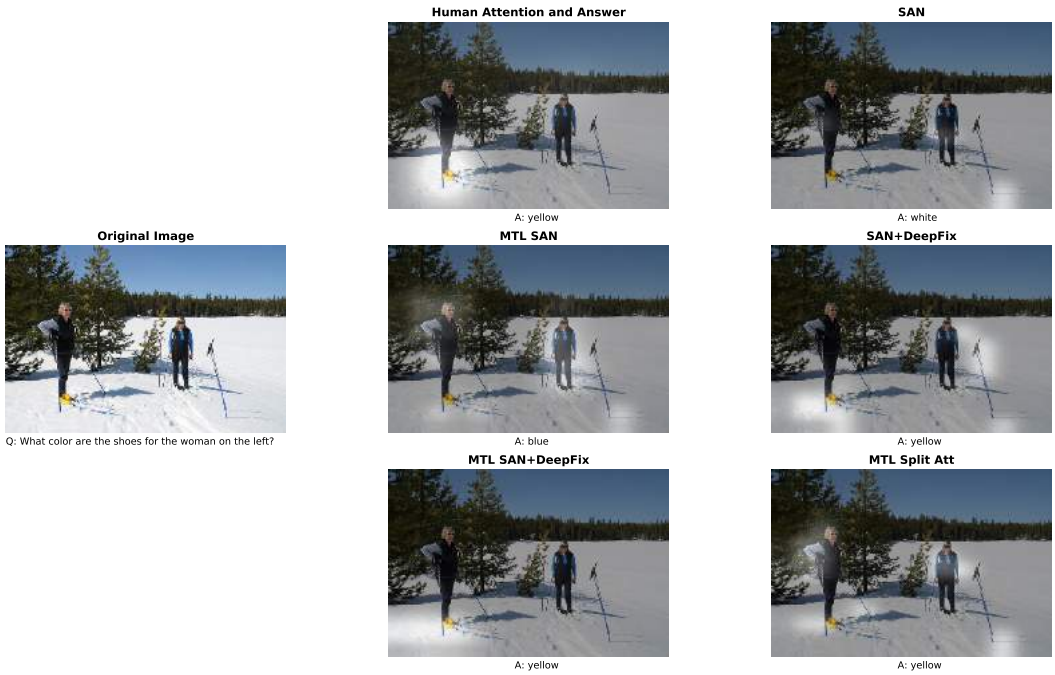
crease of accuracy that is likely insignificant. Furthermore, the results on the saliency task show a correlation much closer to the baseline results than the DeepFix results, which suggests that the model learned to not include much of the human attention map prediction into the combined human-machine attention, which led to a performance in VQA closer to the baseline.

We can observe in Figure 5.1 examples of the attention maps from the models presented in this dissertation along with their answers (more examples can be found in Appendix A). These attention maps were obtained by up-sampling the output of the final attention layer to the size of the original image and smoothing it. In this examples, we will mostly focus on analyzing the attention maps of the MTL SAN+DeepFix model when compared to SAN+DeepFix, the baseline SAN and the ground truth attention map and answer.

We can see in Figure 5.1a that the baseline answers the question incorrectly and that its respective attention map is not focusing on the right region of the image. In contrast, MTL SAN+DeepFix model focuses on the correct region of the image, which in turn leads to a correct answer. However, we can also see that the SAN+DeepFix model answers correctly and looks at the right region - but it does so while also focusing on regions that are not helpful to answer the image.

In Figure 5.1b, we can observe that all models answered the question incorrectly. However, the MTL SAN+DeepFix model is the only one that does not answer *white*, answering instead *red*. Also in contrast with the rest of the models, the attention layer of MTL SAN+DeepFix actually focuses on the railings of the stadium while the other models focus mostly on the tennis player in the center of the image - which is likely the reason why their answer is *white*. While it answers incorrectly, we can see from these examples and the ones in Appendix A how MTL SAN+DeepFix learned the best attention representation, which also shows in the results of Table 5.4, where its correlation with human attention is the best of all models, besides the DeepFix baseline (which is solely trained on the attention maps).

Both of these examples show an important attribute of the MTL SAN+DeepFix model. A necessary and sufficient attention map is one that includes just the visual cues necessary to answer the question, being the smallest region of the image necessary to complete the task correctly. On the other hand, an attention map that is only sufficient can be any superset of the necessary and sufficient attention map [2]. We argue that out of all the models, the MTL SAN+DeepFix attention maps seem to be the most efficient at obtaining the necessary and sufficient regions of the image, instead of only



(a) Example of better attention maps leading to a correct answer.

(b) Example of better attention maps not helping to get the correct answer.

Figure 5.1: Answers and respective attention maps of each model presented in this dissertation. The leftmost images are the original images and one of its questions while *Human Attention and Answer* is the respective VQA-HAT attention map and the most popular answer of the image-question pair. The remaining images and answers are the attention maps from our models and their respective answers to the question.

the sufficient regions. We can observe this in both examples shown, where the attention map of this model in Figure 5.1a is the smallest and most correct attention map out of all the remaining models and in Figure 5.1b where the attention map seems to be at least the most correct attention map when compared to the human attention map, even though the model answered the question incorrectly.

Furthermore, we can observe on Tables 5.3 and 5.4 how the Generalized Variance is the lowest for the MTL SAN+DeepFix model, further showing that the attention maps of this model are closer to the necessary and sufficient attention maps when compared to the remaining models. We can observe that, in general, the lower the GV metric is, the better the accuracy of the VQA model will be. Consequently, the generalized variance can be a good metric to observe if the attention maps of a VQA model are relatively close to the ideal necessary and sufficient map by checking this metric along with the VQA accuracy of the model. We can observe that, for instance, the MTL Split Attention model has a close generalized variance to the MTL SAN+DeepFix model, but there is a gap between the accuracies of both models, which suggests that MTL Split Attention may have attention maps focused on specific regions of the image, but these regions are often not the necessary and sufficient.

According to the GV metric, we can observe that one of the worst possible attention maps - a uniform distribution over the whole image - has a GV of 0.096. It's interesting to see how the actual human attention maps have one of the GV's closest to this upper-bound, suggesting that these target distributions used to train our models are not necessarily the best possible attention maps (absolute necessary and sufficient distributions). Nevertheless, our models that use DeepFix turn out to have a generally lower GV, which indicates a tendency of the DeepFix attention model to predict more focused attention maps.

If MTL SAN+DeepFix has seemingly the closest to necessary and sufficient attention maps when compared to other models and has such a high correlation with human attention why does it compare so closely to the unsupervised attention model of SAN+DeepFix in the VQA task? We argue that one of the main reasons is the high language priors of the question-answer pairs of the VQA data set.

During the writing of this dissertation, a second version of the VQA data set was released [56]. The authors observed that the original VQA data set has several unbalanced attributes that make the models developed for it very biased towards some answers that are most common, making their results seemingly good when in reality their visual representations are not as used in the answer generation as initially in-

tended. As an example of these strong language priors, if a question in the VQA data set starts with "Do you see a...", there is a 87% chance that blindly answering "yes" will result in a correct answer.

Due to this priors, the authors of version 2 of the VQA data set created a much more balanced data set where the number of questions in the data set was doubled when compared to the original VQA data set [1]. This balancing was performed by taking an image-question-answer tuple (I, Q, A) of the VQA data set and showing subjects different images I' that are close in the semantic space of the `fc7` layer of VGG Net. The subjects were then asked to find an image where the same question Q could be asked, but where the correct answer would be $A' \neq A$. Some examples of these complementary images can be observed in Figure 5.2.

Figure 5.2: Some image pair examples from version 2 of the VQA data set [56].

Take the first example of Figure 5.2 - if an attention map is very spread out (only sufficient) it would include both faces of the man and woman in the regions of highest probability, most likely leading to a lower chance of answering correctly. With these complementary images, this new version of the VQA data set will lead new VQA models that use attention mechanisms to have attention maps that are necessary and sufficient and not only sufficient. We argue that the human attention information would be much more useful with models trained on this data set, especially if there were human attention examples of these complementary image pairs. The information contained in human attention data leads to attention maps that are much more focused on a region of the image and models that aim to be closer to how humans look at images when performing the VQA task will most probably benefit from this human supervision.

Chapter 6

Conclusions and Future Directions

In this work, we presented a model capable of learning to mimic human behavior in the VQA task while learning the task itself. This was performed using Multitask Learning by including an additional supervision in the attention layer of an existing VQA model. The technique presented in this work can be seamlessly integrated into any model that uses soft attention - not only the VQA model we picked. Furthermore, we also used a state-of-the-art model at saliency prediction to use as a substitute of the regular feedforward attention layer used in soft attention models. This was used to drastically improve the correlation between the predicted attention maps and human attention. As the additional supervision of the saliency task, this human attention prediction model can also be smoothly included in place of any soft attention layer of any other model of VQA.

Despite the success of the models presented, they are not without criticisms. The inclusion of DeepFix instead of a regular attention layer greatly increases the complexity of the model, making it much more time consuming to train. Due to time constraints, we were not able to use the original number of kernels of the DeepFix model, which not only could further improve our model but also further increase said complexity.

Moreover, DeepFix is a model that excels at bottom-up attention prediction, which has been the main focus of saliency prediction research. From comparing our human rank correlation results with the correlation between humans (human agreement), it seems as if DeepFix can also be used for top-down attention since these correlations were very close to each other. It is worth noticing how, in contrast to the original DeepFix work, the input to our DeepFix attention model is, in fact, a combined representation of the question and image and not just of the image. This may drive DeepFix

to be better at the task of top-down attention since cognitive information was added to its input, but it remains unclear if an attention prediction model that excels at top-down attention would have a much better performance or, even if these attention mechanisms are conceptually different, models for both can be used interchangeably as long as the input representations are different.

We have also found that, on average, the attention maps of VQA-HAT used as a target for our model's attention prediction have a generalized variance that is not very distant to the generalized variance of a uniform distribution over the whole image. This leads us to believe that this data set may not contain attention maps that are *necessary* and *sufficient* but only sufficient, since our best model, MTL SAN+DeepFix, ends up learning representations that output attention maps of lesser variance, while still having a high correlation with humans and a good VQA accuracy. This higher variance is likely due to the method used to obtain these human attention maps - subjects were asked to sharpen a blurred image in the regions that they think would help them correctly answer the VQA question. While this will probably make the subjects choose the sufficient regions, there are no guarantees they will choose the absolute smallest necessary and sufficient regions.

We also argued that there are strong language priors in the first version of the VQA data set. These priors result in models learning to use them to their advantage and end up having results that are misleading in the sense that the models are not actually learning how to interpret images but instead the most likely outcomes for most questions. The second version of this data set is much more balanced which would be of great interest to use to train the models presented in this dissertation. We believe that the representations learned by MTL SAN+DeepFix can further improve VQA accuracy if trained in this data set. Having a data set where the selected regions of the image by the attention model is crucial to correctly answer the question results in a need to develop models that predict the necessary and sufficient attention maps. We believe that a model that uses an attention mechanism capable of mimicking human saliency behavior - as the models presented in this dissertation - will have an advantage over other methods that do not.

Finally, in light of the arguments presented in the paragraphs above, we think there are some very interesting paths to follow in future work related to this dissertation. Firstly, it would be interesting to improve the literature in saliency prediction about top-down attention and understand what kind of model is ideal for this type of attention. Furthermore, the creation of a data set for human attention in VQA that aims to

obtain the necessary and sufficient attention maps would also be of great interest. Additionally, these human attention maps could include the complementary image pairs of the second version of VQA and aim to have a higher human agreement than VQA-HAT. These directions will hopefully lead this area of research to arrive at a deeper understanding of the role of attention in VQA.

Appendix A

Attention Map Examples

In this appendix we will show some more examples of the attention maps and respective answers that the models presented in this dissertation output.

Figure A.1: Answers and respective attention maps of each model presented in this dissertation (Example 1).

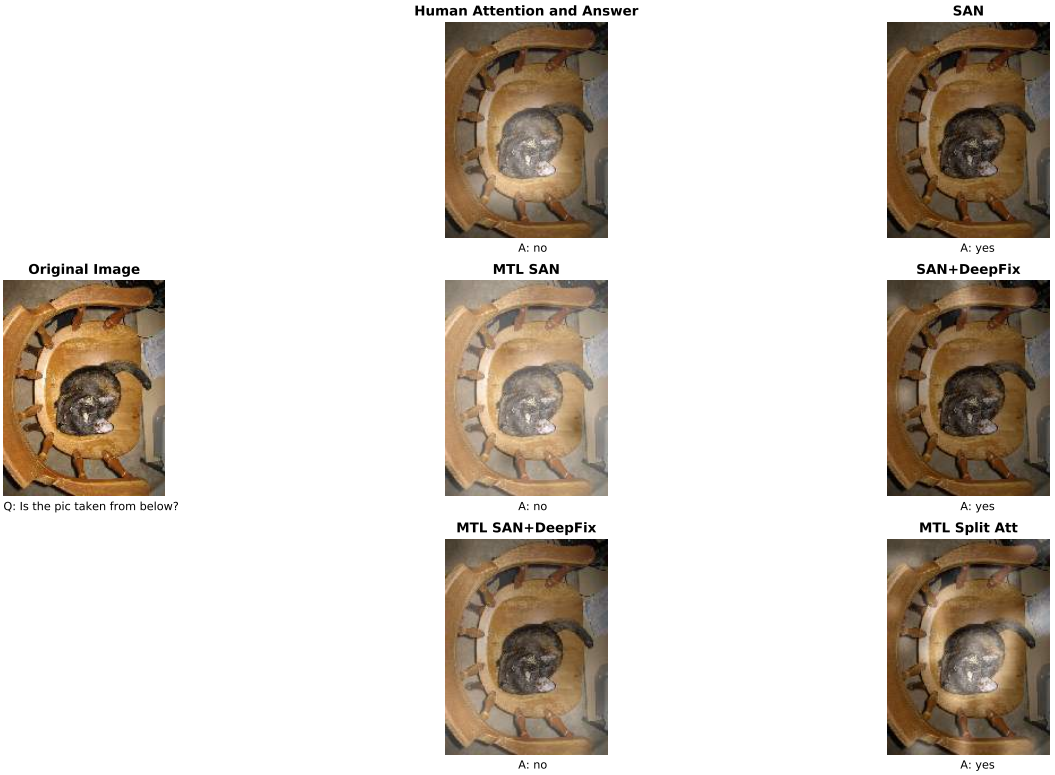


Figure A.2: Answers and respective attention maps of each model presented in this dissertation (Example 2).

Figure A.3: Answers and respective attention maps of each model presented in this dissertation (Example 3).

Figure A.4: Answers and respective attention maps of each model presented in this dissertation (Example 4).

Figure A.5: Answers and respective attention maps of each model presented in this dissertation (Example 5).

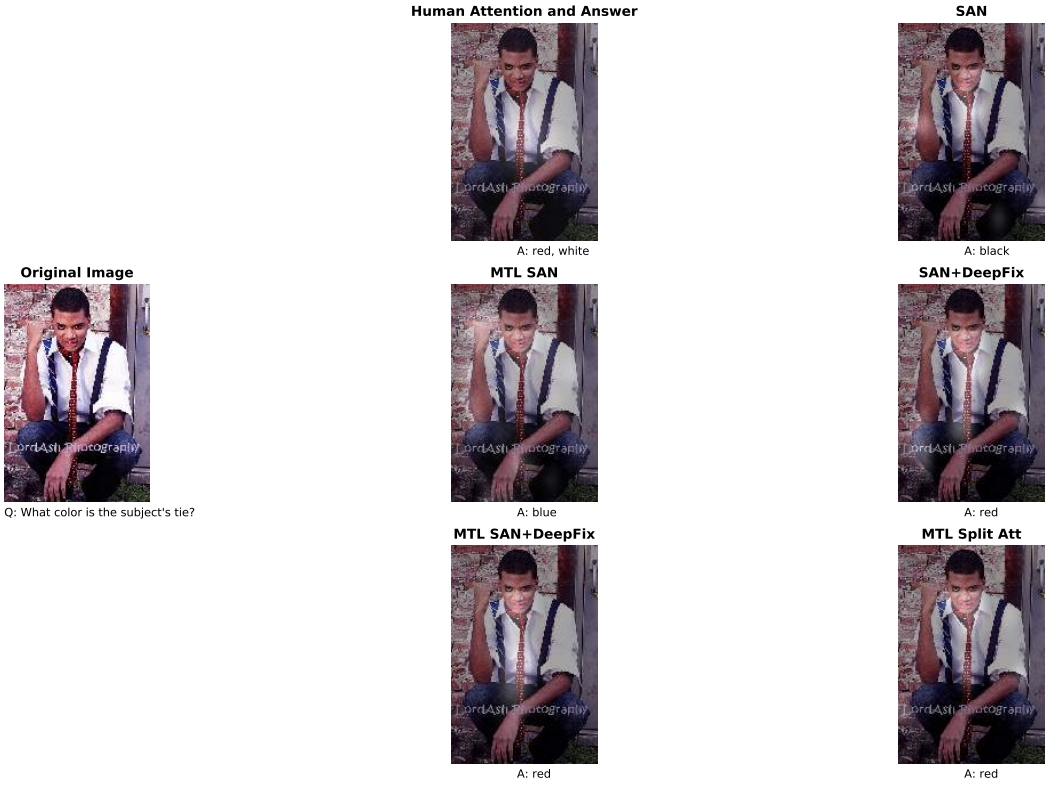


Figure A.6: Answers and respective attention maps of each model presented in this dissertation (Example 6).

Figure A.7: Answers and respective attention maps of each model presented in this dissertation (Example 7).

Figure A.8: Answers and respective attention maps of each model presented in this dissertation (Example 8).

Figure A.9: Answers and respective attention maps of each model presented in this dissertation (Example 9).

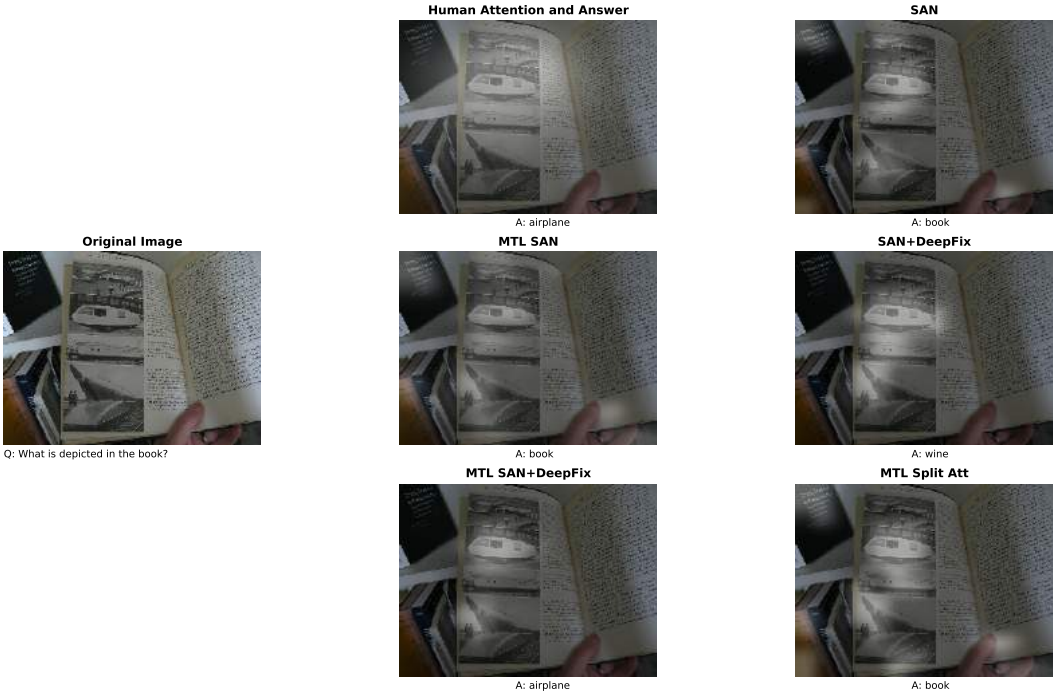


Figure A.10: Answers and respective attention maps of each model presented in this dissertation (Example 10).

Bibliography

[1] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, and D. Parikh, "Vqa: Visual question answering," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2425–2433, 2015.

[2] A. Das, H. Agrawal, C. L. Zitnick, D. Parikh, and D. Batra, "Human attention in visual question answering: Do humans and deep networks look at the same regions?," *arXiv preprint arXiv:1606.03556*, 2016.

[3] R. Caruana, "Multitask learning," in *Learning to learn*, pp. 95–133, Springer, 1998.

[4] R. V. Yampolskiy, "Ai-complete, ai-hard, or ai-easy-classification of problems in ai.," in *MAICS*, pp. 94–101, 2012.

[5] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3156–3164, 2015.

[6] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting on association for computational linguistics*, pp. 311–318, Association for Computational Linguistics, 2002.

[7] R. A. Rensink, "The dynamic representation of scenes," *Visual cognition*, vol. 7, no. 1-3, pp. 17–42, 2000.

[8] M. Corbetta and G. L. Shulman, "Control of goal-directed and stimulus-driven attention in the brain," *Nature reviews. Neuroscience*, vol. 3, no. 3, p. 201, 2002.

[9] H. Hadizadeh and I. V. Bajic, "Saliency-aware video compression," *IEEE Transactions on Image Processing*, vol. 23, no. 1, pp. 19–33, 2014.

[10] T. Yubing, F. A. Cheikh, F. F. E. Guraya, H. Konik, and A. Trémeau, "A spatiotemporal saliency model for video surveillance," *Cognitive Computation*, vol. 3, no. 1, pp. 241–263, 2011.

[11] D. P. Papadopoulos, A. D. Clarke, F. Keller, and V. Ferrari, "Training object class detectors from eye tracking data," in *European Conference on Computer Vision*, pp. 361–376, Springer, 2014.

[12] M. Jiang, S. Huang, J. Duan, and Q. Zhao, "Salicon: Saliency in context," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1072–1080, 2015.

[13] D. E. Rumelhart, G. E. Hinton, R. J. Williams, *et al.*, "Learning representations by back-propagating errors," *Cognitive modeling*, vol. 5, no. 3, p. 1, 1988.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[15] V. Kurkova, "Kolmogorov's theorem and multilayer neural networks," *Neural networks*, vol. 5, no. 3, pp. 501–506, 1992.

[16] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[17] I. Ilievski, S. Yan, and J. Feng, "A focused dynamic attention model for visual question answering," *arXiv preprint arXiv:1604.01485*, 2016.

[18] Y. Zhu, O. Groth, M. Bernstein, and L. Fei-Fei, "Visual7w: Grounded question answering in images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4995–5004, 2016.

[19] C. Xiong, S. Merity, and R. Socher, "Dynamic memory networks for visual and textual question answering," in *International Conference on Machine Learning*, pp. 2397–2406, 2016.

[20] K. J. Shih, S. Singh, and D. Hoiem, "Where to look: Focus regions for visual question answering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4613–4621, 2016.

[21] J. Lu, J. Yang, D. Batra, and D. Parikh, "Hierarchical question-image co-attention for visual question answering," in *Advances In Neural Information Processing Systems*, pp. 289–297, 2016.

[22] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach, "Multimodal compact bilinear pooling for visual question answering and visual grounding," *arXiv preprint arXiv:1606.01847*, 2016.

[23] H. Xu and K. Saenko, "Ask, attend and answer: Exploring question-guided spatial attention for visual question answering," in *European Conference on Computer Vision*, pp. 451–466, Springer, 2016.

[24] J.-H. Kim, S.-W. Lee, D. Kwak, M.-O. Heo, J. Kim, J.-W. Ha, and B.-T. Zhang, "Multimodal residual learning for visual qa," in *Advances in Neural Information Processing Systems*, pp. 361–369, 2016.

[25] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola, "Stacked attention networks for image question answering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 21–29, 2016.

[26] Y. LeCun *et al.*, "Generalization and network design strategies," *Connectionism in perspective*, pp. 143–155, 1989.

[27] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[28] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 315–323, 2011.

[29] CS231n Convolutional Neural Networks for Visual Recognition, "Convolutional layer," 2017. [Online; accessed July 18, 2017].

[30] M. Thom and G. Palm, "Sparse activity and sparse connectivity in supervised learning," *Journal of Machine Learning Research*, vol. 14, no. Apr, pp. 1091–1143, 2013.

[31] Y. Zhou and R. Chellappa, "Computation of optical flow using a neural network," in *IEEE International Conference on Neural Networks*, vol. 1998, pp. 71–78, 1988.

[32] CS231n Convolutional Neural Networks for Visual Recognition, "Maxpool layer," 2017. [Online; accessed July 19, 2017].

[33] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[34] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[35] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[36] D. Jurafsky, *Speech & language processing*. Pearson Education India, 2000.

[37] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[38] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[39] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International Conference on Machine Learning*, pp. 2048–2057, 2015.

[40] J. Ba, V. Mnih, and K. Kavukcuoglu, "Multiple object recognition with visual attention," *arXiv preprint arXiv:1412.7755*, 2014.

[41] V. Mnih, N. Heess, A. Graves, *et al.*, "Recurrent models of visual attention," in *Advances in neural information processing systems*, pp. 2204–2212, 2014.

[42] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[43] heuritech, "A brief report of the heuritech deep learning meetup 5," 2016. [Online; accessed July 28, 2017].

[44] K. Shanmuga Vadivel, T. Ngo, M. Eckstein, and B. Manjunath, "Eye tracking assisted extraction of attentionally important objects from videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3241–3250, 2015.

[45] Y. Sugano and A. Bulling, "Seeing with humans: Gaze-assisted neural image captioning," *arXiv preprint arXiv:1608.05203*, 2016.

[46] S. S. Kruthiventi, K. Ayush, and R. V. Babu, "Deepfix: A fully convolutional neural network for predicting human eye fixations," *IEEE Transactions on Image Processing*, 2017.

[47] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *arXiv preprint arXiv:1606.00915*, 2016.

[48] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[49] theano, "Convolution arithmetic tutorial," 2017. [Online; accessed August 6, 2017].

[50] L. Itti, "Visual salience," *Scholarpedia*, vol. 2, no. 9, p. 3327, 2007.

[51] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*, pp. 740–755, Springer, 2014.

[52] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 20, no. 11, pp. 1254–1259, 1998.

[53] G. Li and Y. Yu, "Visual saliency based on multiscale deep features," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5455–5463, 2015.

[54] P.-H. Tseng, R. Carmi, I. G. Cameron, D. P. Munoz, and L. Itti, "Quantifying center bias of observers in free viewing of dynamic natural scenes," *Journal of vision*, vol. 9, no. 7, pp. 4–4, 2009.

[55] S. S. Wilks, "Certain generalizations in the analysis of variance," *Biometrika*, pp. 471–494, 1932.

[56] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh, "Making the v in vqa matter: Elevating the role of image understanding in visual question answering," *arXiv preprint arXiv:1612.00837*, 2016.