Towards Understanding Deep Saliency Prediction

Thesis submitted in partial fulfillment
of the requirements for the degree of

*Master of Science in **Electronics and Communication Engineering** by Research*

by

M. Navyasri
201431141
navyasri.reddy@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
May 2021

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "**Towards Understanding Deep Saliency Prediction** " by **Navyasri**, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Prof. Vineet Gandhi

To

My Family and Friends

Acknowledgments

I would like to take this opportunity to sincerely thank my adviser, Prof. Vineet Gandhi, for his guidance through my time at IIIT, and for all the suggestions and discussions towards my work. I am grateful for his support, for all the opportunities he has given me, and for allowing me the freedom to explore and learn.

I would like to thank sairam, sravan and avinash for solving problems together in 3D. We together have explored and learned many things. I would also like to thank samyak and pradeep for helping me in saliency work. I have learned a lot from all the discussions.

I thank all my friends for making my time at IIIT a fun and rewarding experience. I feel immensely blessed to have found a set of peers I learnt a lot from, and have memories with, which I will cherish for life.

Most importantly, I would like to thank my parents, brother and sister for being my constant pillar of support. Thank you for giving me all the opportunities, listening to my endless complains and worries patiently, and for all the love and encouragement. You guys are my forever cheerleaders.

Abstract

Learning computational models for visual attention (saliency estimation) is an effort to inch machines/robots closer to human visual cognitive abilities. Data-driven efforts have dominated the landscape since the introduction of deep neural network architectures. In deep learning research, the choices in architecture design are often empirical and frequently lead to more complex models than necessary. The complexity, in turn, hinders the application requirements. In this work, we identify four key components of saliency models, i.e., input features, multi-level integration, readout architecture, and loss functions. We review the existing state of the art models on these four components and propose novel and simpler alternatives. As a result, we propose two novel end-to-end architectures called SimpleNet and MDNSal, which are neater, minimal, more interpretable and achieve state of the art performance on public saliency benchmarks. SimpleNet is an optimized encoder-decoder architecture and brings notable performance gains on the SALICON dataset (the largest saliency benchmark). MDNSal is a parametric model that directly predicts parameters of a GMM distribution and is aimed to bring more interpretability to the prediction maps. The proposed saliency models can be inferred at 25fps, making them suitable for real-time applications. We also explore the possibility of improving saliency prediction in videos by using the image saliency models and existing work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this era of rapidly growing information on the internet, there are numerous images online. According to Mary Meeker's annual Internet Trends report, people uploaded an average of 1.8 billion digital images every single day [1]. If we think of this as a huge dataset, we need to look for patterns or common occurrences of tiny things in order to draw a line between the meaningful and non meaningful data. Consider the examples of image classification, object detection and image style transfer which are done by extracting the required features. Identifying the major patterns plays an important role in helping us achieve the specific tasks.

Visual attention enables humans to quickly analyse complex scenes and devote the cognitive abilities towards important regions. Simulating this behaviour with images/videos will decrease the computational complexity for many problems in various fields such as computer vision, robotics, human computer interaction etc. and this is modelled as saliency prediction. It consists of predicting human eye fixations and interesting regions in an image. Saliency is represented as a heat map having higher values at more significant regions. Figure 1.1 shows various applications like drone surveillance, indoor navigation and social interactions

In this thesis we propose new models and provide extensive study on the choice of each component in the model. We propose a) Encoder-decoder model which is simple and shows real-time inference, b) A novel parameter based model which predicts parameters of GMM model instead of predicting saliency map. Before going into the thesis let's look at the what saliency is and the datasets used to train saliency models and the metrics used to evaluate them. Section 1.1 and 1.2 focuses on datasets and metrics used for saliency

Saliency prediction has RGB image as input and the predicted image is distribution map called predicted saliency map as shown in figure 1.2. There are two types of ground truths i.e ground truth

[1] https://www.theatlantic.com/technology/archive/2015/11/how-many-photographs-of-you-are-out-there-in-the-world/413389/

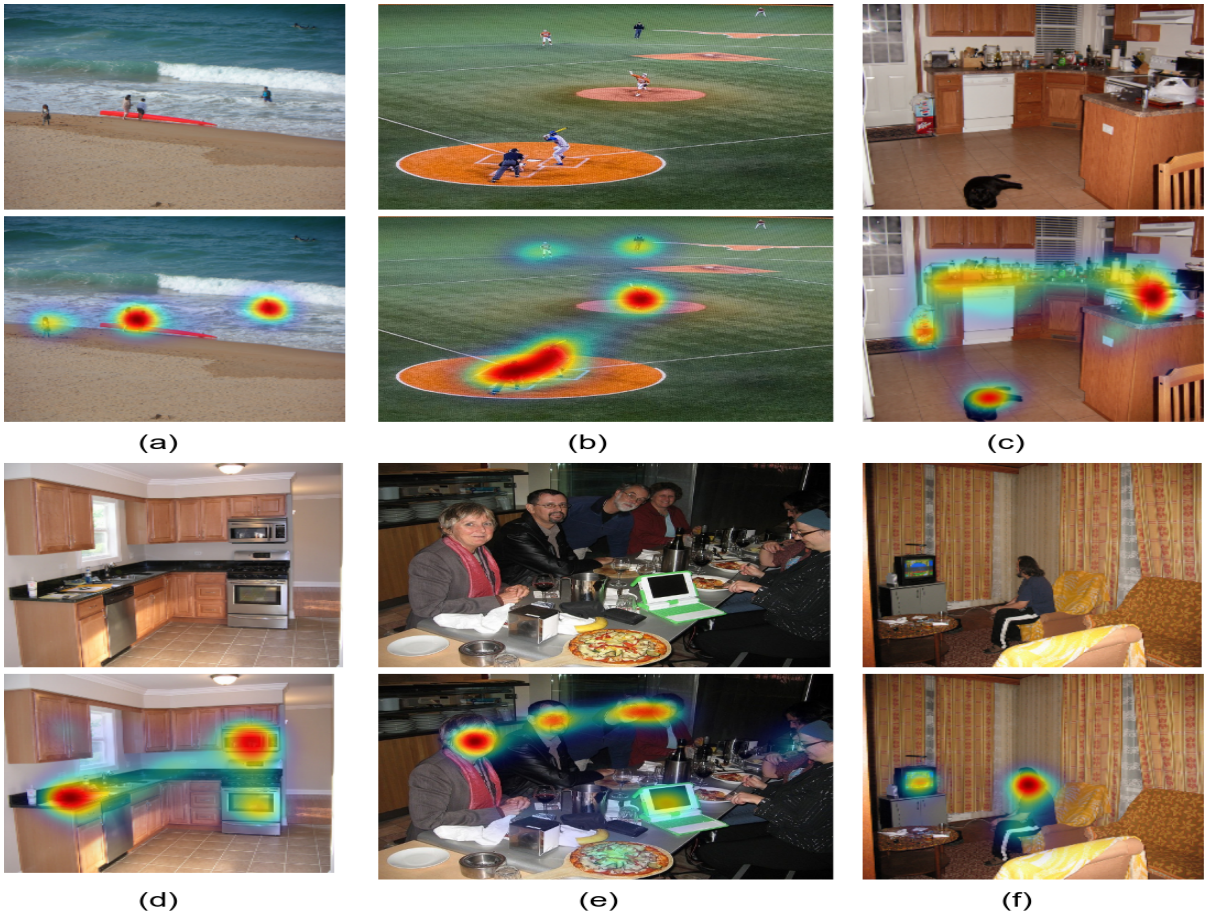

Figure 1.1 Example results of our approach on images from Salicon dataset. Saliency models can play key role in application like (a) drone surveillance , (b) robotics cameras for sports , (c,d) indoor navigation and (e,f) social interactions

.

saliency map and fixation map. Fixation map is a binary map with values either 0 or 1 having eye fixations where as saliency map is a distribution based map to which the output has to match. In figure 1.2 (b) and (c) are ground truth saliency and fixation maps respectively and (d) is the predicted saliency map using our model MDNSal.

Predicting the salient regions in a scene is a fundamental ability, which empowers primates to rapidly analyze/interpret the complex surroundings by locating and devoting the focus only on sub-regions of interest [26]. The work by [20] triggered early interest in the computational modeling of visual saliency from images, i.e., identifying areas that are salient in a scene. Since then, a large variety of saliency detection models have been proposed and find usages in a wide range of applications involving machine vision. Many recent works show that availability of saliency maps enhance cognitive abilities

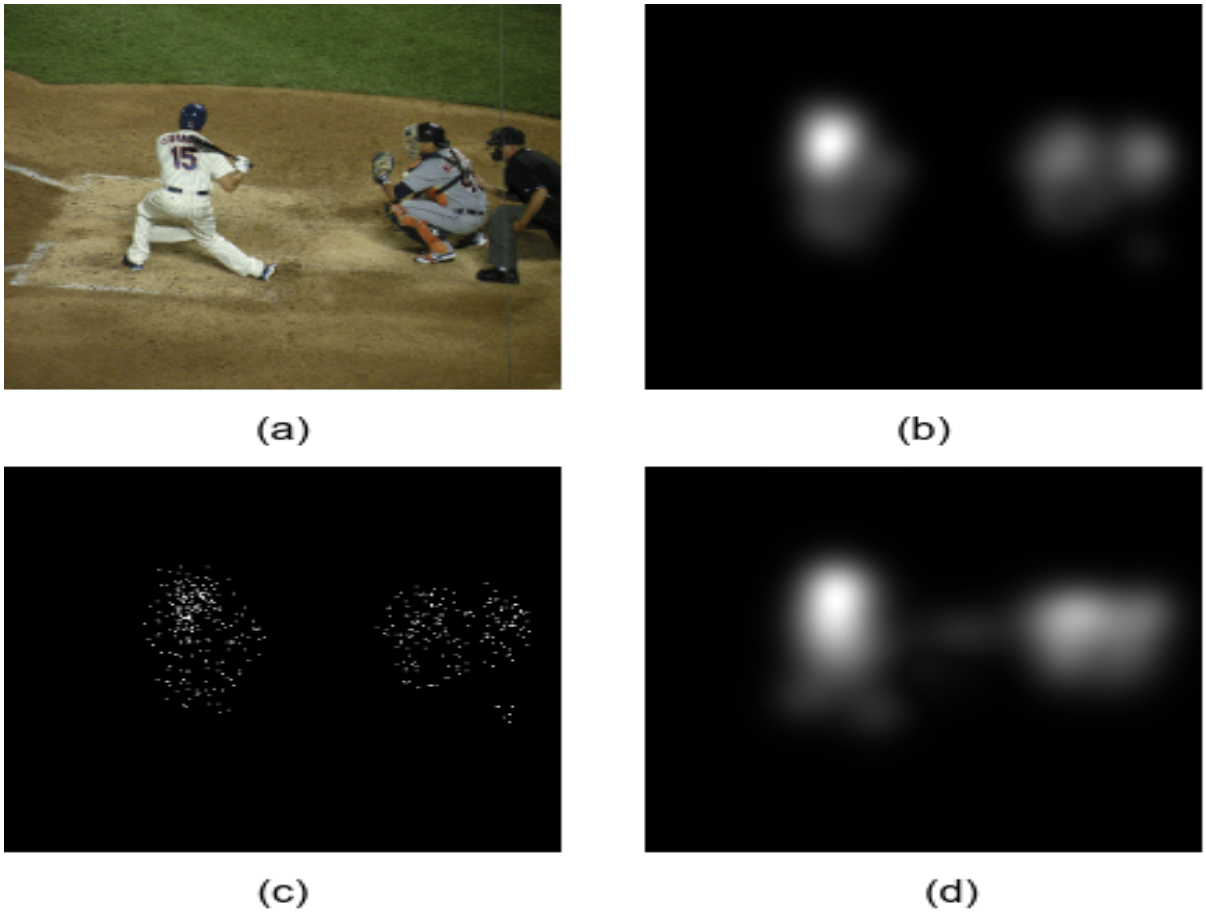

Figure 1.2 a) RGB Image b) Ground truth saliency map c) Ground truth fixation map d) Predicted saliency map using model MDNSal

of robots and helps improving performance in variety of tasks including human-robot interaction [50]; identification, and recognition of objects [47]; scene classification [4]; detecting and tracking regions of interest [15]; proposal refinement [10] and visual search in unknown environments (allowing search of regions with higher importance first) [43]. Our work is application agnostic and focuses on improving the general saliency prediction and can cater to large variety of applications in robotic vision. Some example results from our SimpleNet model are illustrated in Figure 1.1.

The last few years have seen tremendous advancements in the field, mainly due to the application of Deep Neural Network architectures for the task and the availability of large scale datasets [23]. Let's look into datasets available for saliency prediction so far.

1.1 Datasets

There are three popular datasets available for image saliency prediction, SALICON, MIT and CAT.

1.1.1 SALICON

SALICON is the largest crowd-sourced image saliency dataset [23]. Images for this dataset have been taken from Microsoft COCO dataset. It consists of 10,000 training, 5000 validation, and 5000 test images. It was labeled based on mouse-tracking (shown to be equivalent to the eye-fixations recorded with an eye-tracker). All images in this dataset has equal resolution of 640x480 but we resize these images to 256x256 while training.

We use the SALICON dataset for training our models. The benchmark on SALICON test set is known as LSUN saliency challenge [2]. It offers seven evaluation scores and our experiments are based on the newest release, SALICON 2017, from the LSUN challenge.

1.1.2 MIT300

MIT300 test dataset consists of 300 natural images with eye-tracking data of 39 observers. This dataset is a collection of images from the Flickr Creative Commons and personal collections. The labels of MIT300 are non-public.

We use MIT1000 [24] consisting of 1003 images to fine-tune the initial model trained on SALICON dataset. We have done 10 fold cross-validation by splitting the images into 903 train, and 100 validation and have chosen the best model for test submission. This dataset has images with various resolutions such as 1024x768, 685x1024 etc and also synthetic images, thus becomes challenging for model to predict and becomes crucial in comparison of efficiency of model.

1.1.3 CAT2000

CAT2000 consists of 4000 images(2000 train and 2000 test) taken from 20 different categories like Action, Art, Cartoon, Inverted, Sketch, Social etc., with eye-tracking data from 24 observers. Each category contains 100 images each with 1920x1020 resolution. These images are collected from various computer vision datasets and also by using search engines.

Similar to MIT1003, we split the images into 1800 train and 200 validation and perform fine-tuning on our model. Both CAT and MIT test sets are evaluated at MIT saliency benchmark [3]. Figure 1.3

[2] http://salicon.net/challenge-2017/

[3] http://saliency.mit.edu/

shows one each from each of the above datasets.

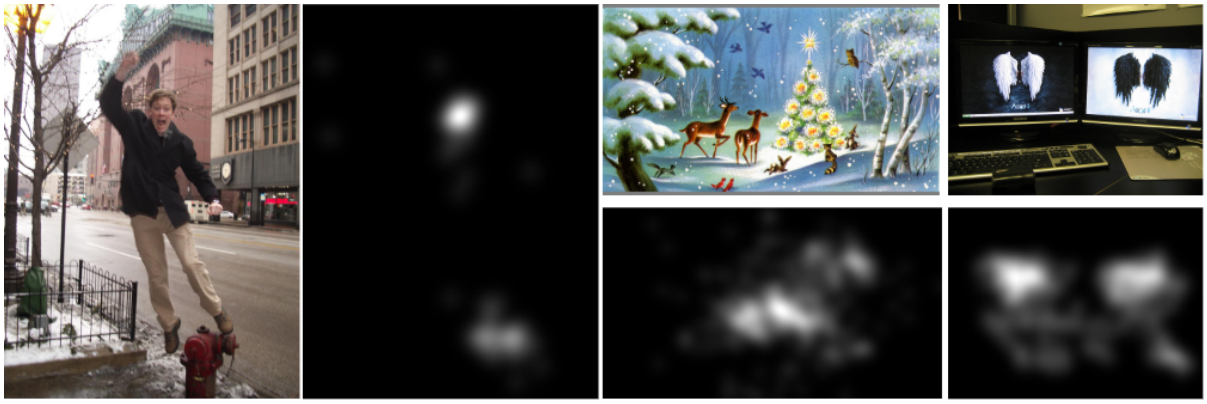

Figure 1.3 RGB images and ground truth saliency maps from MIT, CAT and SALICON datasets respectively.

The above MIT and LSUN benchmarks uses different evaluation metrics to compare submitted models. Let's look into the metrics in this section.

Formally, computational saliency models predict the probability distribution of the location of the eye fixations over the image, i.e., the saliency map. Where human observers look in images is often used as a ground truth estimate of image saliency. The predictions are evaluated using a variety of metrics, which are broadly classified as location-based or distribution-based [7]. The location-based metrics measure the accuracy of saliency models at predicting discrete fixation locations. Distribution based metrics compute the difference/similarity between predicted and ground truth distributions (assuming that the ground truth fixation locations are sampled from an underlying probability distribution).

1.2 Metrics

There are various evaluation measures for saliency prediction that are broadly classified into two types of metrics i.e distribution based and location based metrics. As the names suggest, distribution takes ground truth saliency map and location takes fixation locations as ground truth. Kldiv, Similarity and Correlation(CC) metrics are distribution based metrics where as NSS, AUC metrics are location based metrics. A detailed analysis on how these metrics work as loss functions is shown in Chapter 3. The P, Q terms used below represent predicted saliency map and ground truth respectively.

1.2.1 Distribution based metrics

In distribution based metrics ground truth is saliency map. Some of the distribution based metrics are as follows

KLDiv - Kullback-Leibler(KL) measures the difference between two probability distributions. So, it evaluates the loss of information between predicted saliency map and ground truth saliency map.

$$KLdiv(P, Q) = \sum_i Q_i \log(\epsilon + \frac{Q_i}{P_i + \epsilon}) \tag{1.1}$$

here P, Q are predicted and ground truth maps respectively and ϵ is a regularization term.

CC - The Pearson's Correlation Coefficient(CC) is a statistical method used generally in the sciences for measuring how correlated or dependent two variables are. CC can be used to interpret predicted saliency and ground truth saliency maps, P and Q, as random variables to measure the linear relationship between them.

$$CC(P, Q) = \frac{\sigma(P, Q)}{\sigma(P) \times \sigma(Q)} \tag{1.2}$$

SIM - The similarity metric, SIM(also referred to as histogram intersection), measures the similarity between two distributions, viewed as histograms. SIM is computed as the sum of the minimum values at each pixel, after normalizing the input maps. If P is the predicted saliency map and Q is ground truth saliency map.

$$SIM(P, Q) = \Sigma_i min(p_i, q_i) \tag{1.3}$$

1.2.2 Location based metrics

In location based metrics ground truth is fixation map.

NSS - The NSS function was defined specifically for the evaluation of saliency models. NSS aims to quantify the saliency map values at the fixated locations and to normalize it with the predicted map variance.

$$NSS(P,Q) = \frac{1}{M}\Sigma_i \frac{p_i - \mu(P)}{\sigma(P)} \times q_i^{fix} \tag{1.4}$$

where M is the total number of fixation points, P is normalised to have zero mean and unit standard deviation.

AUC-JUDD - The Area Under the ROC Curve is one of the most widely used metric for evaluating saliency maps. In particular, the AUC-JUDD version, for a given threshold, the true positive rate is the ratio of true positives to the total number of fixations, where true positives are saliency map values above threshold at fixated pixels. And the false positive rate is the ratio of false positives to the total number of saliency map pixels at a given threshold, where false positives are saliency map values above threshold at unfixated pixels.

sAUC - Psychological studies show that the gazes of observers are biased towards center when they look at images [51, 53]. This is because photographers tend to place the object of interest at center of the image. This is called center-bias and sAUC metric captures this if model is also biased to the center of the image.

Most of the datasets tend to include a higher density of fixations around the center of the images. So, if a model has a center bias to predict, it will be able to account for at least part of the fixations on an image, independently of the image content. So, sAUC penalizes models that include this bias by sampling negatives from fixation locations from other images, instead of uniformly at random.

1.3 Contribution of this thesis

Recent works have analyzed the saliency estimation models over different evaluation metrics to add interpretability to saliency scores [7]. Interestingly, the interpretability of saliency model architectures has not been systematically explored. To this end, we propose a componential analysis which can be used to compare a model from another; reduce redundancies in the model without compromising the performance and can help customizations based on application requirements.

We identify four key components in saliency models(figure 1.4). First is the input features i.e., to directly send the image to the saliency models or employ transfer learning using pre-trained networks. The second component is the multi-level integration. It is understood that multiscale features (at different spatial and semantic hierarchy) capture a broad spectrum of stimuli, and the combination improves

model performance. This aspect concerns how the hierarchy is imbibed in the model. The third aspect is the readout architecture, which concerns the form of output i.e., to directly predict a saliency map or to predict parameters of an assumed underlying distribution. The fourth aspect is the loss function. Different works use a different combination of loss functions; however, most of these choices are only justified empirically. We explore ways to validate these combinations more formally. Overall, our work makes the following contributions:

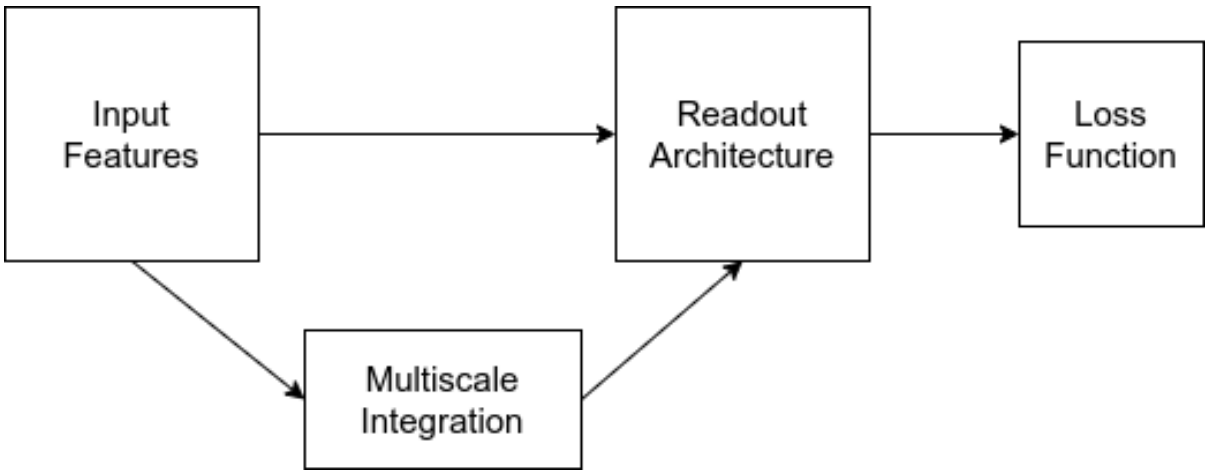

Figure 1.4 Models are analysed in four components a) Input features b) Multi-scale Integration c) Readout Architecture d) Loss Function

- We separate components of saliency models and discuss the progress on each of them in reference to the literature. Such analysis can help better interpret the models i.e., assess component-wise weaknesses, strengths, and novelty. The analysis allows to optimize saliency models by trying alternates for a particular component while freezing the rest of them.

- We propose an encoder-decoder based saliency detection model called SimpleNet. The main novelty of SimpleNet is a UNet like multi-level integration [46]. SimpleNet is fully convolutional; end to end trainable; has lower complexity than counterparts and allows real-time inference. It gives consistent performance over multiple metrics on SALICON and MIT benchmarks, outperforming state of the art models over five different metrics (with significantly notable improvements on KLdiv metric).

- We propose a parametric model called MDNSal, which predicts parameters of a GMM instead of a pixel-level saliency map. The main novelty of MDNSal is in readout architecture with a modified Negative Log Likelihood (NLL) loss formulation. It achieves near state of the art performance on SALICON and MIT benchmarks.

1.4 Thesis Overview

The following chapters are organised as follows: Chapter 2 discusses the related work and recent architectures of image saliency which are compared to our models. Chapter 3 explains our models in detail by discussing the selection of components empirically and theoretically. Chapter 4 explores datasets, related work and experiments of saliency on videos. Chapter 5 is the conclusion to thesis.

Chapter 2

Related Work

This chapter has two sections. First section discusses on the related work of saliency and the second sections goes deep into the architectures of different models which we use to compare our models.

2.1 Key Components of Saliency Models

2.1.1 Input features

Early attempts relied on handcrafted low-level features for saliency prediction. Seminal work by Itti [20] relied on color, intensity, and orientation maps (obtained using Gabor filter). Valenti [6] use isophotes (lines connecting points of equal intensity), color gradients, and curvature features. Zhang [57] computes saliency maps by analyzing the topological structure of Boolean maps generated through random sampling. Bruce [5] use low-level local features (patch level) in combination with information-theoretic ideas. Jude [24] included high-level information by using detectors for faces, people, cars, and horizon. However, most of these methods remain elusive on generic high-level feature representation.

Recent works are dominated by deep learning architectures owing to their strong performance. Most of this success can be attributed to Convolutional architectures [30, 29, 11, 21]. Some works have also explored combining CNN with recurrent architectures [12]. The breakthrough happened via transfer learning of high-level features trained for image classification [27, 48, 16, 58, 18]. The large scale SALICON dataset [23] was pivotal in transfer learning process (allowing efficient fine-tuning). Initial approaches relied on Alexnet or VGG features [30, 11, 35]. Other notable architectures like ResNet, DenseNet and NasNet [12, 21] were then explored. Recent works also explore the combinations of features from multiple pre-trained networks [21]. There is enough evidence to agree that using pre-trained features brings significant gains on the task of saliency prediction. We analyze two important design choices: (a) which pre-trained network to pick and (b) should pre-trained weights be frozen or fine-tuned.

2.1.2 Multi-level integration

It is evident that deep learning models utilizing high-level features significantly outperform the older counterparts, which rely on low-level handcrafted features. However, recent work [32] suggests that the simple low-level model better explains a substantial proportion of fixations when compared to the state-of-the-art model. They quantitatively show this by changing the input features to low-level intensity contrast features (ICF) and keeping the rest of the architecture the same.

Deep networks have employed two main strategies to resolve this concern. The first is to send different image scales as input in parallel. SALICON [19] model uses two different image scales and the idea was then extended to multiple scales [36, 33]. The multi scale spatial stimuli can be tackled by this approach, but not necessarily the semantics. An alternate approach is to take features from different stages of pre-trained CNN. Different levels of semantics (from low, mid, and high-level features) can be thus directly incorporated into the model to resolve the concerns raised in [32]. The work by [30] takes a weighted sum of features at different levels post resizing, where the weights are trained through the network. ML-Net [11] resizes and concatenates features from different levels of VGG-16 model and passes it through additional convolutions layers to predict the map. The work in [54] individually predicts saliency maps for features from different stages of VGG-19 and then fuses them. In this work, we propose a novel UNet [46] like architecture for incorporating multi-level features. The single-stream network with skip connections speeds up training, and the structure allows for an organic hierarchical refinement from high to low-level features (symmetric expansion over high-level context to enable precise localization).

2.1.3 Readout Architectures

The commonly used readout architecture consists of few convolutional layers post the encoder followed by 1×1 convolutions to control the size of the output saliency map. It is also common to learn an additional prior [30, 11, 32]. The prior is often aimed to compensate for the central fixation bias. Work by [12] employs an LSTM based readout architecture and learns a set of 2D Gaussian priors parameterized by their mean and variance (instead of a single one). In this paper, we employ minimal readout architecture with only convolutional layers. We show that combined with an UNet like multi-resolution encoder; such architecture can outperform state of the art models which rely on priors, complex architectures [12] or multi-network feature combinations [21].

Interestingly, most of the state of the art models directly predict an image as output (the saliency map). Parametric models for computing image saliency have not been explored. Although parametric models come with a bound on the complexity of the model (even if the amount of data is unbounded), they come with several advantages; especially, they are easier to understand and interpret (Where saliency models should look next? [8]). Furthermore, they allow better integration with down-

stream applications. The importance of predicting distributions has been nicely motivated in [31]. To this end, we propose a novel readout architecture, which directly predicts parameters of a 2D GMM (mean, variance, and mixture weights). The proposed readout architecture can be plugged at the end of any given architecture to output a parametric distribution. We show that, although bounded, the parametric models can achieve near state of the art performance.

2.1.4 Loss functions

Mean Squared Error (MSE) between predicted and ground truth has been employed as loss function [29]. ML-Net introduced a normalized version of MSE [11]. Most of the recent efforts directly use one of commonly used evaluation metrics or a combination of them as a loss function. The most commonly used loss is computing KL-divergence (KLdiv) between the estimated and ground-truth saliency maps [19]. Some papers use a variation of it like negative log likelihood [30] or cross entropy [54] instead. Recent works use KLdiv in combination with other metrics like Pearson's Correlation Coefficient (CC), Normalized Scanpath Saliency(NSS), and Similarity. These combinations bring clear improvement in performance [21, 12]. However, the combinations are often decided empirically. In this work, we provide formal insights to choose a minimal and comprehensive loss function.

2.2 Related Architectures

In this section, we are going to analyse some architectures with which we compare our models.

2.2.1 SAM NETs

This paper [12] has Attentive ConvLstm model to predict saliency from images. The input image is sent through VGG or Resnet and the output(let's call it as X) will have 512 channels which is sent to Conv LSTM with each LSTM having the previous timestamp output and X. If the output from LSTM is X_t , then a set of 16 gaussian channels are appended to it which are called priors and these are learned during training. These priors add center bias to the system which makes the model closer towards human vision as humans tend to see the center region most. They also used dilated convolutional networks in VGG or ResNet.

The **loss function** used for this network is a linear combination of KLDiv, CC an NSS as shown below.

$$L(P, Q, Q^{fix}) = \alpha L_1(P, Q^{fix}) + \beta L_2(P, Q) + \gamma L_3(P, Q) \tag{2.1}$$

here L_1, L_2, L_3 are NSS, CC and Kldiv respectively and α, β, γ are scalars with values -1, -2 and 10 respectively. Q^{fix} is the fixation map.

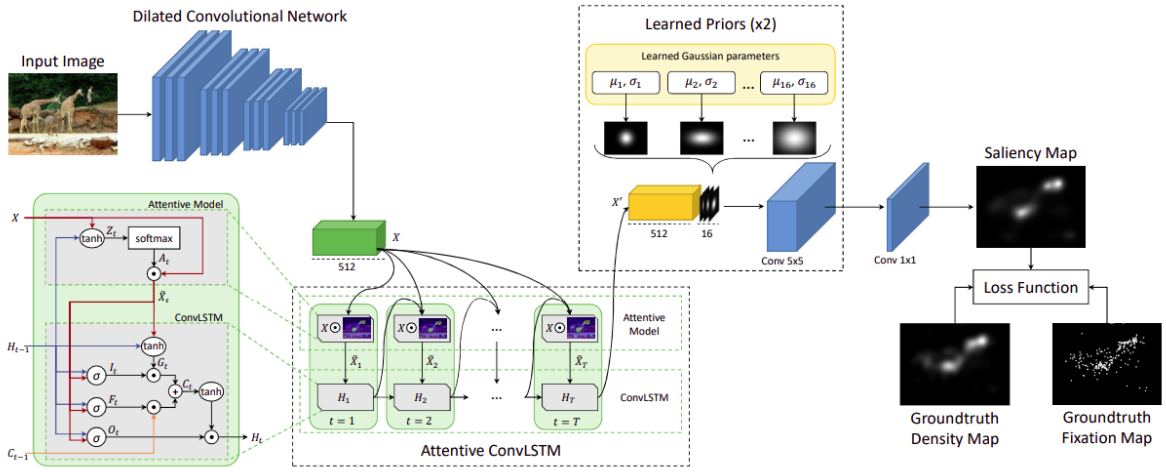

Figure 2.1 SAM Net Architecture

2.2.2 GazeGAN

GazeGAN [9] trains model by data augmentation using multiple transformations like cropping, rotating etc. to the existing datasets by generating 1800 images using 18 transformations from 1000 images of CAT2000(It makes a total of 1900 images for CAT2000) and 60,000 images using 10,000 SALICON images(This makes a total of 70,000 images for SALICON).

MIT1003 is divided into 600 train, 100 validation, 303 test images and data augmentation has been done and fine tuned with the parameters learned using SALICON dataset.

The proposed dataset of GazeGan consists of 19 transformation groups, Thus having 1140 training samples, 190 validation samples and 570 test samples, fine tuned with parameters using SALICON dataset.

Model - GazeGAN(in figure 2.2) uses Conditional GAN for training the network and some additional ideas to improve the model and learn the salient regions. It uses Center-surrounded connection(CSC) to highlight semantic information and suppress artifacts, Skip-Connections to use multiscale information and Local-Global GANS to improve robustness in the model. It is using multiple discriminators at different scales. In this model two discriminators have been used, one for original size input images (G_L) and the other for images which are scaled down by 4 times to original size(G_g). In the end, the decision of these two discriminators are considered for loss function.

Generator has modified U-Net with CSC included in it. U-Net is an encoder decoder network in a symmetric fashion with skip-connections used for bio-medical image segmentation.

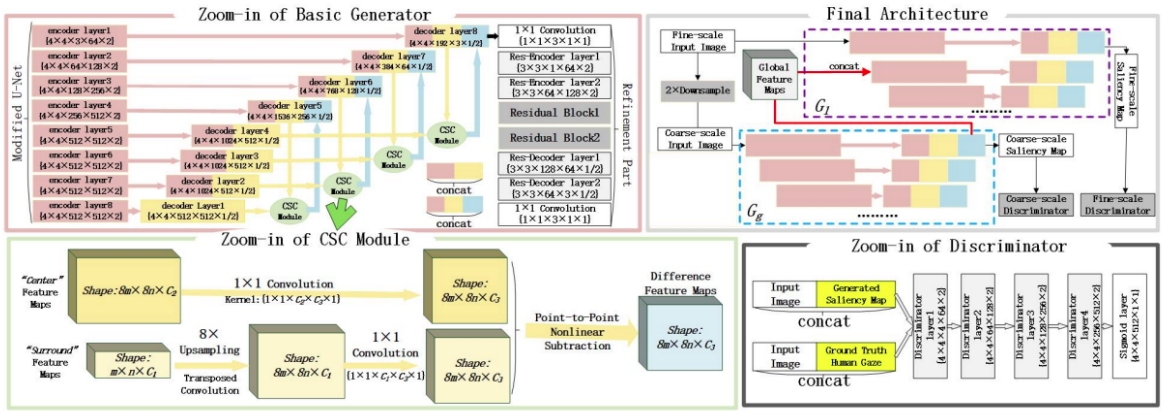

Figure 2.2 GazeGAN Architecture

2.2.3 SALICON

The architecture(in figure 2.3) of this model is applied on two image scales and the output of these are concatenated and passed to 1x1 convolution layer to get the map. Image is passed through image classification models (VGG, ResNet, GoogleNet). Input Image is 600 x 800, output of fine will be 37 x 50 x C. C is 512 for VGG, 256 for AlexNet, 832 for GoogleNet. The Coarse output will be upsampled and concatenated with the output of fine. Metrics are calculated both with and without fine tuning the 3 models.

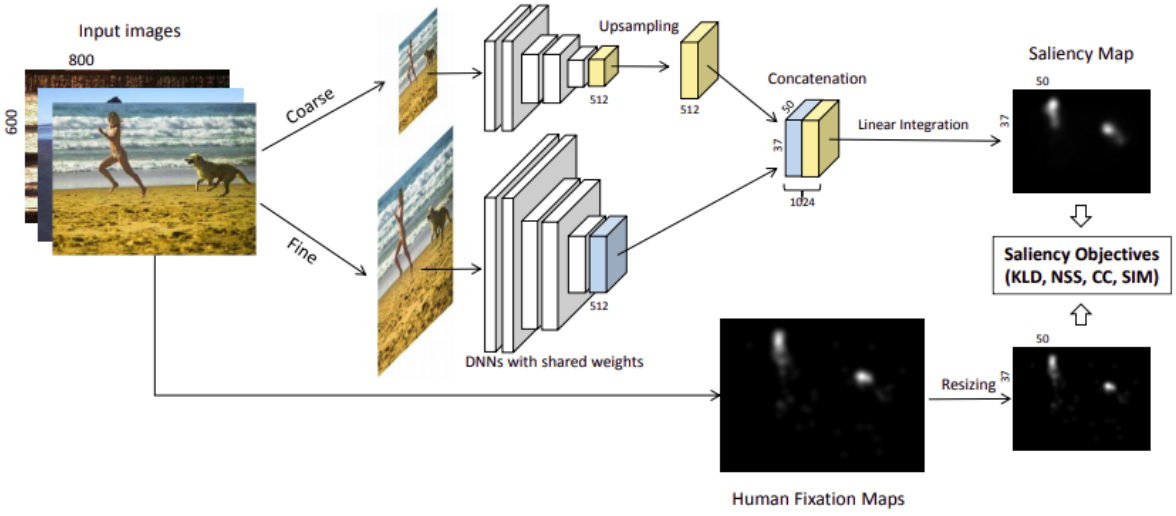

Figure 2.3 SALICON Architecture

2.2.4 EMLNET

EMLNET [21] in figure 2.4 is an encoder-decoder based approach consisting of deep models like NasNet and DenseNet. Both the encoder and decoder are trained seperately to deal with the complexity and space.

In the encoding stage, both the NasNet and DenseNet are separately optimized for saliency prediction by replacing FC with conv layers. In order to relax the requirements on space, the output prediction is compressed into one feature map by applying conv1. In the decoding stage, they train a decoder to combine the learned features from the two CNN models that have been trained at the encoding stage. Multi-Layer features are extracted from both the networks (4 layers from the DenseNet and 3 from NasNet totaling 13,536 feature maps). These 7 maps are upsampled to the same size and are concatenated which is then used in the prediction.

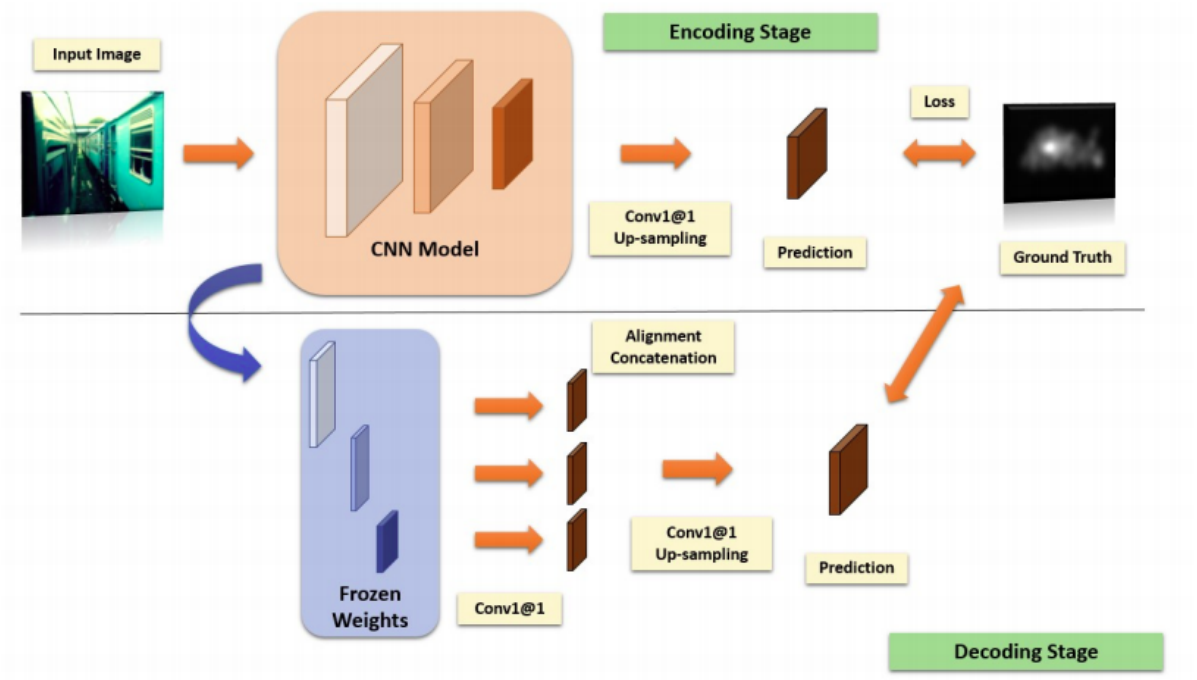

Figure 2.4 EMLNET Architecture

Chapter 3

Proposed Architectures

We propose two end-to-end architectures **SimpleNet** and **MDNSal**. SimpleNet is an encoder-decoder architecture that predicts the pixel-wise saliency values, while MDNSal is a parametric model that predicts parameters of a GMM distribution. We now describe each of the models in detail.

3.1 SimpleNet

The overall architecture of the SimpleNet is shown in figure 3.1. It is a fully convolutional, single-stream encoder-decoder architecture, which is end to end trainable. The name SimpleNet is derived from the design goal to keep each component simple and minimal, resulting in an architecture which is easy to train, comprehend and reproduce, without compromising the performance.

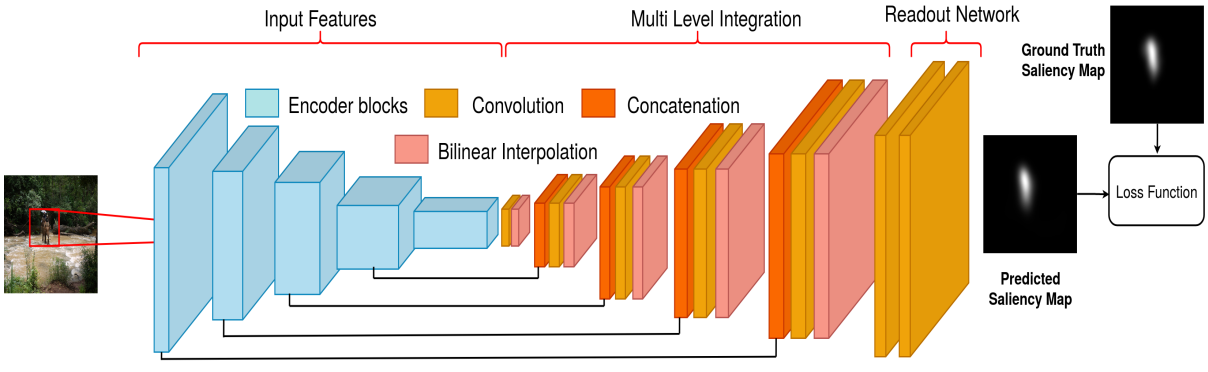

Figure 3.1 SimpleNet Architecture

3.1.1 Input features

SimpleNet directly takes input from the pre-trained architectures designed for image classification. We explore four different architectures VGG-16, ResNet-50, DenseNet-161, and PNASNet-5 and compare their performance. The feature extraction layers (shown as encoder blocks in Figure 3.1) are initialized with the pre-trained weights and later fine-tuned for the saliency prediction task.

3.1.2 Multi-level integration

SimpleNet employs a UNet like architecture that symmetrically expands the input features starting at the final layer of the encoder (input features). The symmetric expansion enables precise localization. Every step of the expansion consists of an upsampling of the feature map, a concatenation with the corresponding scale feature map from the encoder. The number of channels are then reduced using 3×3 convolutions followed by ReLU). Figure 3.2 shows that adding skip connections increases the localisation and also distributes the saliency to high contrast images rather than high-level features as mentioned in [32].

There are four skip connections in the SimpleNet model and to empirically examine how skip connections are effecting the model, A set of experiments are done by removing skip connections i.e no skip connections, adding first skip connection then second, third and fourth respectively. Table 3.1 shows SimpleNet having skip connections has better performance

no. of skip connections	CC	KLdiv	NSS
0	0.856	0.212	1.871
1	0.906	0.205	1.913
2	0.904	0.197	1.909
3	0.904	0.197	1.909
4 (SimpleNet)	**0.907**	**0.193**	**1.926**

Table 3.1 SimpleNet's validation results on SALICON with varying skip connections

3.1.3 Readout architecture

The readout architecture consists of two 3×3 convolutional layers; the first is followed by ReLU, and the second uses a sigmoid to output the saliency map.

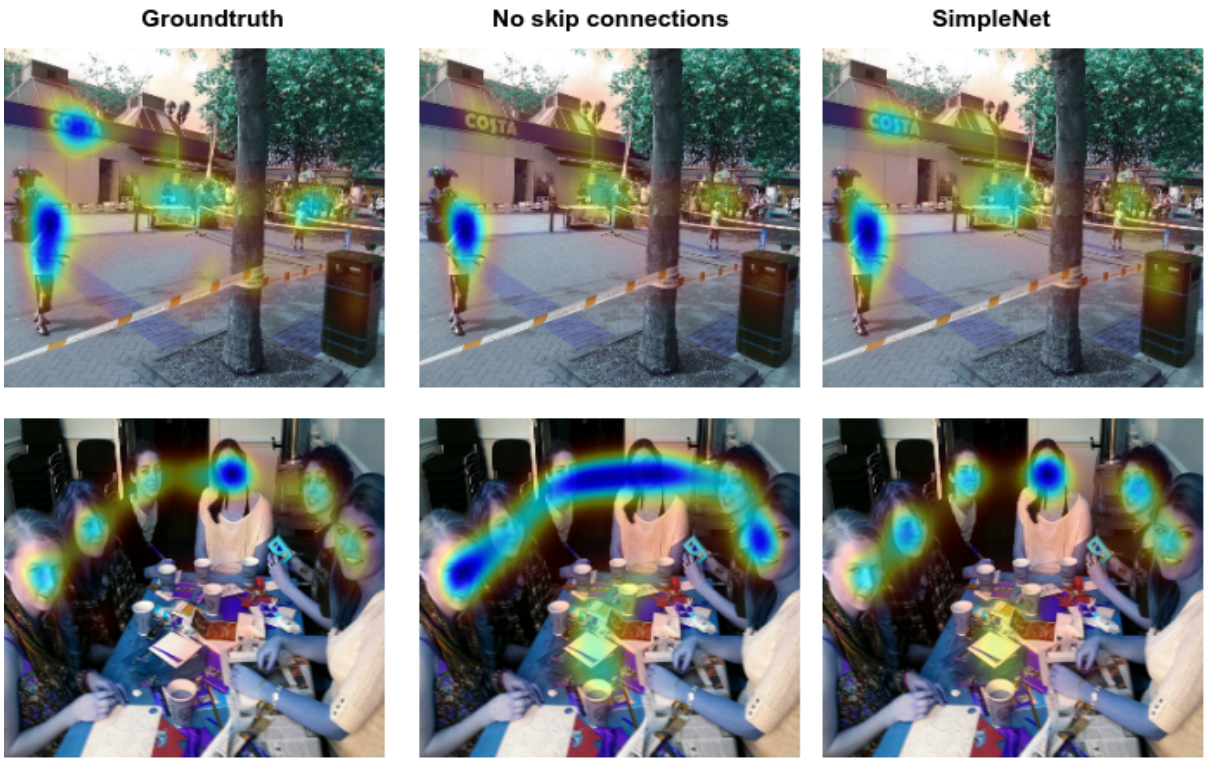

Figure 3.2 Output of SimpleNet architecture with and without skip connections on SALICON validation dataset.

3.1.4 Loss function

The loss function compares the output saliency map with the ground truth. We use a combination of Kullback-Leibler Divergence(KLdiv) and Pearson Cross Correlation (CC) metrics as a loss function. KLdiv is an information-theoretic measure of the difference between two probability distributions:

$$KLdiv(P, Q) = \sum_i Q_i \log(\epsilon + \frac{Q_i}{P_i + \epsilon}), \tag{3.1}$$

where P, Q are predicted and ground truth maps respectively and ϵ is a regularization term.
CC is a statistical method used generally in the sciences for measuring how correlated or dependent two variables are

$$CC(P, Q) = \frac{\sigma(P, Q)}{\sigma(P) \times \sigma(Q)}. \tag{3.2}$$

KLdiv is an asymmetric dissimilarity metric with lower score indicating better approximation of the ground truth [7]. Wherever the ground truth value Q_i is non-zero but P_i is close to or equal to zero, a large quantity is added to the KL score. This makes Kldiv highly sensitive to false negatives as shown in figure 3.3.

CC considering the values as random variables, finds correlation between the maps and treats false positives and false negatives symmetrically. SIM measures the histogram intersection between the maps and thus it becomes less sensitive to false positives than false negatives as shown in figure 3.3(taken from [7]). SIM score for both the images are similar whereas CC score is different.

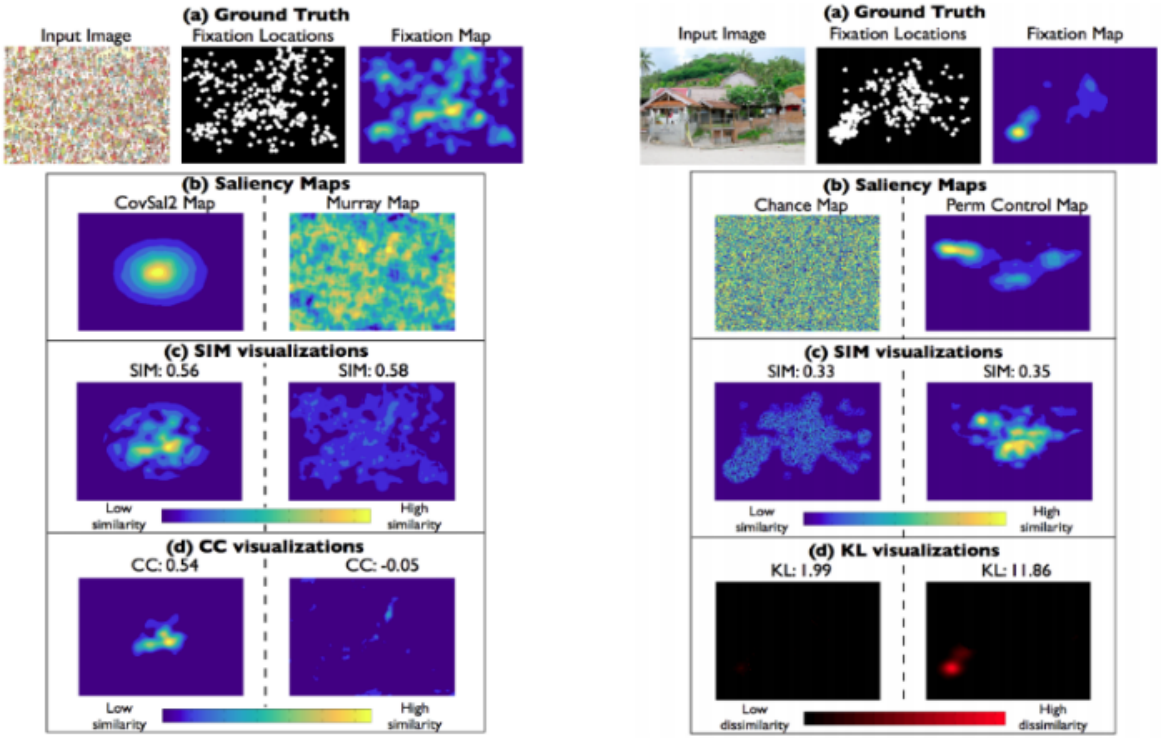

Figure 3.3 CC, SIM score comparison and SIM, KLDiv score comparison

The combination of KLdiv and CC in the loss function is motivated by the analysis presented in figure 3.4 (the analysis is inspired by [7]). We synthetically varied saliency predictions with respect to ground truth in order to quantify effects on the loss functions. Each row corresponds to varying a single parameter value of the prediction: (a) Variance, (b) location on a single mode, (c) location between two modes, (d) relative weights between two modes. The x and y-axis in the graphs spans the parameter range and values of loss functions respectively and the dotted red line corresponds to the ground truth.

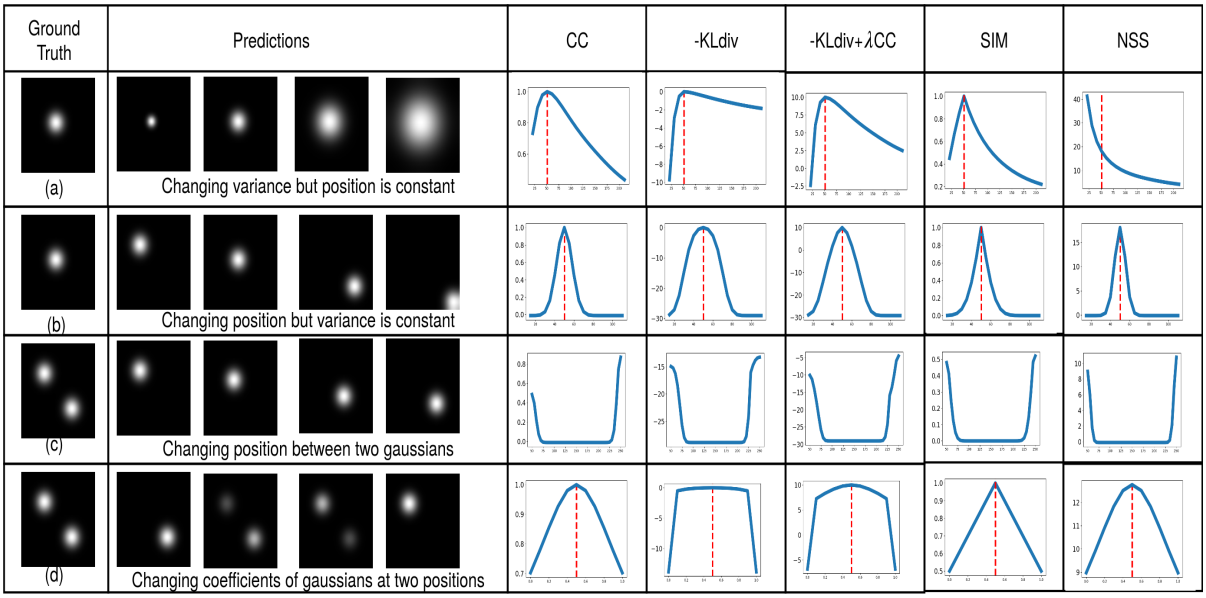

Figure 3.4 We synthetically varied saliency predictions w.r.t the ground truth in order to quantify effects on the loss functions. Each row corresponds to varying a single parameter value of the prediction: (a) Variance, (b) location on a single mode, (c) location between two modes, (d) relative weights between two modes. The x and y-axis spans the parameter range and values of loss functions respectively and the dotted red line corresponds to the ground truth (if applicable).

KLdiv being highly sensitive to false negatives results in steeper costs (consider case (a), (b) in Figure 3.4). Steeper costs lead to larger gradients, which are crucial in initial training (which motivates the use of KLdiv or its variants as a backbone of loss function). CC as it is symmetric to false positives and false negatives gives typical behavior in each scenario of Figure 3.4. The combination provides appropriate behavior in each of the studied scenarios while maintaining steeper costs (scenarios (a), (d) (Figure 3.4). We also explore the use of Normalized Scanpath Saliency (NSS) as a loss function in the ablation studies. Table 3.2 empirically shows Kldiv and CC combination loss gives best validation metrics.

Loss Functions	CC	KLdiv	NSS	AUC	SIM
KL	0.904	0.223	1.935	0.870	0.797
KL + CC	**0.906**	**0.192**	1.925	0.871	**0.798**
KL + CC + NSS	0.900	0.204	**1.998**	**0.872**	0.794

Table 3.2 SimpleNet's validation results on SALICON with various Loss Functions

Considering the above loss function, input features are varied with various pre-trained architectures such as VGG-16, ResNet-50, DenseNet-161 and PNASNet-5. The quantitative results in table 3.3 shows PNASNet giving better results than the others.

Models	CC	NSS	KLdiv	AUC	SIM
VGG-16	0.871	1.863	0.238	0.864	0.772
ResNet-50	0.895	1.881	0.211	0.868	0.786
DenseNet-161	0.902	**1.930**	0.210	0.87	0.795
PNASNet-5	**0.907**	1.926	**0.193**	**0.871**	**0.797**

Table 3.3 SimpleNet's validation results on SALICON with various Encoders

3.2 MDNSal

The overall architecture of the MDNSal is shown in Figure 3.5. The network is inspired by the literature on Mixture Density Networks [3]. MDNSal is a parametric model which gives a compressed representation, allowing faster processing at the application level.

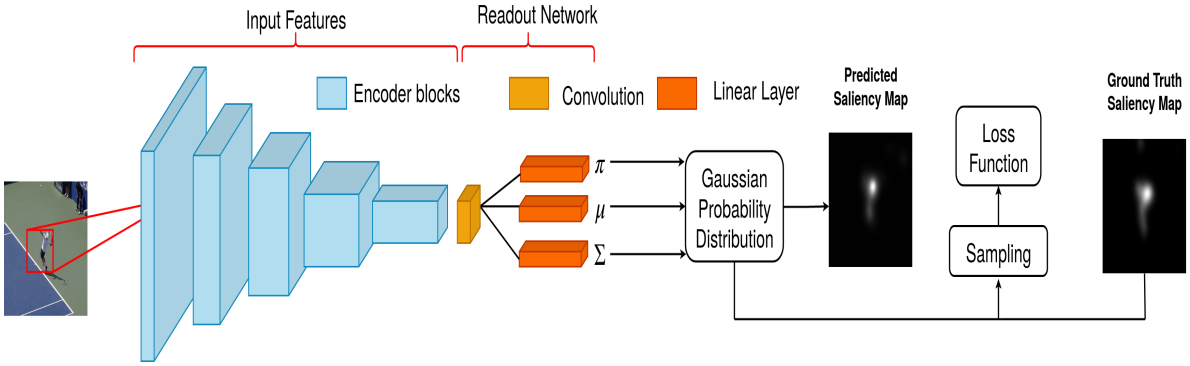

Figure 3.5 MDNSal Architecture

3.2.1 Input features

Similar to SimpleNet; we apply transfer learning from pre-trained image classification networks. The fine-tuning of the pre-trained features is extremely crucial in MDNSal and leads to significant performance improvements.

3.2.2 Multi-level integration

MDNSal only uses the features from the last convolutional layer of the pre-trained networks and is devoid of multi-level integration. Since the outputs are parameters instead of a per-pixel map, the multi-level features do not play a significant role.

3.2.3 Readout architecture

The readout architecture consists of a convolutional layer to reduce the number of channels followed by a ReLU. The output is then passed to three parallel fully connected layers predicting mixture weight (π), mean (μ), and the covariance matrix (Σ) for each Gaussian. For C mixtures, the sizes of the output layers are C, $C \times 2$ and $C \times 2$ for π, μ and Σ respectively. We predict only the diagonal elements of the covariance matrix Σ.

We also relax the constraint of only predicting the diagonal values of the covariance matrix. We predict the full covariance matrix with positive-definite constraints, which is necessary to compute the loss. To enforce this constraint we adopt the method by [13], where Cholesky decomposition is used to calculate covariance matrix. If A is the precision matrix and L is the lower traingular matrix, then

$$\begin{aligned} A &= LL^T \\ \Sigma^{-1} &= A \end{aligned} \tag{3.3}$$

We predict lower triangular matrix(L) and get covariance matrix using equations 3.3. Using full covariance matrix however did not give any visible improvements (CC remained same as 0.899 on SALICON validation set). Hence, we use the diagonal approximation in all the remaining experiments on MDNSal.

As the number of gaussians(C) are also variable, we change the number of gaussians to understand it's impact. The results are presented in Table 3.4 and it is observed that using 32 Gaussians gives best performance and thus we use the same number of mixtures in the later experiments.

3.2.4 Loss function

We define Negative log-likelihood (NLL) loss function to train the parameters of the Gaussians as follows:

No of Gaussians(C)	CC	KLdiv	NSS	AUC	SIM
8	0.882	0.256	1.849	0.864	0.778
16	0.892	0.240	1.881	0.867	0.787
24	0.895	0.233	1.887	0.867	0.789
32	**0.899**	**0.224**	**1.892**	**0.868**	**0.797**
48	0.896	0.231	1.889	**0.868**	0.790
64	0.896	0.230	**1.892**	**0.868**	0.790

Table 3.4 MDNSal's validation results on SALICON with various number of Gaussians

$$NLL(P,Q) = -\sum_{i} q_i \log(p_i + \epsilon). \tag{3.4}$$

Where i represents exhaustive sampling across spatial dimensions of the image, p_i is the likelihood of the sampled point to fit the distribution with C Gaussians, q_i is the corresponding ground truth value and ϵ is a small constant. p_i is further defined as follows:

$$p(x;\pi,\mu,\Sigma) = \sum_{c=1}^{C} \pi_c \frac{1}{\sqrt{(2\pi_c)^2|\Sigma_c|}} e^{-\frac{1}{2}(x-\mu_c)^T \Sigma_c{}^{-1}(x-\mu_c)}. \tag{3.5}$$

Similar to SimpleNet we use a combination of NLL and CC for training MDNSal.

3.3 Training and Experimental Setup

Both SimpleNet and MDNSal are evaluated on three datasets SALICON, MIT300 and CAT2000. Both the models are first trained on SALICON dataset which has 10000 training images and 5000 validation images. As MIT has only 1003 images, 903 images are used to fine tune the models trained on SALICON dataset and 100 images are used for validation. Similarly, CAT has 2000 images and 1800 images are used for training and 200 are used for validation. Both MIT and CAT are trained using cross validation and the best model is chosen for test submission. Validation results of the three datasets on both the models are in table 3.5.

We resize the input images into 256x256 resolution for both the models. We train SimpleNet for 10 epochs with learning rate starting from 1e-4 and reducing it after 5 epochs. MDNSal is trained for 50 epochs with learning rate 1e-4. We use 32 Gaussians ($C = 32$) in MDNSal. Backpropagation was performed using ADAM optimizer in both the networks. The model trained on SALICON was fine-tuned using MIT1003 and CAT2000. We submitted the test results for SALICON to LSUN17[1] and MIT300 test results to MIT Saliency Benchmark[2]. We only present validation results on the CAT2000 dataset. The test results comparison for both SALICON and MIT datasets can be seen in tables 3.6 and 3.7.

	SimpleNet					MDNSal				
	KLdiv	CC	AUC	NSS	SIM	KLdiv	CC	AUC	NSS	SIM
SALICON	0.193	0.907	0.871	1.926	0.797	0.217	0.899	0.868	1.893	0.797
MIT1003	0.558	0.786	0.907	2.870	0.626	0.634	0.779	0.904	2.814	0.624
CAT2000	0.256	0.895	0.883	2.400	0.758	0.293	0.889	0.878	2.329	0.751

Table 3.5 Validation results on all three studied datasets

	KLdiv↓	CC↑	AUC↑	NSS↑	SIM↑	IG↑	sAUC↑
EMLNET [21]	0.520	0.886	0.866	2.050	0.780	0.736	0.746
SAM-Resnet [12]	0.610	0.899	0.865	1.990	0.793	0.538	0.741
MSI-Net [28]	0.307	0.889	0.865	1.931	0.784	0.793	0.736
GazeGAN [9]	0.376	0.879	0.864	1.899	0.773	0.720	0.736
MDNSal (Ours)	0.221	0.899	0.865	1.935	0.790	0.863	0.736
SimpleNet (Ours)	0.201	0.907	0.869	1.960	0.793	0.880	0.743

Table 3.6 Our proposed models performance on the SALICON TEST(Red values represent the best and blue represent the second best).

3.4 Comparison with state of the art

We quantitatively compare our models with state of the art on SALICON and MIT300 test sets. Table 3.6 shows the results on the SALICON dataset in terms of KLdiv, CC, AUC, NSS, SIM, IG, and sAUC metrics. SimpleNet gives consistent results on all seven metrics. It achieves the best performance on five different metrics and outperforms state of the art by a large margin on KLdiv and IG. We

[1]http://salicon.net/challenge-2017/
[2]http://saliency.mit.edu

	KLdiv↓	CC↑	AUC↑	NSS↑	SIM↑	sAUC↑	EMD↓
EMLNET [21]	0.84	0.79	0.88	2.47	0.68	0.70	1.84
DeepGaze2 [32]	0.96	0.52	0.88	1.29	0.46	0.72	3.98
SALICON [19]	0.54	0.74	0.87	2.12	0.60	0.74	2.62
DPNSal [41]	0.91	0.82	0.87	2.41	0.69	0.74	2.05
DenseSal [40]	0.48	0.79	0.87	2.25	0.67	0.72	1.99
DVA [54]	0.64	0.68	0.85	1.98	0.58	0.71	3.06
MDNSal (Ours)	0.47	0.78	0.86	2.25	0.67	0.71	1.96
SimpleNet (Ours)	0.42	0.79	0.87	2.30	0.67	0.71	2.06

Table 3.7 Our proposed models performance on the MIT Saliency Benchmark.

are third-best in NSS metric; however, if crucial, that can be compensated by adding an NSS loss, as indicated in the ablation studies. Although parametric, surprisingly, MDNSal also gives competent performance across various metrics (only second to SimpleNet on four metrics). SimpleNet and MDNSal also achieve state of the art performance on MIT300 test dataset, as shown in Table 3.7. SimpleNet gives the best results on KLdiv and is competent in all other metrics. MDNSal gives a similar performance with the second-best results on KLdiv and EMD.

The work by [7] recommends CC as one of the ideal metrics to report, as it makes limited assumptions about input format and treats both false positives and negatives symmetrically. They further suggest KL and IG as good choices concerning benchmark intended to evaluate saliency maps as probability distributions. Both our models give a leading performance on KLdiv and CC on both MIT300 and SALICON test sets, which makes them an ideal choice for the task of saliency prediction.

We qualitatively compare results of the proposed models with other state of the art methods. The results on couple of images from MIT300 test dataset are shown in Figure 3.6. The ground truth images are chosen from the carefully curated set in [8]. Our model performs well both in terms on coverage (predicting all the salient regions), accurate localization and the relative order of importance.

3.5 Ablation Analysis

We examine the effects of (a) changing the input feature, (b) using different combinations of the loss function, and (c) the significance of hierarchy. The analysis is made using SimpleNet model on SALICON validation set. Table 3.3 illustrates the results by varying the pre-trained network for the input

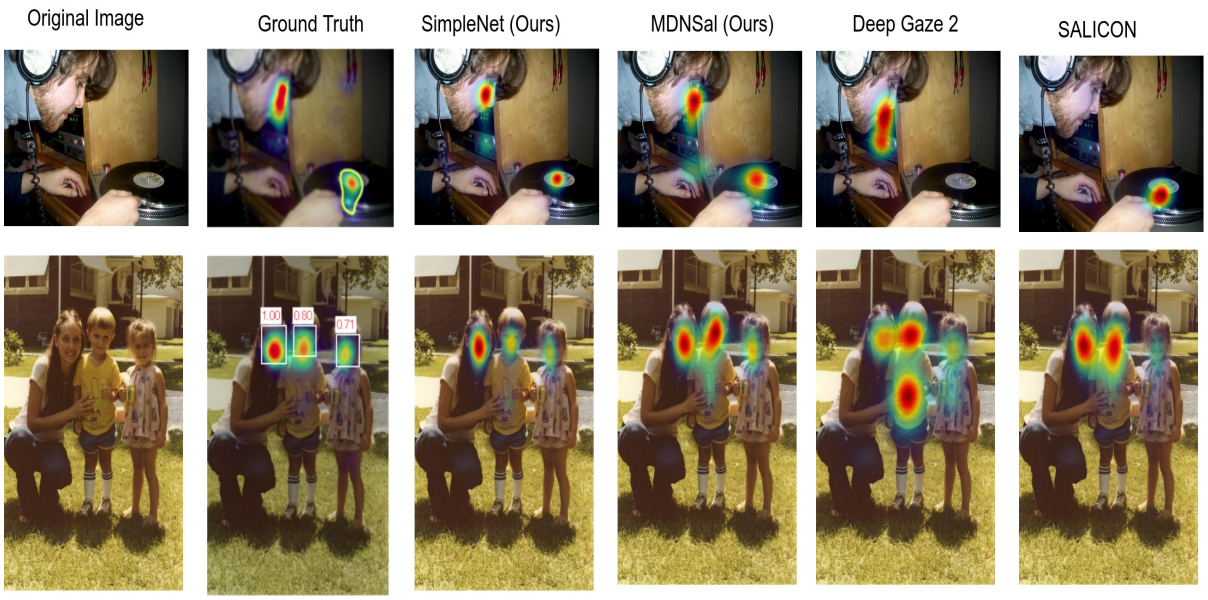

Figure 3.6 Examples of predicted saliency maps. Both images are taken from MIT300 test set and ground truth images are taken from [8]. We compare results of the proposed SimpleNet and MDNSal models with other state of the art approaches. First row (action): SimpleNet and MDNSal accurately predicts both the person's face and where he is looking (indicated by yellow boundary). In contrast other models miss out on either the action or the face. Second row (Faces with relative importance): our model gives accurate localization on all three faces and preserves relative importance.

feature component. PNASNet-5 achieves the best overall results and is used as the backbone for all the following experiments. Ablation results by adding CC and NSS to the KLdiv loss are presented in Table 3.2. Adding CC improves the performance over just using KLdiv loss (on both KL and CC metrics). Higher performance on the NSS metric can be achieved by adding an NSS term to the loss; however, it brings minor reductions in the KLdiv and CC metric. To keep things minimal, we use KLdiv+CC loss for later experiments.

We also explore the significance of multi-level integration by learning SimpleNet by just using the last conv layer of PNASNet-5. The CC drops to 0.89 from 0.907, and KLdiv increases to 0.22 from 0.193, indicating the importance of multi-level integration. Finally, the results of the validation set on all three datasets are presented in Table 3.5. We also explore the optimiser influence in training and explore three different optimisers Adabound, Adamod, and Adam. Adam is fixed for training as per table 3.8

We perform another set of experiments on MDNSal. The first ablation experiment is aimed to understand the impact of changing the number of Gaussians(C). The results are presented in Table 3.4 and it is observed that using 32 Gaussians gives best performance and thus we use the same number of mixtures in the later experiments. As next experiment, we relax the constraint of only predicting the diagonal values of the covariance matrix. We predict the full covariance matrix with positive-definite constraints, which is necessary to compute the loss. To enforce this constraint we adopt the method by [13], where we predict lower triangular matrix(L) and get covariance matrix using $A = LL^T$ and $\Sigma^{-1} = A$. Using full covariance matrix however did not give any visible improvements (CC remained same to 0.899 on SALICON validation set). Hence, we use the diagonal approximation in all the remaining experiments on MDNSal.

optimiser	CC	KLdiv	NSS
Adabound	0.9017	0.209	1.9303
Adamod	0.9042	0.2072	1.9277
Adam	**0.907**	**0.193**	**1.926**

Table 3.8 SimpleNet's validation results on SALICON different optimisers

Chapter 4

Video Saliency

This chapter aims in extension of image saliency to videos, explore the work that has been done, datasets used and experiments we have done. Unlike saliency prediction in images, videos have extra information like temporal aspect and audio which should be considered to achieve better results.

The below sections 4.1 talks about the datasets, 4.2 discusses various work that has been done and 4.3 talks about the experiments done on video saliency.

4.1 Datasets

There are two main datasets available for video saliency.

4.1.1 DHF1K

It contains 1000 videos with diverse content and length with eye-tracking annotations from 17 observers(10 males and 17 females) [55]. Each video is 30 fps with 640x360 spatial resolution. In total DHF1K has 582,605 frames with total duration of 19,420 seconds. The dataset is mainly classified into 7 categories which include human(daily activities, sports, social activities and art), animal, artifact and scenery. The dataset is split into 600 training, 100 validation and 300 test videos.

4.1.2 Hollywood

Hollywood movie dataset [38] has 1707 action videos. It contains 12 classes: answering phone, driving a car, eating, fighting, getting out of a car, shaking hands, hugging, kissing, running, sitting down, sitting up and standing up [37]. These actions are collected from a set of 69 Hollywood movies. The data set is split into a training set of 823 sequences and a test set of 884 sequences. There is no overlap between the 33 movies in the training set and the 36 movies in the test set. The data set consists

of about 487k frames, totaling about 20 hours of video.

4.1.3 UCF Sports

UCF Sports [45, 49] dataset consists of a set of actions collected from various sports which are typically featured on broadcast television channels. The dataset includes a total of 150 sequences with the resolution of 720 x 480. It includes 10 actions i.e diving, golf swing, kicking, lifting, riding horse, running, skateboarding, swing-bench, swing-side, walking.

4.2 Related Work

Recently there has been lot of work on video saliency. Some works relied on optical flow and LSTM [17] based networks. STSConvNet [1] uses two stream network and processes spatial and temporal streams independently, with temporal features extracted using optical flow. OM-CNN [22] first extracts spatial and temporal features from YOLO [44] and FlowNet [14] subnets, which represent objectness and motion respectively, and feed them into a twolayer LSTM. RMDN [2] extracts spatio-temporal features using C3D [52] whose output becomes the input of a LSTM network. Then a linear layer projects the LSTM representation to the parameters of a Gaussian mixture model. ACL-Net [55] has attention modelling where attention is trained on static saliency data(SALICON). It uses attention and learns temporal information using LSTM.

SalEMA and SalCLSTM are proposed in [34], by modelling and comparing temporal part differently. The architecture follows an encoder-decoder adopted from SALGAN [42] and processes the temporal recurrence. The temporally aware component into the SalBCE network. This is either the addition of a ConvLSTM layer or an exponential moving average (EMA) applied on a pre-existing convolutional layer. Tased-Net [39] also uses encoder decoder architecture where encoder is S3D [56] trained on kinetics dataset [25] and uses auxilary pooling for unpooling layers in the decoder.

4.3 Experiments

4.3.1 Experiment 1

This experiment is to use image saliency prediction for videos by learning temporal information by using LSTM(Long Short Term Memory) Network. LSTM has input gate(i_t), forget gate(f_t), output gate(o_t), cell state(c_t) and hidden state(h_t). At each time stamp t LSTM takes the previous cell state c_{t-1}, hidden state h_{t-1} and outputs current cell state(c_t) and hidden state(h_t).

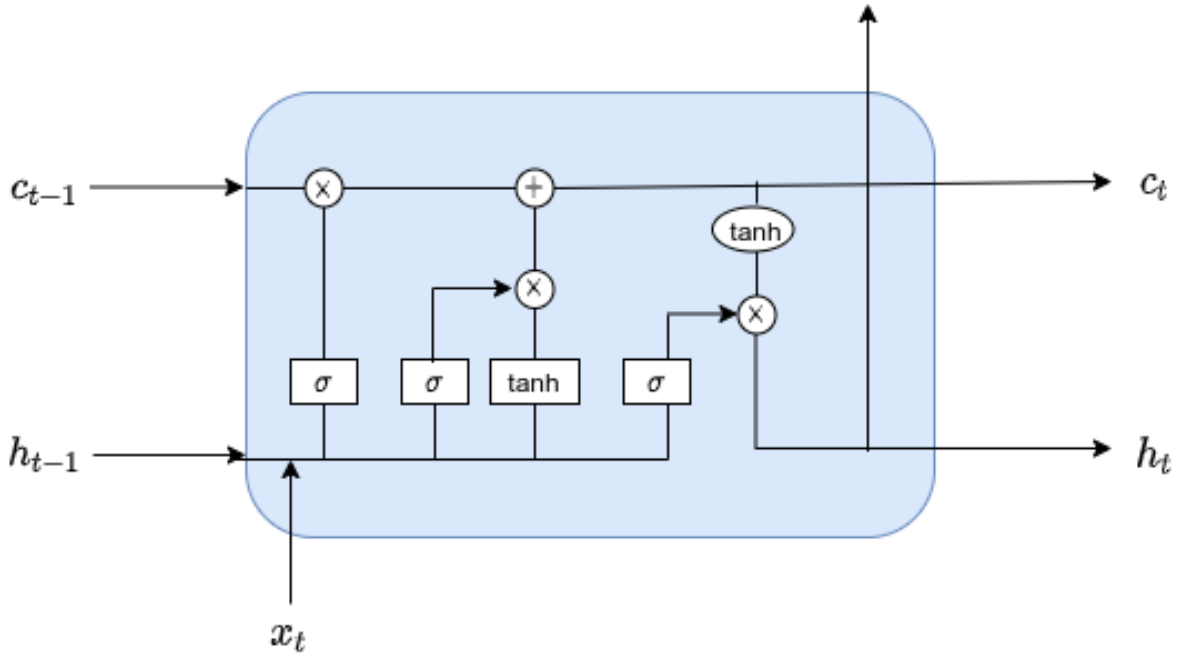

Figure 4.1 LSTM Cell

The following are the equations(4.1) showing computations of LSTM cell.

$$\begin{aligned} f_t &= \sigma_g(W_f x_t + U_f h_f + b_f) \\ i_t &= \sigma_g(W_i x_t + U_i h_t + b_i) \\ o_t &= \sigma_g(W_o x_t + U_o h_f + b_o) \\ c_t &= f_t \cdot c_{t-1} + \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \\ h_t &= o_t \cdot \sigma_h(c_t) \end{aligned} \tag{4.1}$$

This model has SimpleNet with LSTM [17] before the last convolutional layer. LSTM will have 256x256x128 channel input from SimpleNet and output of LSTM will have same resolution. The last convolutional layers after LSTM reduces the channel dimension to one. A set of 32 frames are sent each time covering all the frames in all videos for training. The loss function used for training this model is KLdiv.

4.3.2 Experiment2

This is encoder decoder based architecture with 3D convolutional layers. The encoder is S3D [56] model which is trained for action classification on kinetics dataset. Kinectics dataset is collected

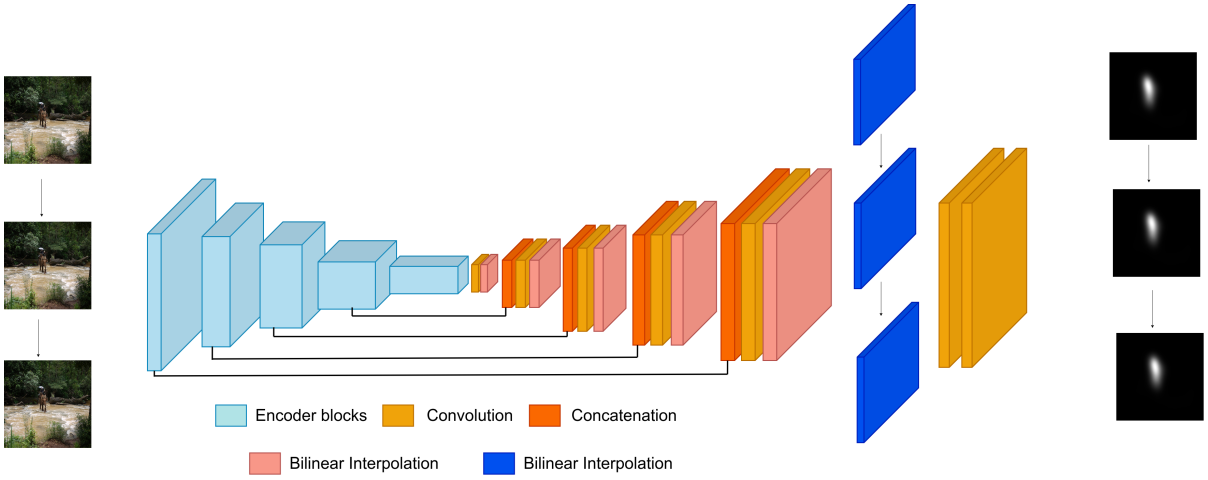

Figure 4.2 This architecture has LSTM after some decoder layers of SimpleNet.

from youtube videos and has 240k samples and 400 classes. S3D has been trained on mini kinectics dataset [25] which has 80k samples and 200 classes with each class having 200 samples.

The architecture in figure 4.3 is inspired from TasedNet [39]. Encoder blocks i.e base1, base2, base3 and base4 consists of set of convolutional layers and pooling layers to reduce the dimension of the frames and extract features. Decoder blocks i.e., Convspts(1,2,3,4) consists of transpose convolutions and increases the spatial size of the features. Each decoder layer has an input of previous decoder layer concatenated with the corresponding spatial dimension features from the encoder blocks, with concatenation along the time dimension.

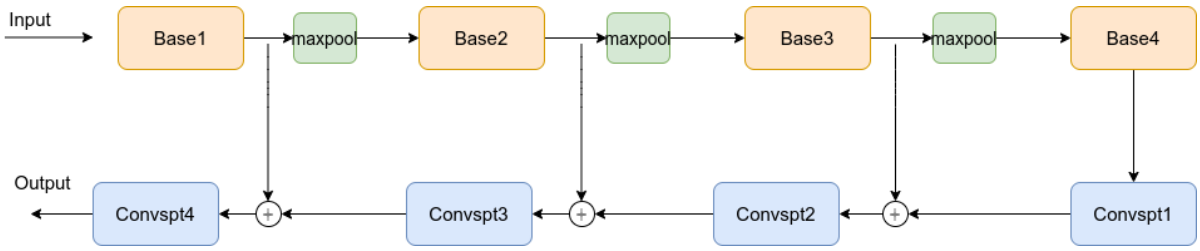

Figure 4.3 Experiment 2 takes 32 frames as input and gives output as one frame for the 32nd frame. Base 1,2,3 and 4 are from S3D model and Convspts are decoder blocks

Exp2 takes 32 frames as input and outputs one frame which corresponds to the output of 32nd frame in the input. So, all the saliency maps for a video are generated by sliding window fashion. During training, A set of 32 consecutive frames are selected from random start point from all the videos for

every epoch. The loss function used for training this model is KLdiv

Model	CC	KLdiv	NSS
Tased-Net	0.481	2.465	**2.706**
Exp1	0.457	2.214	2.584
Exp2	**0.507**	**1.956**	2.613

Table 4.1 Validation results on DHF1K dataset

Experiment 2 has shown better results than Experiment 1 as shown in the table 4.1. Exp2 after adding skip connections performed better than Tased-Net in CC and KLdiv.

Chapter 5

Conclusions

In this thesis, we identify four key components of the saliency detection architectures and analyze how the previous literature has approached the individual components. The analysis helps to explore agreements, redundancies, gaps, or need for optimization over these components. Using that as a basis, we propose two novel architectures called SimpleNet and MDNSal. SimpleNet improves upon the encoder-decoder architectures, and MDNSal opens up a new paradigm of parametric modeling. Both models are devoid of complexities like prior maps, multiple input streams, or recurrent units and still achieve the state of the art performance on public saliency benchmarks. Our work suggests that the way forward is not necessarily to design more complex architectures but a modular analysis to optimize each component and possibly explore novel (and simpler) alternatives.

Related Publications

Navyasri Reddy, Samyak Jain, Pradeep Yarlagadda, Vineet Gandhi, 2020. "Tidying Deep Saliency Prediction Architectures". International Conference on Intelligent Robots and Systems (IROS 2020).

Bibliography

[1] C. Bak, A. Kocak, E. Erdem, and A. Erdem. Spatio-temporal saliency networks for dynamic saliency prediction. *IEEE Transactions on Multimedia*, 20(7):1688–1698, 2017.

[2] L. Bazzani, H. Larochelle, and L. Torresani. Recurrent mixture density network for spatiotemporal visual attention. *arXiv preprint arXiv:1603.08199*, 2016.

[3] C. M. Bishop. Mixture density networks. 1994.

[4] A. Borji and L. Itti. Scene classification with a sparse set of salient regions. In *2011 IEEE International Conference on Robotics and Automation*, pages 1902–1908. IEEE, 2011.

[5] N. D. Bruce and J. K. Tsotsos. Saliency, attention, and visual search: An information theoretic approach. *Journal of vision*, 9(3):5–5, 2009.

[6] Z. Bylinskii, T. Judd, A. Borji, L. Itti, F. Durand, A. Oliva, and A. Torralba. Mit saliency benchmark, 2015.

[7] Z. Bylinskii, T. Judd, A. Oliva, A. Torralba, and F. Durand. What do different evaluation metrics tell us about saliency models? *IEEE TPAMI*, 41(3):740–757, 2018.

[8] Z. Bylinskii, A. Recasens, A. Borji, A. Oliva, A. Torralba, and F. Durand. Where should saliency models look next? In *ECCV*, 2016.

[9] Z. Che, A. Borji, G. Zhai, X. Min, G. Guo, and P. Le Callet. How is gaze influenced by image transformations? dataset and model. *IEEE TIP*, 2019.

[10] L. Chen, P. Huang, and Z. Zhao. Saliency based proposal refinement in robotic vision. In *2017 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pages 85–90. IEEE, 2017.

[11] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara. A deep multi-level network for saliency prediction. In *ICPR*, 2016.

[12] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara. Predicting human eye fixations via an lstm-based saliency attentive model. *IEEE TIP*, 27(10):5142–5154, 2018.

[13] G. Dorta, S. Vicente, L. Agapito, N. D. Campbell, and I. Simpson. Structured uncertainty prediction networks. In *CVPR*, 2018.

[14] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015.

[15] S. Frintrop and M. Kessel. Most salient region tracking. In *2009 IEEE International Conference on Robotics and Automation*, pages 1869–1874. IEEE, 2009.

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[17] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[18] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *CVPR*, 2017.

[19] X. Huang, C. Shen, X. Boix, and Q. Zhao. Salicon: Reducing the semantic gap in saliency prediction by adapting deep neural networks. In *ICCV*, 2015.

[20] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE TPAMI*, (11):1254–1259, 1998.

[21] S. Jia and N. D. Bruce. Eml-net: An expandable multi-layer network for saliency prediction. *arXiv:1805.01047*, 2018.

[22] L. Jiang, M. Xu, and Z. Wang. Predicting video saliency with object-to-motion cnn and two-layer convolutional lstm. *arXiv preprint arXiv:1709.06316*, 2017.

[23] M. Jiang, S. Huang, J. Duan, and Q. Zhao. Salicon: Saliency in context. In *CVPR*, 2015.

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

[25] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[26] C. Koch and S. Ullman. Shifts in selective visual attention: towards the underlying neural circuitry. In *Matters of intelligence*, pages 115–141. Springer, 1987.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[28] A. Kroner, M. Senden, K. Driessens, and R. Goebel. Contextual encoder-decoder network for visual saliency prediction. *arXiv 1902.06634*, 2019.

[29] S. S. Kruthiventi, K. Ayush, and R. V. Babu. Deepfix: A fully convolutional neural network for predicting human eye fixations. *IEEE TIP*, 26(9):4446–4456, 2017.

[30] M. Kümmerer, L. Theis, and M. Bethge. Deep gaze i: Boosting saliency prediction with feature maps trained on imagenet. *arXiv:1411.1045*, 2014.

[31] M. Kummerer, T. S. Wallis, and M. Bethge. Saliency benchmarking made easy: Separating models, maps and metrics. In *ECCV*, 2018.

[32] M. Kummerer, T. S. Wallis, L. A. Gatys, and M. Bethge. Understanding low-and high-level contributions to fixation prediction. In *ICCV*, 2017.

[33] G. Li and Y. Yu. Visual saliency based on multiscale deep features. In *CVPR*, 2015.

[34] P. Linardos, E. Mohedano, J. J. Nieto, N. E. O'Connor, X. Giro-i Nieto, and K. McGuinness. Simple vs complex temporal recurrences for video saliency prediction. *arXiv preprint arXiv:1907.01869*, 2019.

[35] N. Liu and J. Han. A deep spatial contextual long-term recurrent convolutional network for saliency detection. *IEEE TIP*, 27(7):3264–3274, 2018.

[36] N. Liu, J. Han, T. Liu, and X. Li. Learning to predict eye fixations via multiresolution convolutional neural networks. *IEEE transactions on neural networks and learning systems*, 29(2), 2016.

[37] M. Marszalek, I. Laptev, and C. Schmid. Actions in context. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2929–2936. IEEE, 2009.

[38] S. Mathe and C. Sminchisescu. Actions in the eye: Dynamic gaze datasets and learnt saliency models for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(7):1408–1424, 2014.

[39] K. Min and J. J. Corso. Tased-net: Temporally-aggregating spatial encoder-decoder network for video saliency detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2394–2403, 2019.

[40] T. Oyama and T. Yamanaka. Fully convolutional densenet for saliency-map prediction. In *ACPR*, 2017.

[41] T. Oyama and T. Yamanaka. Influence of image classification accuracy on saliency map estimation, 2018.

[42] J. Pan, E. Sayrol, X. G.-i. Nieto, C. C. Ferrer, J. Torres, K. McGuinness, and N. E. O'Connor. Salgan: Visual saliency prediction with adversarial networks. In *CVPR Scene Understanding Workshop (SUNw)*, 2017.

[43] A. Rasouli, P. Lanillos, G. Cheng, and J. K. Tsotsos. Attention-based active visual search for mobile robots. *Autonomous Robots*, pages 1–16, 2019.

[44] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[45] M. D. Rodriguez, J. Ahmed, and M. Shah. Action mach a spatio-temporal maximum average correlation height filter for action recognition. In *2008 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2008.

[46] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

[47] B. Schauerte, J. Richarz, and G. A. Fink. Saliency-based identification and recognition of pointed-at objects. In *IROS*, 2010.

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

[49] K. Soomro and A. R. Zamir. Action recognition in realistic sports videos. In *Computer vision in sports*, pages 181–208. Springer, 2014.

[50] M. Staudte and M. W. Crocker. Visual attention in spoken human-robot interaction. In *2009 4th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 77–84. IEEE, 2009.

[51] B. W. Tatler. The central fixation bias in scene viewing: Selecting an optimal viewing position independently of motor biases and image feature distributions. *Journal of vision*, 7(14):4–4, 2007.

[52] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015.

[53] 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*, 9(7):4–4, 2009.

[54] W. Wang and J. Shen. Deep visual attention prediction, 2017.

[55] W. Wang, J. Shen, F. Guo, M.-M. Cheng, and A. Borji. Revisiting video saliency: A large-scale benchmark and a new model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4894–4903, 2018.

[56] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 305–321, 2018.

[57] J. Zhang and S. Sclaroff. Saliency detection: A boolean map approach. In *ICCV*, 2013.

[58] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le. Learning transferable architectures for scalable image recognition. In *CVPR*, 2018.