

Learning Driver Gaze

by

Anying Li

Submitted to the Department of Electrical Engineering and Computer
Science

in partial fulfillment of the requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science
at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2017

© Massachusetts Institute of Technology 2017. All rights reserved.

Author
Department of Electrical Engineering and Computer Science
August 18, 2017

Certified by
Wojciech Matusik
Associate Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Certified by
Antonio Torralba
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by
Christopher J. Terman
Chairman, Masters of Engineering Thesis Committee

Learning Driver Gaze

by

Anying Li

Submitted to the Department of Electrical Engineering and Computer Science
on August 18, 2017, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Driving is a singularly complex task that humans manage to perform successfully day in and day out, guided only by what their eyes can see. Given how prevalent, complex, and not to mention dangerous driving is, it's surprising that we don't really understand how drivers actually use vision to drive. The release of a large scale driving dataset with eye tracking data, DrEyeVe [1], makes analyzing the role of vision feasible. In this thesis, we 1) study the impact of various external features on driver attention, and 2) present a two-path deep-learning model that exploits both static and dynamic information for modeling driver gaze. Our model shows promising results against state-of-the-art saliency models, especially on sequences when the driver is not just looking straight ahead on the road. This model enables us to estimate important regions that the driver should be aware of, and potentially allows an automatic driving assistant to alert drivers of hazards on the road they haven't seen yet.

Thesis Supervisor: Wojciech Matusik

Title: Associate Professor of Electrical Engineering and Computer Science

Thesis Supervisor: Antonio Torralba

Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

I would like to thank my thesis supervisors Professors Wojciech Matusik and Antonio Torralba for their expertise and guidance on this project.

Special thanks goes to Petr Kellnhofer for his mentorship, advice, humor, and support. I truly appreciate everything you've done for me this past year.

Last, but certainly not least, I want to thank my roommates, friends, and family for their never-ending support and care. You all bring me more happiness than I could ever express in words here.

THIS PAGE INTENTIONALLY LEFT BLANK

Contents

1	Introduction	15
2	Related Work	19
2.1	Image Saliency	19
2.2	Video Saliency	21
2.3	Driver Attention	22
3	Analysis of DrEyeVe Data	25
3.1	Dataset Details	25
3.2	Average Gaze	27
3.3	Gaze and Speed	29
3.4	Gaze and Driving Maneuvers	31
3.5	Gaze and Objects	35
3.6	Conclusions	40
4	Learning Driver Gaze	43
4.1	Objective	43
4.2	Model Overview	43
4.3	Network Architecture	45
4.4	Training	47
5	Experiments	49
5.1	Evaluation Strategy	49
5.2	Models	50

5.3	Results	51
5.3.1	Comparison with DrEyeVe Model	53
5.4	Performance Breakdown	55
5.4.1	Qualitative	55
5.4.2	Prediction Size Study	58
5.4.3	Pathway Visualization	60
6	Conclusion and Future Work	63

List of Figures

3-1	A sample of data from the DrEyeVe data set. All samples here are from recording '06'. See Section 3.1 for a more detailed description of the different data types.	26
3-2	Video frames taken in a variety of different environments, covering all 9 environments present in the dataset.	27
3-3	Dataset-wide averages of all 555,000 RGB frames and saliency maps, and the overlay of the two means. Average saliency is scaled to be between [0, 1]. The overlay shows that the average saliency is very center-biased and tightly centered at the vanishing point of the road.	28
3-4	Average saliency when traveling at speeds within 6 different intervals. The mean is shown as a red circle, and the standard deviation is represented by the red line. From the figure, we can see that as speed increases, the saliency "ring" converges much more tightly around the vanishing point in the road. Note that all averages are scaled to be between [0, 1].	29
3-5	Pairwise F-tests between variances for pairs of speed intervals. Green implies significance at $\alpha = 0.05$, yellow for significant at $\alpha = 0.10$, and red to mean no significance. Variances between intervals $\leq 30\text{km/h}$ and intervals $\geq 30\text{km/h}$ are all statistically different.	31
3-6	Illustration of the trajectory segmentation method. Trajectories are segmented into 10 second intervals by increments of 2 seconds – [0, 10], [2, 12], and so on.	32

3-7	Fig. 3-7a illustrates how features are calculated for each segment. These features are then used in k -means clustering (we chose $k = 10$). Fig. 3-7b visualizes the resulting cluster space.	33
3-8	Visualization of average saliency per cluster. We note that y -mean does not change much between clusters, but x -mean shifts left for clusters with larger $-x$ displacements (4, 9, and 10) and shifts right for clusters with larger $+x$ displacements (1, 3, and 6).	34
3-9	Pairwise T-tests between cluster x -means. Green indicates statistical significance at $\alpha = 0.05$, yellow indicates statistical significance at $\alpha = 0.10$, and red means no significance. Clusters 1, 3, and 10 have statistically significant differences in x -means from many other clusters.	35
3-10	Selected examples of running SegNet on frames from the world camera. Fig. 3-10a contains a legend mapping colors to object categories. Fig. 3-10b shows examples of environments in which SegNet works well while Fig. 3-10c shows examples of environments in which SegNet works poorly. SegNet tends to not work well on images taken at night, when it's raining, and of certain countryside roads.	37
3-11	Average proportions that drivers are focused on various semantic categories under two different proportion functions, p_{fix} and p_{tot} (see Section 3.5 for their definitions); solid bars represent p_{fix} values and cross-hatched bars represent p_{tot} values. Fig. 3-11a contains averages across the whole dataset, while Fig. 3-11b contains averages across the filtered dataset, the portion with qualitatively better segmentations. The p_{fix} bars (solid) suggest that drivers spend most of their time focusing on other vehicles, the road, trees, and the sky. The p_{tot} (hatched) bars suggest that pedestrians, bikes, and road signs are the most significant to drivers.	39

4-1	A high level diagram of DualNet. Our network is composed of two main branches, the static branch and the dynamic branch, the results of which are combined to yield the final prediction. We hypothesize that the static branch will learn salient color and luminance features while the dynamic branch will learn salient motion features. More details on specific layer architectures for each branch can be found in Tbl. 4.1.	45
5-1	Visualization of the results of our models and the baselines. Column 1 is the input image, Column 2 is the ground truth saliency map, Columns 3 and 4 are results from our models, and Columns 5 and 6 are results from the baselines. We can see that our models generally produce tighter, more well localized maps than the baselines.	54
5-2	Some examples of salient regions that our model (DN-ResNet) has learned very well. These include the identifying the vanishing point in the road, splits in the road, and pertinent vehicles.	56
5-3	Some tasks our model (DN-ResNet) struggles with. These include predicting turns in advance (before the car has physically begun turning) and identifying traffic lights/signs.	57
5-4	Normalized CC, KL, MAE, and AUC values for DN-CaffeNet with predictions at various scales. 40x24 predictions outperform smaller predictions on all 4 metrics, and there is no discernable performance gain from increasing prediction size past 40x24.	59
5-5	Visualization of performance of DN-CaffeNet with predictions of different sizes. The suffix of the model name is the size of the prediction. We can see that smaller predictions are usually just as well localized, but with a much wider radius due to the higher level of upsampling needed to scale the prediction back up to the original size of the input (320x192 in our case).	60

5-6 Visualization of the contributions of both the static and dynamic branches to the overall saliency prediction. The arrow on the images in Column 2 indicate the direction the car is traveling. This figure illustrates that the static branch has learned to spot salient objects to drivers (the road ahead, other vehicles, joining roads), while the dynamic branch has learned to predict a saliency map based on the direction the vehicle is travelling. . . . 61

List of Tables

3.1	Mean location and standard deviations for all saliency maps within each of the speed intervals. All values are calculated from pixels normalized between the range $[0, 1]$. The means are fairly consistent between intervals, but the standard deviation drops sharply as speed increases.	30
3.2	Mean location and standard deviations per cluster. All values are calculated from pixels normalized between the range $[0, 1]$. y -means do not change significantly, but x -means vary widely over the different clusters.	34
4.1	Modified layer parameters for each of the static saliency networks. Given a kernel size k and number of outputs o , a row is of the form $(k \times k, o)$, and stride is 1 unless otherwise specified. All layers are response normalized.	46
5.1	We evaluate our models against the baselines on both the whole testing test as well as the acting portion of that set (a manually annotated set by [36] that is sufficiently different from the mean). On AUC and CC, higher numbers are better, while on KL and MAE, lower numbers are better, as indicated by the direction of the arrows. On both datasets, our models outperform the baselines, and our models significantly outperform the mean on the acting dataset. See Section 5.3 for a more detailed discussion of the results.	51

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 1

Introduction

Every day, millions of us load ourselves up into two ton steel boxes, steering ourselves across networks of pavement at speeds upwards of 65mph (with others doing the same) so we can get where we need to go. We manage to accomplish this seemingly daunting task successfully and with apparent ease time and time again, guided only by what our eyes can see. The amount of visual information we must process while driving is quite remarkable – we must track other vehicles, look out for road signs, steer between the lanes, remain vigilant for pedestrians, and many other tasks, all virtually in parallel. Of course, when we don’t perform these tasks well, the consequences can be catastrophic; 35,000 people died in car crashes in 2015 in the US alone [15].

For how ubiquitous, complicated, and dangerous driving is, we don’t actually have much generalized knowledge about how drivers use vision. Most of the research up until now on visual attention while driving has been very task-specific, performed on simulated data, or only focused on high level comparisons between different groups of drivers (see section 2.3 for a more in-depth discussion). Even less has been done to build a truly generalized model of driver attention – i.e. a model that predicts where the driver is most likely looking (also known as *salient* regions) given what the driver is currently seeing. While saliency models have been cropping up in many other contexts [26, 27, 29, 46], no such models have emerged for driving saliency (with the exception of [36], a work completed contemporaneously with ours).

Building a good model for driver saliency has many wide-reaching implications. For

one, it could revolutionize how advanced driver-assistance systems (ADAS) monitor for risks on the road. Currently, most ADAS rely on monitoring hard-coded cues such as an object being too close to the vehicle or when the driver has spent too much time looking away from the road, and warns the user should any of those cues occur. Now, with a model of what is salient to drivers in any given scene, an ADAS can warn the driver if an object it deems important is not yet salient. This implies that it can selectively warn drivers of only hazards they will most likely miss. An ADAS can also combine that information with an estimate of where driver is currently looking (using head position as in [16]) to determine whether the driver has seen an important area. This has the potential to be a much better tracker of driver inattention (which the CDC labels as the leading cause of car accidents) than monitoring physical cues.

The narrowing down of an entire scene to only a few salient regions could also improve the performance of autonomous vehicles. As of today, autonomous vehicles are equipped with a huge range of sensors – cameras facing every direction, LIDAR, GPS, etc. – in order to perform a task humans can do using only vision. They must process all of this data before making any driving decisions. Perhaps knowledge of salient areas in the environment could help reduce the amount of information the vehicles must process or allow them to process visual information more like a human. This reduction of data size is also useful outside of the autonomous vehicle case: other applications include using knowledge of salient regions in an image for better compression, or to reduce the memory footprint when running large neural networks [35].

Given the consequences of the topic, we hypothesize that the fact that driver attention has not been studied in much depth is due in part to a lack of real world driving datasets that include gaze information. However, with the release of DrEyeVe [1] in 2016, a large scale dataset with eye-tracking data, analysis and modeling of driver gaze across a wide variety of driving scenarios suddenly became much more feasible. We take advantage of the DrEyeVe dataset to further understand how drivers use vision.

In this thesis, we present the following contributions:

- A quantitative study of how various external factors – specifically speed, driving maneuver, and objects in the environment – affects driver gaze.

- A novel multi-branch neural-network based model for driving saliency evaluated against various saliency models.
- A more in-depth assessment of what the network has learned by analyzing how it performs in different driving environments and exploring what each branch of the network specifically has learned.

We now turn to a more detailed outline of the thesis. In chapter 2, we introduce related works concerning visual attention during driving. Then, in chapter 3, we study the relationship between certain external factors and driver gaze. We wish to establish some general rules about how gaze patterns change under different driving scenarios. To this end, we analyze some basic features of driving scenarios such as vehicle speed, driving maneuver, and objects in the environment, and quantitatively assess their impact on driver attention.

In chapters 4 and 5, we discuss the model we trained for driving saliency prediction. We propose a two branch deep learning model that utilizes both spatial and temporal information for prediction in chapter 4. In chapter 5, we evaluate the performance of our model against various saliency baselines, and also analyze its performance under different driving scenarios. We show that our model has promising performance, especially in circumstances when the driver is not just looking straight ahead. Finally, in chapter 6, we discuss future work and conclude.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 2

Related Work

2.1 Image Saliency

Image saliency is the study of where a person is most likely to be paying attention – i.e. determining which areas are the most *salient* – in any given image. There are two main flavors of saliency: free-viewing and task-driven. As the names suggest, the former refers to areas of interest to a viewer when simply looking at the image (tracked in datasets like MIT300 [9], CAT2000 [6], and SALICON [22]), while the latter refers to areas of interest to a viewer who is attempting some objective (for example, a memory [8] or a search [46] task).

Determining which areas of an image would be salient to drivers is certainly a (task-driven) saliency problem, although an important caveat to note is that the DrEyeVe dataset differs from other saliency datasets in a critical way. In traditional saliency datasets, a saliency map for a given image is built using eye fixations from *multiple* viewers, whereas DrEyeVe only uses fixations from *one*. This implies that each saliency map in the DrEyeVe dataset is not an accurate representation of what an *average* driver would find salient in that scenario, but rather only what *one* driver did. Obviously, this limitation of the dataset is not really avoidable when capturing real world driving scenarios because it's not feasible to simulate the same environment for multiple drivers in the real world. We hope that even with this limitation, given all the attention maps collected across all drivers, our saliency model will be able to learn a defacto "average" saliency map for any given situation.

Traditionally, saliency has been modeled by analyzing various features of the image (see [5] for an overview). For example, in bottom-up approaches ([20, 51], etc.), salient regions are identified by analyzing color, contrast, intensity, discontinuities and other low-level image properties. Conversely, top-down modelling approaches ([23, 49], etc.) revolve around the idea that humans focus on salient *objects*. As a result, many use objects detectors for faces, text, and other salient objects, or use image segmentation and super-pixel methods during preprocessing. Saliency models built for task-driven saliency tend to incorporate more top-down features in order to encode salient objects to the task. For example, Torralba et al. [46] show that saliency prediction for viewers performing a search task (e.g. answering questions such as "where is the pedestrian in this image?") can be improved significantly by incorporating top-down features such as global scene context (e.g. the location of sidewalks, roads, etc.) over simply using traditional bottom-up approaches.

In the past few years though, neural networks have gained enormous traction in the computer vision community, including in saliency modeling. In fact, 14 of the top 15 submissions on the MIT300 benchmark use some form of deep learning. Many of these models including DeepGaze [27], DeepFix [26], and SALICON [19] use networks (or features learned from them) with good performance on the popular ILSVRC [42] image classification challenge. Others such as SAM [12] or DSCLRCN [30] use recurrent neural networks (RNNs) in conjunction with purely convolutional neural networks (CNNs) for continuous prediction refinement. Because saliency modeling is a "dense" prediction task – i.e. a label is predicted for every pixel –, most of these models also employ some sort of "deconvolution" strategy to recover dimensionality lost from pooling layers. This includes using dilated convolutional networks [50], including simple deconvolutional layers (more aptly described as "transposed" convolutional layers) [14], or more complicated schemes that involve combining feature maps at different scales during the deconvolution to incorporate information from finer scales, as in [31].

Aside from the performance gains, another pro of neural network-based approaches is that they're generally more versatile. Traditional models tend to be more inflexible because most models rely on hard-coded computations on pre-chosen image features. This implies that when the task changes or the dataset is updated, the model is frequently rendered

invalid. Adapting a neural network-based model to a new saliency dataset (perhaps one where viewers are completing a different task) can be as simple as retraining that model on the new data.

2.2 Video Saliency

Video saliency is a natural offshoot of image saliency – it is simply the areas of interest to a viewer when watching a video, either while free-viewing or when accomplishing some task. While their problem statements are very similar, video saliency differs from image saliency in a few important ways. First of all, the temporal aspect introduces motion into the equation. Human vision is naturally drawn to motion, especially to unexpected motion [39]. Secondly, each frame in a video is only viewed for a fraction of a second, while a still image can be viewed over a longer period of time. This causes video saliency to generally produce "tighter" saliency maps that are more focused on a single object [41].

Many of the initial video saliency methods simply added motion features (flicker and motion) to traditional bottom-up features used in image saliency (color, intensity, orientation, etc.) to produce updated saliency maps for video [38, 32]. One particular approach used two successive video frames as input into a two-branch model, one of which computes a static saliency map from color and lumination, while the other computes a dynamic saliency map from motion estimated features [33]. Our model uses a very similar architecture, although we use neural networks (and therefore, learned filters), rather than custom filters (described more in Chapter 4).

More sophisticated video saliency models add a level of temporal coherence on top of incorporating new motion features. Temporal coherence implies that the saliency map for a frame at time $t + 1$ must be consistent with the saliency map at time t , and so on. Rudoy et al. [41] enforce temporal coherence by explicitly conditioning the saliency map at $t + 1$ on the saliency map at t . Wang et al. [48] took a different route and instead proposes a spatiotemporal saliency energy function that makes it so that saliency values of different foreground/background objects should not change dramatically per frame.

In contrast to image saliency, there have been very few video saliency models that use

any deep learning components. The first of these we could find is [11], which trains a neural network on optical flow features and individual frames. Chaabouni et al. actually train a patch based saliency predictor, which is then assembled together to build a global saliency map. Bazzani et al. [4] present a more sophisticated CNN-LSTM model; CNN-LSTMs are an architecture where the outputs of a convolutional neural network (CNN) are fed into a long-short term memory network (LSTM). Their model, which they call RMDN, takes in 16 frames as input into a 3D-convolutional network, the results of which are used as features in an LSTM, which are finally used as parameters in a Gaussian mixture model that predicts the final saliency map. RMDN currently holds the state-of-the-art on a few video saliency benchmarks.

2.3 Driver Attention

Most of the research up until now on visual attention while driving has been only focused on high level comparisons between different groups of drivers or targeted to very specific driving scenarios. In the former category, there are studies like [47] on how visual scanning of the road changes between novice and advanced drivers, and [3] on how certain features of driver attention change in older drivers and how that impacts crash rates. In the latter category, there are studies on how gaze is affected by interactions with in-vehicle technologies (phone, GPS, music, etc.) [18], and also on where drivers look when steering on a winding road [28].

To the best of our knowledge, the closest we get to more generalized research on driver gaze can be found in [24] and [44]. In [44], Simon et al. build a simple model to identify the saliency of road signs. While their method only addresses road signs, it can be expanded to include other objects on the road. In [24], driver eye movements are analyzed when driving in different environments (day, night, and rain), although they use data from a driving simulator instead of the real world.

The one exception to this is the generalized saliency model released by Palazzi et al. [36], which is also trained on the DrEyeVe [1] dataset. Their work was done concurrently with ours, meaning there is some overlap in the methods we tried, especially on the data

analysis side. Palazzi et al.'s model features three main branches: one for RGB data, one for precomputed optical flow maps, and one for precomputed segmentations, the results of which are combined into a final saliency map. Each branch is fed with 16 frames of data in a video clip, which is then passed through a 3D-convolutional network. The authors use weight sharing and aggressive cropping to reduce overfitting. See Section 4.2 for more details on our model architecture and how it compares to theirs.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 3

Analysis of DrEyeVe Data

In this chapter, we explore how attention is affected by a variety of factors while driving. We wish to quantitatively assess whether conventional knowledge about driving holds up under data analysis, and to what extent. In particular, we will closely evaluate how speed, driving maneuvers, and objects in the environment affects what a driver chooses to focus on. To this end, we will analyze data from the DrEyeVe [1] dataset, the only sizable and publicly available driving dataset which includes eye tracking data to our knowledge.

3.1 Dataset Details

We first lay out the data provided in the DrEyeVe dataset. Driving data is collected from two sources: the *SMI ETG 2w* eye tracking glasses worn by the driver and a roof-mounted Garmin VirbX camera (with built-in GPS, accelerometer, and gyroscope). The different types of data gathered from these sources offered by DrEyeVe is itemized and described in depth below. Fig. 3-1 also contains samples of the different types of data.

- **Driver FOV:** The driver FOV is captured in 720p at 30fps by the scene camera on the eye tracker. This camera follows the driver through severe head pose changes because the camera is mounted directly on the glasses worn by the driver. An example frame captured by the ETG scene camera is shown in Fig. 3-1d.
- **Fixations:** The eye tracker predicts an (x, y) fixation point for each frame of the scene



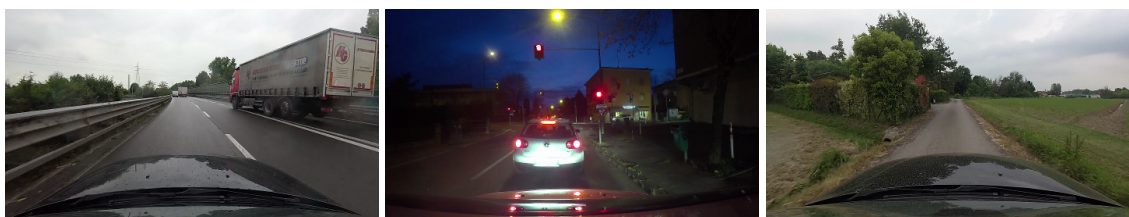
Figure 3-1: A sample of data from the DrEyeVe data set. All samples here are from recording '06'. See Section 3.1 for a more detailed description of the different data types.

camera.

- **World view:** The roof-mounted Garmin VirbX camera (heretofore referred to as the RGB camera) captures a 120° world view in 1080p at 25fps. The FoV of the VirbX camera is wide enough to include fixations from side glances, but not usually from severe head pose changes (such as turning to check blind spots when changing lanes). Fig. 3-1a shows an example frame from the RGB camera.
- **Saliency maps:** Saliency maps are constructed for each frame of the RGB camera. In order to gather fixation points with respect to the RGB camera instead of the ETG one, Alleto et al. [1] first learn a homography H between the ETG frame and the corresponding RGB frame. H is then used to project the fixation point from the ETG space to the RGB one. Each saliency map F_i is then constructed by aggregating the fixations from frames $i - 12, \dots, i, \dots, i + 12$ (a total of 25 frames) and then smoothing with a spatio-temporal Gaussian (see more details in Section 4 of [1]). An example saliency map is shown in Fig. 3-1b, and the overlay onto the corresponding RGB frame in Fig. 3-1c.

- GPS/speed/course data: Speed and heading are captured every frame (30fps), while GPS is captured approximately every second.

The dataset is meant to be representative many different driving conditions. As such, data is collected across a variety of different environments with different drivers. A few key characteristics of the dataset are described below:



(a) Morning, Rainy, Highway (b) Night, Sunny, Downtown (c) Evening, Cloudy, Country

Figure 3-2: Video frames taken in a variety of different environments, covering all 9 environments present in the dataset.

- Recordings: There are a total of 74 driving recordings, each 5 minutes long. Each video is taken at 25fps for a total of 555,000 total frames per type of data.
- Driving conditions: Data is collected across varying weather conditions, road types, and times of day. Each category is split into three subcategories: weather conditions are divided into sunny, rainy, and cloudy; road types are split into highway, downtown, and countryside; times of day are divided into morning, evening, and night. Fig. 3-2 contains examples of images taken of each category.
- Drivers: A total of 8 drivers alternate driving. While 8 is still a relatively small number, it does prevent the dataset from being biased towards one driver's habits.

3.2 Average Gaze

To get a general sense of where drivers look when driving, we first compute a dataset wide average of all the gaze maps. As shown in Figure 3-3, gaze while driving is very center-biased, tending towards the vanishing point of the road. These findings are consistent with the results from Land et al. in [28]; they discovered that drivers tend to look towards

the "tangent point" on the inside of each curve while driving on curved roads, which is equivalent to the vanishing point in the road.

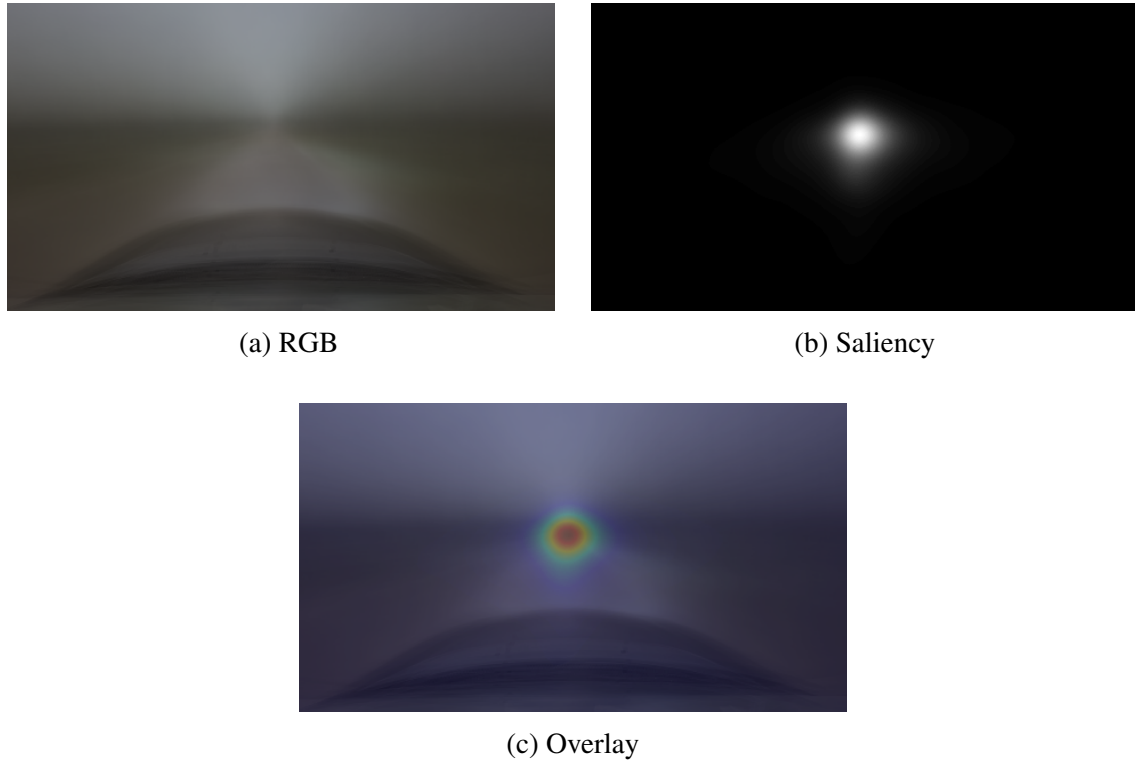


Figure 3-3: Dataset-wide averages of all 555,000 RGB frames and saliency maps, and the overlay of the two means. Average saliency is scaled to be between $[0, 1]$. The overlay shows that the average saliency is very center-biased and tightly centered at the vanishing point of the road.

This behavior also matches our expectations; in most circumstances, drivers will focus straight ahead on the road and rely on peripheral vision to detect any changes in the environment. Research finds that central vision (what you see straight ahead in focus) is actually only about three degrees of visual field. Peripheral vision covers a much larger area (about 50 degrees) and while objects are not as in focus, peripheral vision is very sensitive to motion and changes in lighting. Looking straight ahead on the road is the best use of those three degrees of central vision, as it allows drivers to be alerted of potential hazards on the road as early as possible.

3.3 Gaze and Speed

We now look into the relationship between speed and gaze. To do so, we adopt a strategy very similar to the one used by Palazzi et al. in [36], although we both qualitatively and quantitatively (with statistical tests) assess the results, only the former of which was done in [36].

We begin by bucketing saliency maps into six intervals: when the car is travelling under 15 km/h, between 15 km/h and 30 km/h, and so on. We then calculate average saliency maps for each speed bucket, the results of which are shown in Fig. 3-4. We also calculate mean pixel locations and standard deviations (using the saliency map values as weights) for the average saliency maps in each interval, which are listed in Tbl. 3.1. Note that all values are calculated after scaling pixel locations to the range $[0, 1]$. The mean and standard deviation are then plotted on the average saliency maps in 3-4, represented by a red circle and a red line respectively.

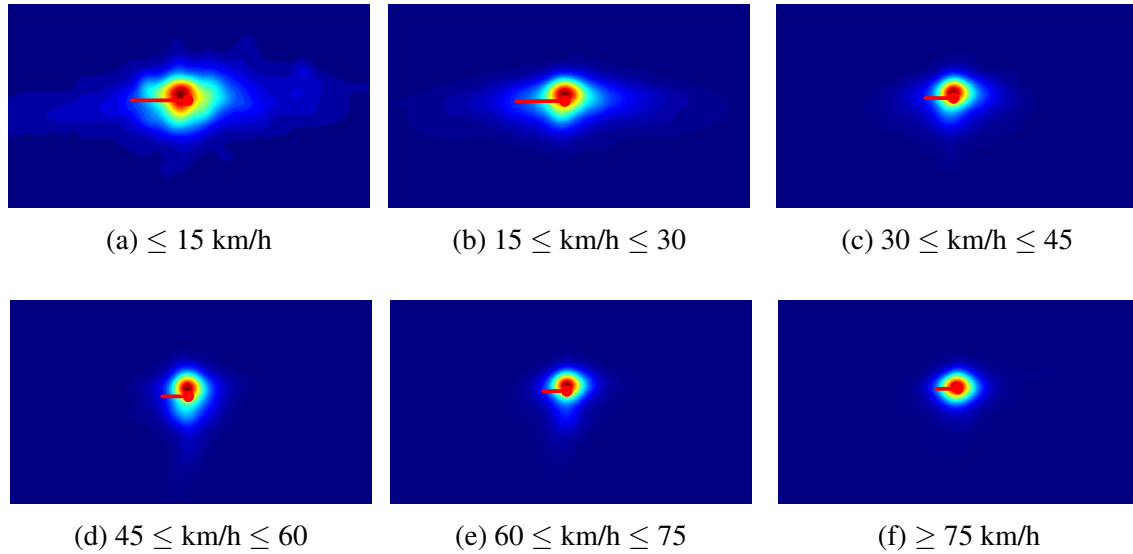


Figure 3-4: Average saliency when traveling at speeds within 6 different intervals. The mean is shown as a red circle, and the standard deviation is represented by the red line. From the figure, we can see that as speed increases, the saliency "ring" converges much more tightly around the vanishing point in the road. Note that all averages are scaled to be between $[0, 1]$.

From Fig. 3-4, we can see that the gaze ring tightens considerably around the vanishing point as speed increases. This phenomenon is corroborated by the statistics in Tbl. 3.1 -

Interval	x-mean	y-mean	std	#
≤ 15 km/h	0.47	0.50	0.175	72,632
$15 \leq$ km/h ≤ 30	0.48	0.49	0.149	69,681
$30 \leq$ km/h ≤ 45	0.46	0.49	0.108	106,482
$45 \leq$ km/h ≤ 60	0.47	0.49	0.100	104,262
$60 \leq$ km/h ≤ 75	0.45	0.49	0.096	90,665
≥ 75 km/h	0.45	0.50	0.081	111,278

Table 3.1: Mean location and standard deviations for all saliency maps within each of the speed intervals. All values are calculated from pixels normalized between the range $[0, 1]$. The means are fairly consistent between intervals, but the standard deviation drops sharply as speed increases.

as speed increases, standard deviation sharply decreases. The difference in standard deviations is especially pronounced between average saliency for speeds less than 30 km/h and that higher than 30 km/h, dropping from ≈ 0.149 to ≈ 0.108 . This phenomenon is consistent with Palazzi et al.’s results in [36]; they too find that as speed increases, the driver’s gaze deviates less and less from the vanishing point in the road.

To prove to ourselves that these differences in standard deviation between speed intervals are not merely consequential, we conduct F-tests – a statistical test to check for the equality of variances – between every pair of intervals. We plot the results of these tests in Fig. 3-5; a green square indicates significance at $\alpha = 0.05$ for that pair of intervals, yellow indicates significance at $\alpha = 0.1$, and red means no significance. As we conjectured above, the variances between every pair of intervals (I_i, I_j) , where I_i is an interval less than 30 km/h and I_j is an interval greater than 30 km/h, is statistically significantly different. We also see that variances for the interval ≥ 75 km/h is significantly different from those ≤ 60 km/h.

The fact that the standard deviation falls so sharply after 30 km/h is actually not so surprising. We conjecture that this phenomenon occurs for two main reasons. First, driving at speeds less than 30 km/h usually implies that one is driving locally in downtown areas. Downtown driving means many more hazards to keep track of (pedestrians, bikes, vehicles on intersecting roads), whereas during highway or countryside driving, there are very few hazards that can’t be caught by simply looking towards the vanishing point in the road and using peripheral vision to catch the rest (discussed more in depth in Section 3.2). Second, it

	[0, 15)	[15, 30)	[30, 45)	[45, 60)	[60, 75)	>75
[0, 15)	Red	Red	Green	Green	Green	Green
[15, 30)	Red	Red	Green	Green	Green	Green
[30, 45)	Green	Green	Red	Red	Red	Green
[45, 60)	Green	Green	Red	Red	Red	Yellow
[60, 75)	Green	Green	Red	Red	Red	Red
>75	Green	Green	Green	Yellow	Red	Red

Figure 3-5: Pairwise F-tests between variances for pairs of speed intervals. Green implies significance at $\alpha = 0.05$, yellow for significant at $\alpha = 0.10$, and red to mean no significance. Variances between intervals $\leq 30\text{km/h}$ and intervals $\geq 30\text{km/h}$ are all statistically different.

is more dangerous to look away from the road when driving fast. The faster you're driving, the more distance you've covered if you even look away for fraction of a second, and thus the more hazards you could've potentially missed on the road.

3.4 Gaze and Driving Maneuvers

We expect that gaze patterns will change significantly depending on what the driver is currently doing. For example, we expect that a driver will look left when making a left turn, or look right when driving down a right curve in the road.

In order to evaluate whether gaze is actually impacted by driving maneuvers, we must first segment the the drive into portions corresponding to different driving maneuvers. We do so programatically by first segmenting the video into overlapping 10 second intervals in increments of two seconds, illustrated in Fig. 3-6. As an example, if t to $t + 10$ is a segment, then so is $t + 2$ to $t + 12$, $t + 4$ to $t + 14$, and so on. An (x, y) displacement feature is calculated for each segment, shown in Fig. 3-7a. It's important to note that displacement for each segment is calculated relative to the orientation and position of the car at the beginning of the segment time window. Moving forward is $+y$ displacement, left

is $-x$, and right is $+x$; displacements are scaled by the maximum displacement among all segments in both x and y directions.

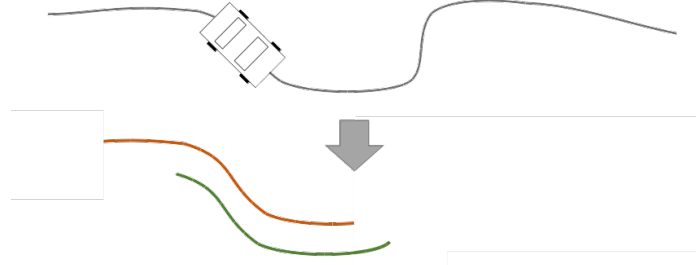


Figure 3-6: Illustration of the trajectory segmentation method. Trajectories are segmented into 10 second intervals by increments of 2 seconds – $[0, 10]$, $[2, 12]$, and so on.

These features are then used for traditional k -means clustering, and the resulting cluster space is visualized Fig. 3-7b (we choose to use $k = 10$). From examining the cluster grid, we can split the clusters into a few semantic driving maneuvers. Clusters 7, 5, 2, and 8 (the thin clusters in the center) all have very little x -displacement with varying y -displacements, and represent driving forward with varying speeds. Clusters 1 and 10, have larger x -displacements with smaller y -displacements (as compared to clusters 4 and 6) - these mostly correspond to make left and right turns at intersections respectively. On the flip side, clusters 4 and 6 have mid-range x -displacements but large y -displacements - the large y -displacements imply that these segments correspond to driving down curves in the road. Finally, clusters 3 and 9 have large x -displacements but varying y -displacements, which corresponds to making very wide turns. These clusters are significantly smaller than the others (see Fig. 3.2 for details).

Now, having classified segments roughly into a few driving maneuver categories, we turn to understanding how gaze changes between clusters. To do so, we again compute average saliency maps per cluster. These averages are visualized in Fig. 3-8, tiled in the same format as the cluster features grid in Fig. 3-7. Means and standard deviations per cluster are listed in Tbl. 3.2. As we can see from the two figures, the mean does not differ much in the y direction (only from ≈ 0.455 to ≈ 0.484), while the mean differs more in the x direction; cluster 10 has x -mean ≈ 0.453 , while cluster 1 has x -mean ≈ 0.538 .

Again, we conduct statistical tests to determine whether these differences between clus-

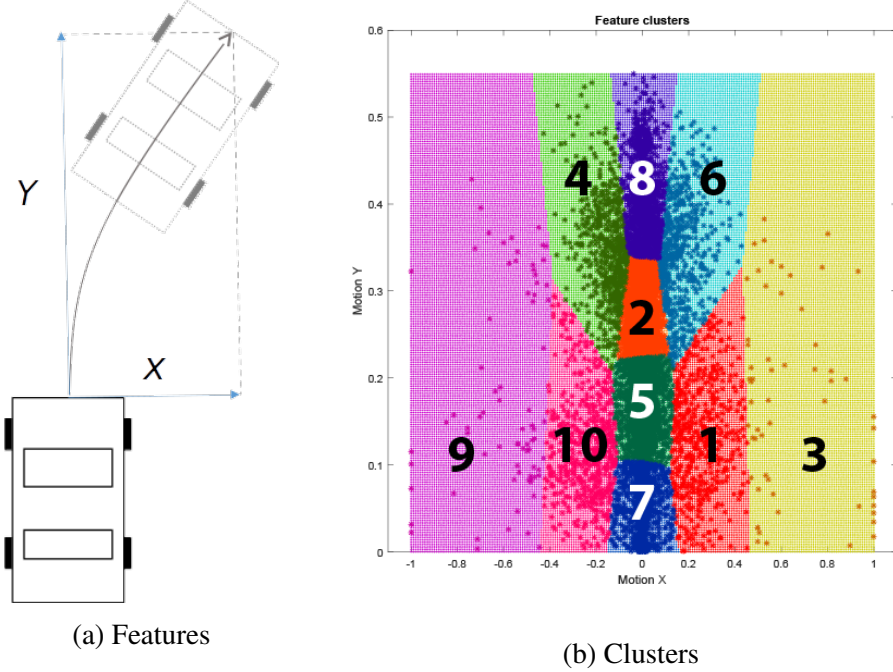


Figure 3-7: Fig. 3-7a illustrates how features are calculated for each segment. These features are then used in k -means clustering (we chose $k = 10$). Fig. 3-7b visualizes the resulting cluster space.

ter saliency means are statistically significant. This time, we perform standard one-sided T-tests to determine whether cluster means are significantly different from each other and in which direction. We tested both x -means and y -means; we discovered no significant difference between any of the y -means, but did find differences between x -means. The results of pairwise t-tests between the x -means are visualized in Fig. 3-9. From this plot we can see that clusters 1, 3, and 10 have x -means that are statistically significantly different from most other clusters. For example, the plot shows that cluster 1 (which has large $+x$ displacement) is skewed significantly more right than 7/9 of the remaining clusters, with the exception of the other clusters with $+x$ displacement (clusters 3 and 6). From these observations we can say that clusters 1 and 3 skew towards the right while cluster 10 skews towards the left.

As discussed above, clusters 1, 3, and 10 (also 9) correspond to segments in which the driver is turning. The fact that average saliency skews significantly left (as compared to the other clusters) for cluster 10 and significantly right for cluster 1 is not surprising at all

Cluster	x-mean	y-mean	std	# frames
1	0.538	0.484	0.215	65,659
2	0.488	0.471	0.142	191,878
3	0.535	0.465	0.200	12,116
4	0.475	0.457	0.147	68,844
5	0.491	0.476	0.178	219,725
6	0.514	0.455	0.138	66,677
7	0.490	0.479	0.216	116,903
8	0.496	0.458	0.141	96,292
9	0.477	0.470	0.213	13,161
10	0.453	0.484	0.208	67,950

Table 3.2: Mean location and standard deviations per cluster. All values are calculated from pixels normalized between the range $[0, 1]$. y -means do not change significantly, but x -means vary widely over the different clusters.

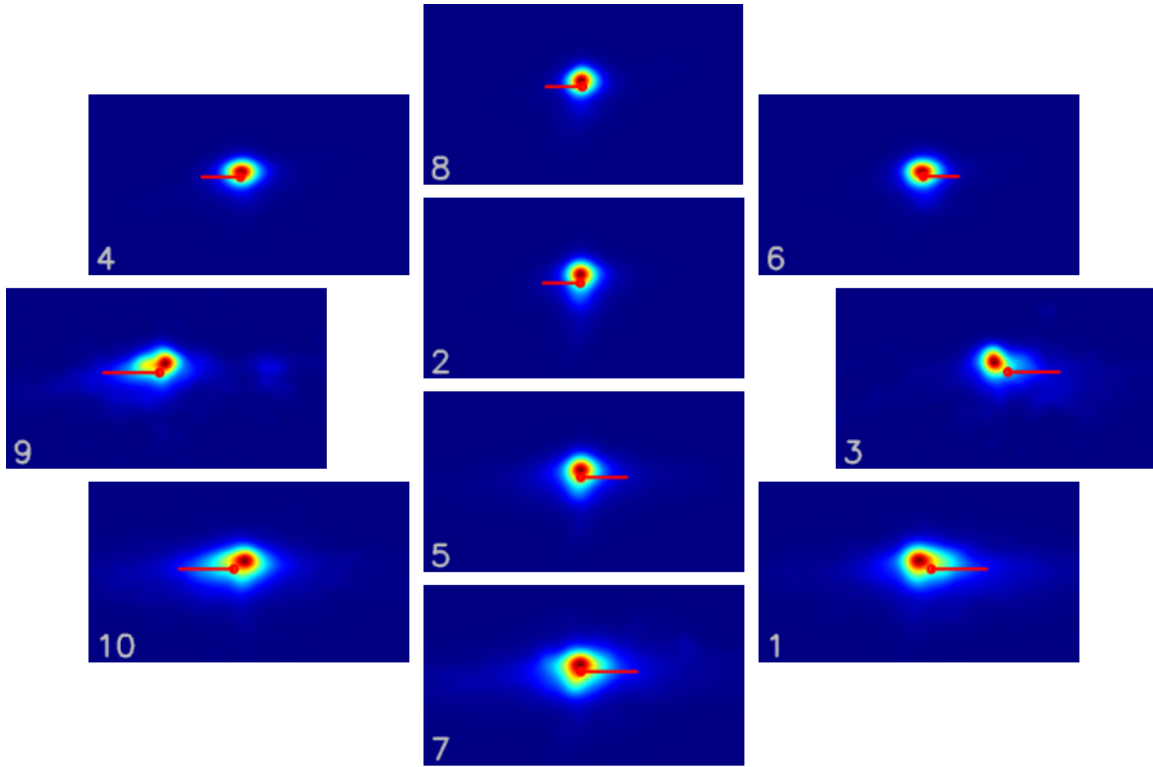


Figure 3-8: Visualization of average saliency per cluster. We note that y -mean does not change much between clusters, but x -mean shifts left for clusters with larger $-x$ displacements (4, 9, and 10) and shifts right for clusters with larger $+x$ displacements (1, 3, and 6).

	1	2	3	4	5	6	7	8	9	10
1		Green	Red	Green	Green	Red	Green	Green	Green	Green
2	Green		Green	Red	Red	Red	Red	Red	Red	Yellow
3	Red	Green		Green	Green	Red	Yellow	Yellow	Green	Green
4	Green	Red	Green		Red	Green	Red	Red	Red	Red
5	Green	Red	Green	Red		Red	Red	Red	Red	Yellow
6	Red	Red	Red	Green	Red		Red	Red	Yellow	Green
7	Green	Red	Yellow	Red	Red	Red		Red	Red	Red
8	Green	Red	Yellow	Red	Red	Red	Red		Red	Green
9	Green	Red	Green	Red	Red	Yellow	Red	Red		Red
10	Green	Yellow	Green	Red	Yellow	Green	Red	Green	Red	

Figure 3-9: Pairwise T-tests between cluster x -means. Green indicates statistical significance at $\alpha = 0.05$, yellow indicates statistical significance at $\alpha = 0.10$, and red means no significance. Clusters 1, 3, and 10 have statistically significant differences in x -means from many other clusters.

- when making a turn at an intersection, gaze should sharply change in the direction of the turn. More interesting is that while saliency does shift somewhat for clusters 4 and 6 (following a curve in the road), it does not do so considerably. Upon further consideration, this trend also follows logically because curves in the road are much less sharp than a turn at an intersection, and thus gaze does not need to change as dramatically. We can conclude that gaze patterns *do* change significantly with driving maneuver, especially when making turns at intersections or driving down a sharp curve in the road.

3.5 Gaze and Objects

In this section, we will explore what objects are actually pertinent to drivers on the road. We wish to understand both *what* objects drivers look at and *which* objects are the most important to look at.

In order to recognize driving-pertinent objects, we will segment each frame of video

from the world camera with SegNet [2]. Segnet is a deep neural model with state-of-the-art performance on the CamVid 11 [7] road class segmentation task, and competitive performance on other segmentation tasks. The model segments (i.e. labels each pixel in) a scene into the following 12 categories: sky, building, pole, road marking, road, pavement, tree, sign, fence, vehicle, pedestrian, and bike.

Using a segmentation model means that we don't need manually segment all 555,000 frames (a prohibitively time-intensive task), but it also means that the segmentation is not always accurate. Fig. 3-10 visualizes the performance of SegNet on the DrEyeVe dataset; Fig. 3-10b shows examples of environments in which SegNet works well and Fig. 3-10c shows examples of environments in which it works poorly. As we can see from Fig. 3-10b, Segnet is especially successful with images taken when it's sunny, on the highway, and during the day. On the flip side, SegNet performs poorly when it's very rainy (water droplets significantly impact predictions) or during the night (especially when there is glare from streetlights). SegNet also occasionally misclassifies certain countryside roads as buildings. With this in mind, during our analysis, we present results both when using the whole dataset (74 videos), and also only using videos not taken at night, while it's raining, or on manually selected countryside roads with poor segmentations (32 videos). We refer to this second, smaller dataset as the "filtered" portion from now on.

To understand what drivers look at while driving, we define two metrics that provide two different measurements of how often drivers look at different categories of objects. To do so, we must first define what it means for a driver to have looked at (i.e. fixated on) something. To transform continuous saliency maps (S) to binary fixation maps (F), we threshold S using a custom threshold of $\text{thr} = 0.2$. Traditionally, other saliency-related works have used $\text{thr} = 2 \times \text{avg}_{\text{saliency}}$ [29], but because most areas in DrEyeVe saliency maps are empty, $2 \times \text{avg}_{\text{saliency}}$ is a tiny number. We find that visually, using $\text{thr} = 0.2$ yields much better fixation maps.

The first of these metrics we denote p_{fix} , which calculates the densities of different semantic categories within the fixated area of a fixation map. More specifically, we define the proportion of fixation for a category c in a binary fixation map F with a corresponding

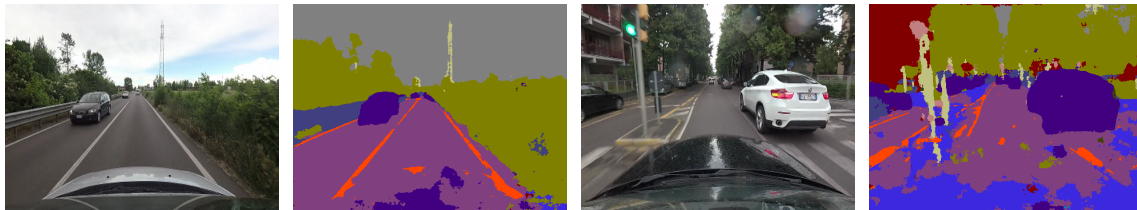


(a) SegNet colors legend



Morning, Sunny, Downtown

Morning, Rainy, Highway



Morning, Cloudy, Countryside

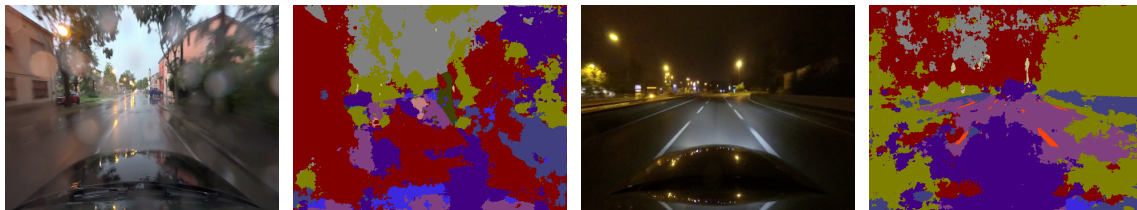
Evening, Cloudy, Downtown

(b) Examples of successful segmentation



Night, Sunny, Downtown

Morning, Cloudy, Countryside



Evening, Rainy, Downtown

Night, Sunny, Highway

(c) Examples of unsuccessful segmentation

Figure 3-10: Selected examples of running SegNet on frames from the world camera. Fig. 3-10a contains a legend mapping colors to object categories. Fig. 3-10b shows examples of environments in which SegNet works well while Fig. 3-10c shows examples of environments in which SegNet works poorly. SegNet tends to not work well on images taken at night, when it's raining, and of certain countryside roads.

segmentation map SG as:

$$p_{fix}(F, SG, c) = \frac{\sum_i [SG(i) == c] \cdot F(i)}{\sum_i F(i)}$$

where i ranges over all the pixels on the map and brackets signify the indicator function. As an example, if the driver is looking at an area encompassing 100 pixels in a scene, and 50 of those pixels are classified as 'road' by SegNet, then the driver is said to be 50% focused on the road. This metric is also used by Palazzi et al. in [36] to study what drivers focus on.

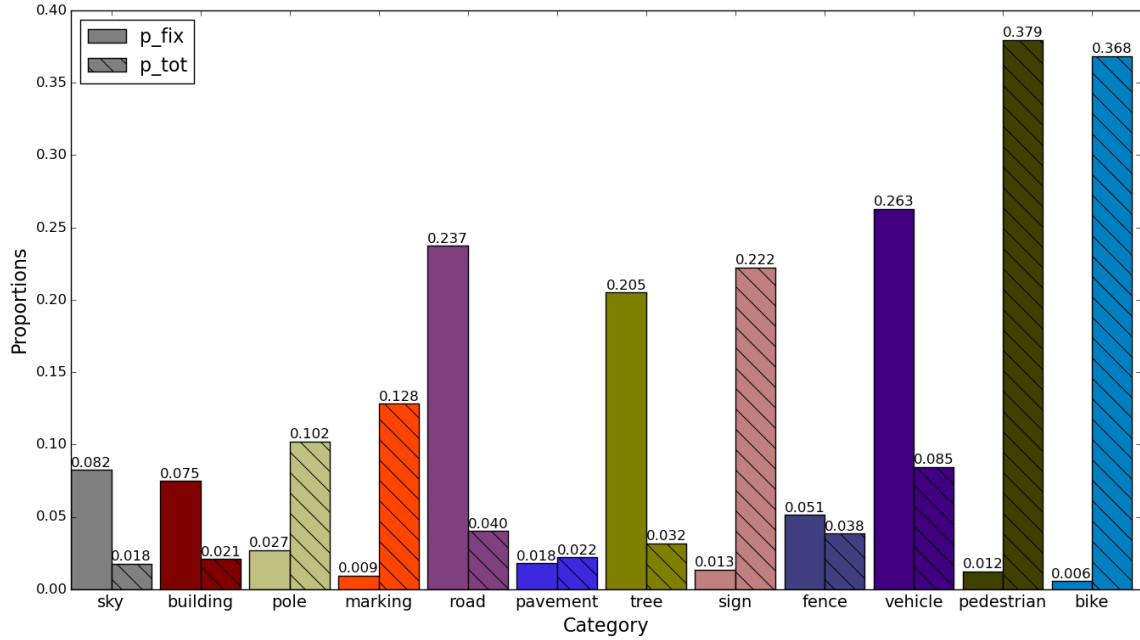
The second proposed metric we denote p_{tot} , which calculates the proportion of all occurrences of a class that the driver is actually fixated on, across all classes. This metric will give us a better sense of which classes of objects drivers are likely to find *important*. Again, for a category c in a binary fixation map F with a corresponding segmentation map SG , we define p_{tot} as:

$$p_{tot}(F, SG, c) = \frac{\sum_i [SG(i) == c] \cdot F(i)}{\sum_i [SG(i) == c]}$$

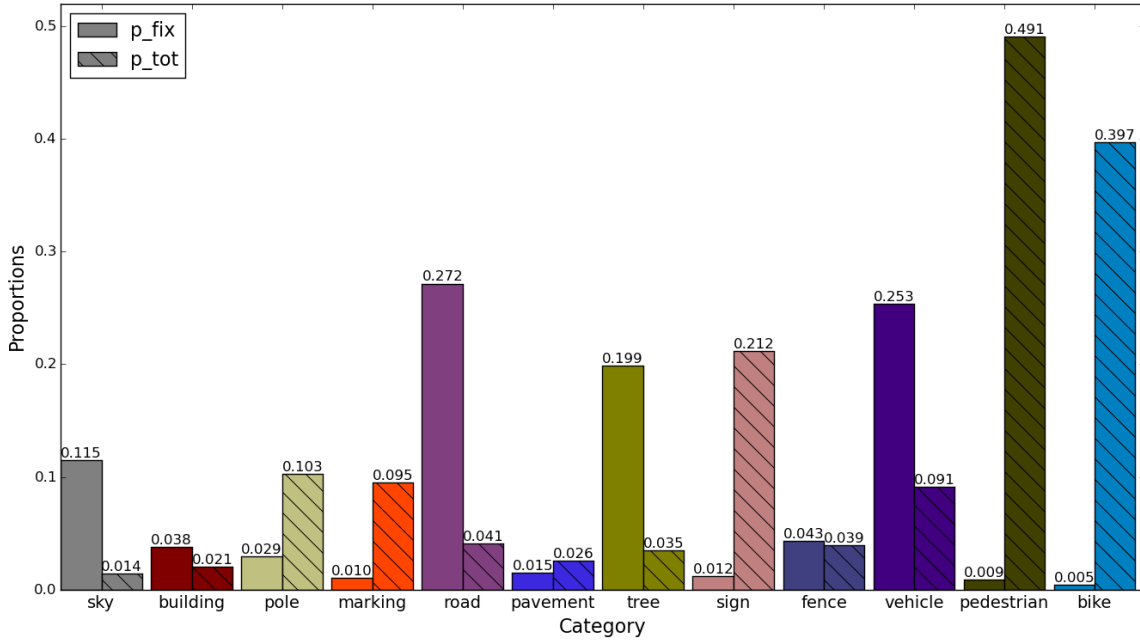
where i ranges across all the pixels and brackets are the indicator function. As an example, if there are 50 pixels in the scene classified as 'pedestrian', and the driver has fixated on 40 of them, then the driver is said to have looked at 80% of all the pedestrians in the scene. Palazzi et al. do not study this metric.

We compute p_{fix} and p_{tot} across all the saliency maps in the dataset, and report averages for each of the semantic categories. We compute two separate averages, one using the whole dataset, and the other only using the filtered dataset. Fig. 3-11 shows a comparison between average p_{fix} and average p_{tot} across all the semantic classes. The solid bars represent p_{fix} while the hatched bars represent p_{tot} .

We first discuss p_{fix} (the solid bars). As seen in Fig. 3-11, both the bar chart for the whole dataset and the bar chart for the filtered dataset are consistent in suggesting that we spend about 70% of our time focused on other vehicles, the road, trees, and the sky. The high rates of vehicles and the road match our expectations, as those are critical classes of objects to be cognizant of while on the road. However, the high incidence of trees and sky



(a) Whole dataset



(b) "Filtered" dataset

Figure 3-11: Average proportions that drivers are focused on various semantic categories under two different proportion functions, p_{fix} and p_{tot} (see Section 3.5 for their definitions); solid bars represent p_{fix} values and cross-hatched bars represent p_{tot} values. Fig. 3-11a contains averages across the whole dataset, while Fig. 3-11b contains averages across the filtered dataset, the portion with qualitatively better segmentations. The p_{fix} bars (solid) suggest that drivers spend most of their time focusing on other vehicles, the road, trees, and the sky. The p_{tot} (hatched) bars suggest that pedestrians, bikes, and road signs are the most significant to drivers.

is somewhat strange – we hypothesize that when looking towards the vanishing point in the road, the area of fixation contains the sky and the trees in the periphery. These results are consistent with what Palazzi et al. discovered in [36], even though they use a different segmentation method from us – they too find that drivers place real attentional focus on roads and vehicles.

While this data is useful for showing what drivers tend to pay attention on the road, it does not tell us which objects are actually the most important. For example, average p_{fix} in Fig. 3-11 states that drivers only spend about 0.1% of their time focused on pedestrians, suggesting that pedestrians are not very significant to drivers. In order to get a better estimate of which classes are the most relevant instead of just the most common, we turn to an analysis of p_{tot} .

The results under this new metric are shown in Fig. 3-11 represented by the hatched bars. Again, both Fig. 3-11a and Fig. 3-11b are consistent in suggesting that pedestrians, bikes, and road signs are also especially important to drivers; drivers look at about 40% of all pedestrians and bikes, and about 20% of all road signs. The rate for pedestrians is relatively high - we would have expected most pedestrians (such as the ones on sidewalks or far away) to not be salient, but this evidence suggests that we tend to at least glance at them anyway. In addition, under this new metric, we note that the results suggest vehicles and roads are markedly less significant than pedestrians, bikes, and signs. As drivers, we know this can't be true; we must combine the results from both charts to get a fuller picture.

Combining the results under these two different metrics, we conclude that vehicles, roads, pedestrians, bikes, and signs are the most important classes of objects to note while driving. We spend most of our time looking at other vehicles and the road, and we pay attention to a significant portion of all pedestrians, bikes, and signs.

3.6 Conclusions

Some key trends that we discovered through analyzing the saliency maps in DrEyeVe are as follows:

- Gaze is very center-biased, tending towards the vanishing point in the road (section

3.2).

- Gaze changes markedly with speed: the faster one is driving, the less gaze strays from the road (section 3.3).
- Gaze shifts significantly from the center when making turns at intersections, in the direction of the turn (section 3.4).
- The most salient objects to drivers are other vehicles, the road, pedestrians, bikes, and signs (section 3.5).

Many of these trends are conventional wisdoms for drivers already know – our analysis just quantifiably assesses and verifies them using real world driving data. Explicitly identifying these patterns also has a few immediate ramifications. First, we expect any model for driving saliency (no matter how the model was built) to also exhibit these traits. Second, we expect an ADAS to incorporate these trends into its algorithms for when to alert a driver of a hazard. As an example, many ADASes already raise warnings if they detect that the driver has spent too much time looking away from the center of the road. From the relationship we discovered between speed and gaze, ADASes should adjust the time threshold at which a warning is raised depending on the speed of the car; looking away from the road for an extended period of time is much more dangerous when traveling fast.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 4

Learning Driver Gaze

4.1 Objective

Motivated by the potential applications of a truly generalized model of driving saliency as discussed in the Introduction such as improved ADAS and more intelligent autonomous vehicles, we turn our attention to building such a model. More specifically, we wish to build a model of driving saliency that is able to predict what a human driver would be *focusing* on, given only a view of what the driver has *seen*. We define I_t as the driver’s field of view at time t , and S_t as the saliency map for drivers on scene I_t . Using these variables, we can reformulate the goal as finding some function F , where

$$\hat{S}_t = F(I_0, I_1, \dots, I_t)$$

.

4.2 Model Overview

Encouraged by the success of deep learning for saliency prediction and many other computer vision-related tasks, we too propose a deep convolutional network to tackle this problem. We wished to build a network that would take into account both static information (from the color and luminance in I_t) and temporal information (from the changes between

I_t and $I_{t-1}, \dots, I_0$). This implies that we could not simply use one of the networks that performed well on MIT300 [9] because 1) these models only consider static saliency from I_t , and 2) free viewing saliency is very different from task-driver saliency – where someone looks in any given scene will differ widely depending on if they are simply free-viewing the image or if they are seeing it out of the windshield of their car while driving. We also chose not to use one of the LSTM architectures for video saliency (as in [4]) because these models are typically quite memory intensive due to the fact that inputs are overlapping "video clips" (represented by a series of frames). For example, if clip length is 16 frames, $I_0 - I_{15}$ is used to predict $\hat{S}_{15}$, $I_1 - I_{16}$ is used to predict $\hat{S}_{16}$, and so on, which creates a large memory overhead due to repeated data. This aspect also implies that these models are generally slower to train because a smaller batch size is required to fit all the data into memory.

With these particular restrictions in mind, we propose a network that has two pathways, one for dynamic saliency and the other for color and luminance-related saliency, the results of which are combined to yield the final prediction. The static (color and luminance only) branch only takes in I_t and learns salient image features from that specific scene. The dynamic branch takes in both I_t and I_{t-1} , which gives it access to temporal information not present in the static branch. We hypothesize that this branch will learn salient motion features, but we do not actively restrict either branch to learning their respective tasks. A high level diagram of the entire model is shown in Fig. 4-1, and each pathway is discussed in more depth in section 4.3.

We choose to only use two inputs to this branch instead of going further back in time ($I_{t-2}, I_{t-3}, \dots$) to reduce the memory footprint. We believe that driving scenes are redundant enough that including a longer time sequence would not yield much performance gains. The DrEyeVe model [36] in particular has a very heavy memory requirements. As discussed in Section 2.3, each branch of their network takes in 16 frames (one with RGB, one with pre-computed optical flow, and the final with pre-computed segmentation) for a total of 48 frames. In contrast, we choose not to use any hand-crafted features. Palazzi et al.'s model also seems susceptible to failures in their optical flow or segmentation algorithms. As we saw in Fig. 3-10c, even state of the art segmentation algorithms are far from

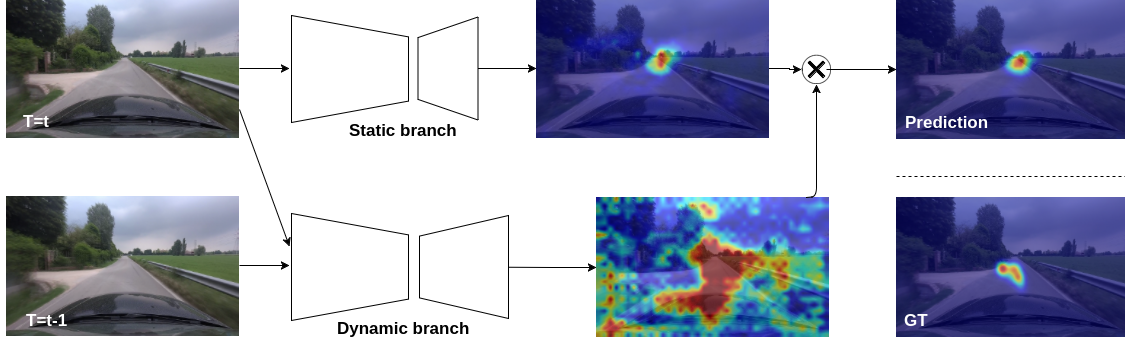


Figure 4-1: A high level diagram of DualNet. Our network is composed of two main branches, the static branch and the dynamic branch, the results of which are combined to yield the final prediction. We hypothesize that the static branch will learn salient color and luminance features while the dynamic branch will learn salient motion features. More details on specific layer architectures for each branch can be found in Tbl. 4.1.

perfect.

Given the two-pronged nature of our architecture, we name our network DualNet, or DN for short. DN’s particular architecture is also partially inspired by the model proposed by Recasens et al in [40], albeit their two-branch network is for the task of gaze following (predicting where somebody *in* an image is looking) rather than saliency prediction (predicting where somebody looking *at* an image is looking). This architecture also fits in nicely with conventional wisdom about what drivers pay attention to on the road – important static features such as the road, traffic signs, and road signs, and also nearby moving objects such as other cars, pedestrians, and bikes.

4.3 Network Architecture

In this section, we delve deeper into the actual architecture of each of the branches, as well as how they are combined to predict the final saliency map. We assume that the input frames and the ground truth saliency maps are all of size $A \times B$.

- **Static saliency:** The color and luminance pathway is passed I_t and produces a $M \times N$ saliency map, where $M = A/8$ and $N = B/8$. We test two different networks for this branch, ResNet-50 [17] and CaffeNet [43] (which is a modified version of

Layer	ResNet	CaffeNet
Conv1x	$7 \times 7, 64, \text{stride} = 2$ $3 \times 3 \text{ pool}, \text{stride} = 2$	$11 \times 11, 96, \text{stride} = 2$ $3 \times 3 \text{ pool}, \text{stride} = 2$
Conv2x	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$5 \times 5, 256$ $3 \times 3 \text{ pool}, \text{stride} = 2$
Conv3x	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$3 \times 3, 384$
Conv4x	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$3 \times 3, 384$
Conv5x	None	$3 \times 3, 256$
Conv6x	None	$3 \times 3, 512$
Deconv1x	$4 \times 4, 512, \text{stride} = 2$	None

Table 4.1: Modified layer parameters for each of the static saliency networks. Given a kernel size k and number of outputs o , a row is of the form $(k \times k, o)$, and stride is 1 unless otherwise specified. All layers are response normalized.

AlexNet [25]). We modify each of the networks to reduce the total amount of down-sampling (each network originally downsampled by 32). For ResNet-50, we remove the entire conv5x segment of the network (leaving 41 layers), and add in a simple deconvolution layer at the end with stride 2. For Caffe-Net, we reduce the stride in conv1 from 4 to 2 and also add another convolutional layer at the end. A more detailed description of modified layer parameters for each network can be found in Tbl 4.1. The output of each of these networks is a $M \times N \times K$ latent space representation of the original image. The outputted $M \times N \times K$ representation is then convolved with a learned $1 \times 1 \times K$ filter to produce the final $M \times N$ static saliency map.

- **Motion saliency:** The motion pathway is passed I_t and I_{t-1} and also produces a $M \times N$ saliency map, where $M = A/8$ and $N = B/8$. We choose the network FlowNet [13] for this portion of the network. FlowNet was originally designed designed to learn the x - y optical flow field between two frames, making it a good candidate for learning motion saliency. Images I_t and I_{t-1} are first concatenated and then passed through FlowNetS up to layer 'flow_concat3'. FlowNetS downsamples input images by a factor of 64 in the "convolutional" portion of the network, and then

upsamples by 8 in the "upconvolutional" section. In FlowNet, the 'upconvolutional' layers are just a convolution on the concatenation of an upsampled version of the previous flow prediction and the corresponding feature map from the convolutional section of the network, which results in a predicted flow map that's twice as large. Again, FlowNetS outputs a $M \times N \times K$ feature map after running I_t and I_{t-1} through the first 12 layers, which is then convolved with a learned $1 \times 1 \times K$ filter to produce the final $M \times N$ motion saliency map. Please see Fischer et al.'s paper [13] for a detailed description of layer parameters.

- **Combining pathways:** The resulting $M \times N$ saliency maps from both pathways are combined via an element-wise product. In order to generate a full-sized prediction $\hat{S}_t$, we upsample the product p by a factor of 8 using bilinear upsampling.

4.4 Training

We train our networks on gaze maps and world camera video (described in detail in section 3.1) provided in the DrEyeVe dataset. We use frames from video taken by the roof-mounted Garmin camera as inputs I_t and the scanpath gaze maps as ground truth for S_t . A randomly selected 80% of the videos for training and the remaining 20% for testing. We partition by videos instead of frames to ensure that the model has not seen any of the test scenes during training.

We train our networks both on the unmodified dataset and a dataset augmented with random crops. The unmodified dataset is generated by only downsampling the original dataset to size (320, 192). The random crops in the augmented dataset are generated by first downsampling the original inputs to (480, 288) and then taking a random crop of (320, 192). This implies that each crop contains less than 50% of the original frame. We incorporate random crops into our dataset to make sure ground truth saliency maps are sufficiently different from the mean, thereby preventing overfitting to the mean.

Our network was implemented and trained in Caffe [21]. Before training, the static and dynamic branches are initialized with weights from their corresponding networks; e.g. the

dynamic branch is initialized with FlowNet, and the static branch is initialized with either ResNet-50 or CaffeNet depending on the specific model being trained. The network was then trained end-to-end with a standard cross entropy loss function. Training happens for 80,000 iterations using SGD with a base learning rate of 5×10^{-4} .

Again, an important caveat to note is that because the two pathways are jointly trained, there is no guarantee that either branch will learn its respective task. In Section 5.4.3 we actually explore what each branch has learned.

Chapter 5

Experiments

In this section, we evaluate the ability of the model we proposed in Chapter 4 to accurately predict salient regions for drivers. We compare our model against a few baselines, selected state-of-the-art image saliency models, and the DrEyeVe model [36], and then analyze the results, both qualitatively and quantitatively. Finally, we dive into the internal layers to gain a deeper intuition of what the network has learned.

5.1 Evaluation Strategy

We compare results from our models against ground truth saliency maps on four commonly used saliency metrics: AUC, Pearson’s correlation coefficient (CC), mean average error (MAE), and Kullback-Leibler divergence (KL). We selected these four in particular given their favorable properties described in [10].

Given a predicted saliency map P and a ground truth saliency map Q , and their corresponding probability distributions $\mathbf{P}$ and $\mathbf{Q}$, the calculations for CC and KL are reproduced below (for reference later).

$$CC(\mathbf{P}, \mathbf{Q}) = \frac{\sigma(\mathbf{P}, \mathbf{Q})}{\sigma(\mathbf{P}) \times \sigma(\mathbf{Q})} \quad (5.1)$$

$$KL(\mathbf{P}, \mathbf{Q}) = \sum_i \mathbf{Q}_i \log \left(\epsilon + \frac{\mathbf{Q}_i}{\epsilon + \mathbf{P}_i} \right)^1 \quad (5.2)$$

MAE is calculated as in [29], and AUC is calculated as in [23]. Note that in order to convert continuous saliency maps into binary fixation maps to calculate AUC, we again use the same strategy of using the custom threshold of 0.2, as discussed in Section 3.6.

To calculate these metrics across the entire testing suite, we simply average results across all saliency maps in the set.

5.2 Models

We will compile results for our models, as well as a few selected baselines. All these models are described below in more detail:

- **Centered Gaussian:** We find the mean fixation location and corresponding standard deviation of all saliency maps in the training set and construct a centered gaussian accordingly.
- **Mean:** We calculate the mean of all the saliency maps in the training set. This baseline is less commonly used in free-viewing saliency research, but may be more applicable in this case as driving is a very center biased task (discussed in Section 3.2).
- **Free-viewing saliency models:** We pick two models that perform favorably on the MIT300 [9] benchmark: SALICON [19] and SalNet [37]. The SALICON model has two VGG-16 branches, one which takes in a "coarse" image map (a downsampled version of the original image), and the other takes in a "fine" image map (the original image). The SalNet model is much smaller, with only nine total convolutional layers. SALICON is currently ranked 2nd while SalNet is ranked 16th on the MIT300 challenge. Both models are fine-tuned on the DrEyeVe dataset, SalNet using their implementation on GitHub, and SALICON using an open source reimplementation [45] (Huang et al. [19] do not release their original code). We must retrain the models, because free viewing saliency is very different from task-driven saliency (further

¹We use $\epsilon = 1.18 \times 10^{-7}$, the same value used in [36] for consistency.

discussed in sections 3.6 and 4.2); an unchanged free viewing saliency model would perform very poorly on this dataset.

- **Ours:** We present three models: DN-ResNet, DN-CaffeNet, and DN-ResNet-crops. To recap on Chapter 4 (where these models are discussed in more depth), DN-CaffeNet has a modified version of CaffeNet [43] for learning static saliency, DN-ResNet has a modified version of ResNet-50 [17] as the static branch, and finally DN-ResNet-crops is DN-ResNet trained on the dataset augmented with random crops.

5.3 Results

Model	upsample	AUC↑	CC↑	KL↓	MAE↓
mean	1	0.9600	0.5205	1.6594	0.00938
center gaussian	1	0.9482	0.2904	2.6455	0.1734
salicon	16	0.9561	0.4947	1.6579	0.01289
salnet	1	0.8946	0.3043	2.6234	0.0127
DN-CaffeNet	8	0.9535	0.5152	1.6831	0.00933
DN-ResNet	8	0.9550	0.5211	1.6134	0.00852
DN-ResNet-crops	8	0.9269	0.5175	1.4876	0.0107

(a) Whole dataset

Model	upsample	AUC↑	CC↑	KL↓	MAE↓
mean	1	0.8920	0.1339	2.9360	0.0523
center gaussian	1	0.8392	0.1964	2.9787	0.0429
salicon	1	0.8974	0.2990	2.4353	0.0225
salnet	1	0.8539	0.2347	2.8361	0.0376
DN-CaffeNet	8	0.9015	0.3088	2.3182	0.0198
DN-ResNet	8	0.9040	0.2980	2.4022	0.0173
DN-ResNet-crops	8	0.8603	0.2892	2.2491	0.0209

(b) Acting dataset

Table 5.1: We evaluate our models against the baselines on both the whole testing test as well as the acting portion of that set (a manually annotated set by [36] that is sufficiently different from the mean). On AUC and CC, higher numbers are better, while on KL and MAE, lower numbers are better, as indicated by the direction of the arrows. On both datasets, our models outperform the baselines, and our models significantly outperform the mean on the acting dataset. See Section 5.3 for a more detailed discussion of the results.

The results for models run on the entire testing dataset are presented in Tbl. 5.1a. On this dataset, our models significantly outperform the baselines on KL and MAE, with a best KL of 1.4876 as compared to the baseline mean's KL of 1.6574, and a best MAE of 0.00852 as compared to the baseline mean's MAE of 0.00938. However, on AUC and CC, our models are only marginally different from simply predicting the mean. Our hypothesis for why this behavior occurs is rooted in the fact that we train using cross entropy loss, which is closely related to KL. KL loss heavily penalizes areas where ground truth is non-zero but prediction is zero, as the term $\frac{Q_i}{\epsilon + P_i}$ can get arbitrarily large. This behavior leads to well-localized, but very wide attention maps, which drives down the other metrics. These wide attention maps can be seen in the visualization of some example outputs in Fig. 5-1 in columns 3 and 4, which are results from DN-ResNet and DN-ResNet-crops respectively. CC in particular is negatively impacted by wide attention maps; CC is only high when predictions are ground truth are similar in magnitude.

From the results in Tbl. 5.1a, we can also glean some interesting comparisons between our three proposed models. Training model DN-ResNet with random crops (DN-ResNet-crops) significantly decreases the KL divergence; this trend is expected, as training with random crops reduces overfitting to the mean. However, even though KL divergence has decreased, CC remains roughly the same. Looking at the results visually in Fig. 5-1 gives a clue as to why this is; in situations in which the salient region is not obvious (in other words, when saliency differs significantly from the vanishing point in the road), DN-ResNet-crops tends to predict wider saliency maps. This behavior is especially pronounced in the last two rows of Fig. 5-1. Again, wider attention maps prevents KL from heavily penalizing pixels where ground truth is non-zero but prediction is zero, but also causes CC to drop. We also note that while DN-ResNet has many more convolutional layers in the static branch than DN-CaffeNet (see Chapter 4 for more details on model architecture), only KL improves significantly, while CC remains largely the same.

While our models do not significantly outperform the mean on the whole dataset (which is very mean-biased), we do significantly outperform the mean on the "acting" dataset. The acting dataset is a subset of the original DrEyeVe dataset, which is generated by Palazzi et al. in [36]. It is a section of the dataset where saliency is sufficiently different from the

mean, and meant to be used as a proxy for situations in which saliency is harder to predict. The acting dataset is only about 13% the size of the original dataset. To generate this dataset (discussed in further detail in section 3 of [36]), Palazzi et al. first filtered the dataset to only include maps where CC with the mean is < 0.3 , and then further filtered that set to only include frames where the driver is paying attention (manual annotation by Amazon Mechanical Turk workers). On the acting dataset, our models have a high of 0.3088 on CC (where higher is better) while the mean only scored a 0.1339, more than twice as low. On KL (where lower is better), our models scored a low of 2.2491 on KL, while the mean scored a 2.9360, again showing a significant decrease. Looking at the performance between our models, we see that DN-Resnet-crops is again the best model according to KL, but now DN-CaffeNet is the best according to CC.

5.3.1 Comparison with DrEyeVe Model

The DrEyeVe model reports performance of 0.559 on CC and 1.398 on KL for the whole dataset (Table 3 in [36]). This is an increase over our models of ≈ 0.04 on CC and ≈ 0.08 on KL, which is certainly a moderate improvement. However, it's hard to make a direct comparison between the two sets of numbers because we did not get the opportunity to retrain their model on our training data, and subsequently test it on our testing data. We do not believe Palazzi et. al. have released a pretrained model.

An interesting note about these metrics is that the magnitude of metrics for models trained on the DrEyeVe dataset compared to those trained on other saliency datasets is significantly lower. For example, we see that SALICON only achieves a CC of 0.4947 and KL of 1.6579 on DrEyeVe, but a CC of 0.85 and KL of 0.54 on MIT300. We hypothesize that this is due to the fact that ground truth saliency maps are generally very tightly localized (as can be seen in column 2 in Fig. 5-1), because they represent only what *one* driver is looking at in a particular situation (discussed in more depth in Section 2.1). Another driver, even in an identical situation, may be focused somewhere else. In traditional saliency datasets, fixations from many viewers are compiled to produce one overall saliency map for that image. However, in DrEyeVe, drivers in similar situations will produce multiple

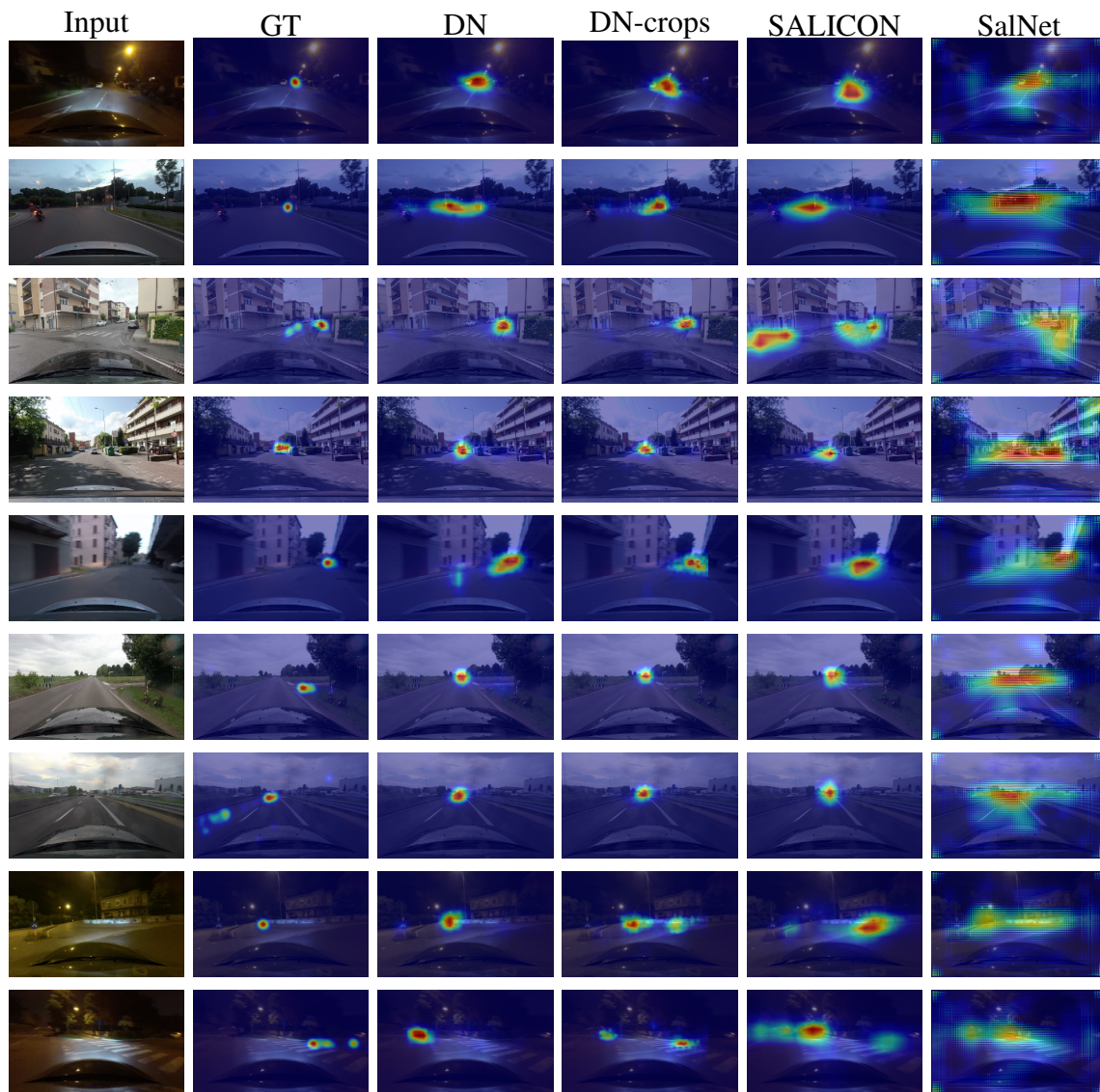


Figure 5-1: Visualization of the results of our models and the baselines. Column 1 is the input image, Column 2 is the ground truth saliency map, Columns 3 and 4 are results from our models, and Columns 5 and 6 are results from the baselines. We can see that our models generally produce tighter, more well localized maps than the baselines.

ground truth saliency maps, each very tightly localized at different locations. This will force any model to learn wider saliency maps, which will not perform well on any of the ground truth maps.

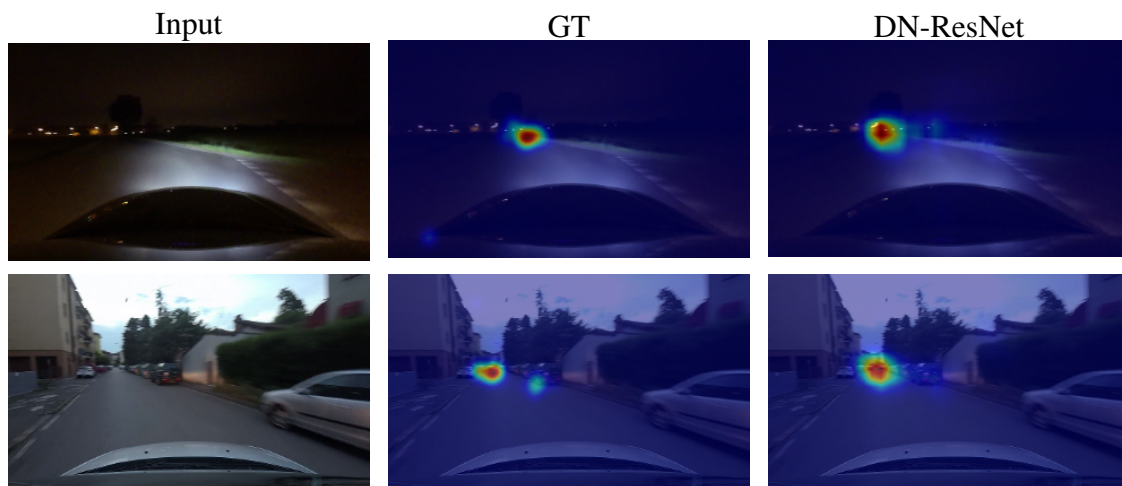
5.4 Performance Breakdown

5.4.1 Qualitative

We qualitatively examine what our models have and haven't learned about salient regions to drivers. We selectively analyze results from the DN-ResNet model. From visual examination, we see that our model has learned to identify the vanishing point in the road, splits in the road, and other vehicles pertinent to driving very well. Fig. 5-2 visualizes some examples of relevant instances. The fact that the model follows the vanishing point of the road so well contributes significantly to why our model outperforms the mean on the acting dataset. As we noted in Section 3.2 and 3.3, drivers tend to fixation on the vanishing point of the road, especially at higher speeds. A sizable portion of the acting dataset are frames from driving down curves in the road (when saliency deviates significantly from the mean), and our model is very good at identifying the vanishing point in those cases. We also note that our models' success at learning roads and vehicles is in line with the observation made in Section 3.5 that roads and vehicles are the most commonly fixated classes of objects for drivers.

On the flip side, we investigate inputs on which our model performs especially poorly. We take frames where our models predict saliency maps with less than CC of 0.1 with the ground truth. Two tasks that come up frequently are predicting traffic lights and predicting the correct salient turn in advance. Fig 5-3a shows examples of instances when our model misses traffic lights. We hypothesize that this behavior happens due to two factors: 1) little of the time waiting at a traffic light is actually spent looking at the light, or 2) traffic lights do not make up a significant enough portion of the dataset.

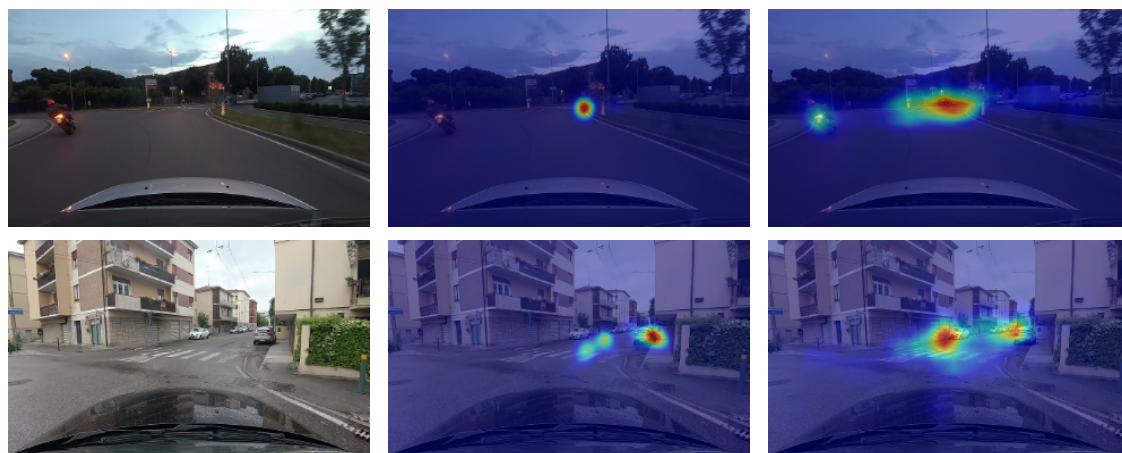
Fig. 5-3b illustrates our performance on situations in which we need to predict turns in advance. We can see again here that our model is able to distinguish between roads well,



(a) Following the road



(b) Following road splits

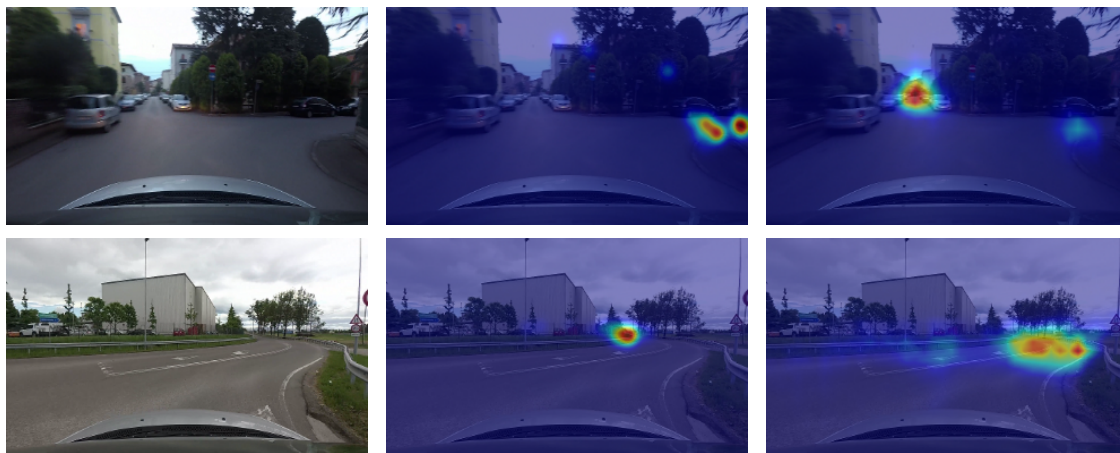


(c) Identifying pertinent vehicles

Figure 5-2: Some examples of salient regions that our model (DN-ResNet) has learned very well. These include the identifying the vanishing point in the road, splits in the road, and pertinent vehicles.



(a) Identifying traffic lights and signs



(b) Predicting turns in advance

Figure 5-3: Some tasks our model (DN-ResNet) struggles with. These include predicting turns in advance (before the car has physically begun turning) and identifying traffic lights/signs.

but chooses the direction incorrectly when predicting too far in advance. The top image especially showcases this effect; our model outputs a multimodal prediction with a peak at each road opening, but with a much higher peak at the opening straight ahead because that's the direction the car is moving currently. However, the driver was intending to make a right turn, and has already begun looking that direction, even before the car has begun turning. Without knowing the driver's intentions beforehand, predicting turns before the car has begun physically turning is near impossible. After the car has physically begun turning, predictions become much more accurate, as can be seen in rows 3 and 8 in Fig. 5-1. SALICON and Salnet in those situations have more poorly localized gaze maps as they do not take into account the temporal information.

From these observations, we note that our current models are well-suited for saliency prediction in most driving scenarios, except in the cases of city driving at intersections, when traffic lights and needing to predict turns in advance are especially pertinent. To improve the performance on these particular cases, we propose both expanding the dataset to include more examples of saliency behavior at intersections and also potentially incorporating information about drivers intended maneuvers into the model.

5.4.2 Prediction Size Study

We now turn to a study on how network prediction size impacts performance. Our current model predicts maps of size 40x24 and upsamples by 8 to get to the original size, but what happens to performance if we instead predicted smaller maps (more upsampling) or bigger maps (less upsampling)? We modify DN-CaffeNet to output predictions of size 10x6, 20x12, 40x24, and 80x48. In order to get smaller predictions, we simply change the stride size on the final convolutional layer (to avoid changing the number of convolutional layers). To get bigger predictions, we add in a correctly sized deconvolutional layer at the end.

The effects on metrics across the entire dataset can be seen in Fig. 5-4. Each set of metrics is normalized by the maximum of that set to yield a value in the range [0, 1]. We can see from the chart that 40x24 predictions outperform smaller predictions on all

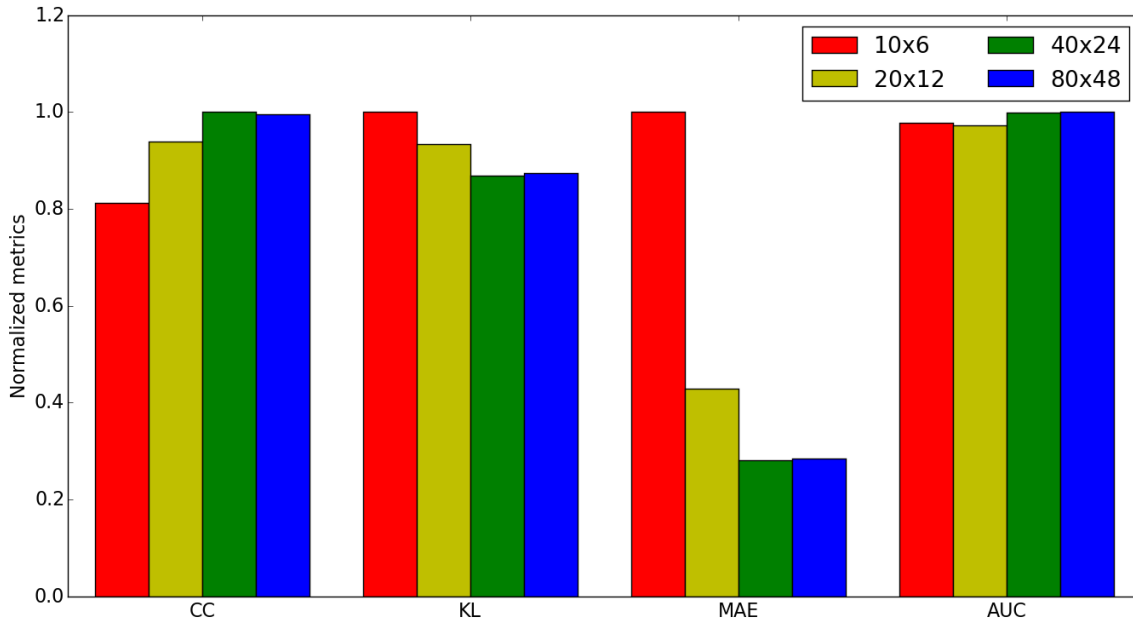


Figure 5-4: Normalized CC, KL, MAE, and AUC values for DN-CaffeNet with predictions at various scales. 40x24 predictions outperform smaller predictions on all 4 metrics, and there is no discernable performance gain from increasing prediction size past 40x24.

four metrics, and have similar performance to 80x48 predictions. One would expect larger predictions to more accurately capture the ground truth, but the reality is that there is no discernable performance gain from increasing the prediction size past 40x24.

We visualize some example outputs for predictions at various scales in Fig. 5-5. We can see from the visualization that models with smaller predictions yield much wider maps due to the larger level of upsampling, but are localized equally well as their larger prediction counterparts. This effect is especially pronounced at the 10x6 prediction level – the difference in width of the saliency maps between 10x6 predictions and 20x12 predictions is much bigger than that between 20x12 predictions and 40x24 predictions. On the flip side, when increasing the prediction size, there starts to be a lot more visual noise in the prediction. More fine-grained predictions means less smoothing when upsampling, causing a more noisy image when predictions are not already smooth.

As a result, we recommend 40x24 size predictions, which are 8 times smaller than the original input size of 320x192. Decreasing the prediction size degrades performance, while

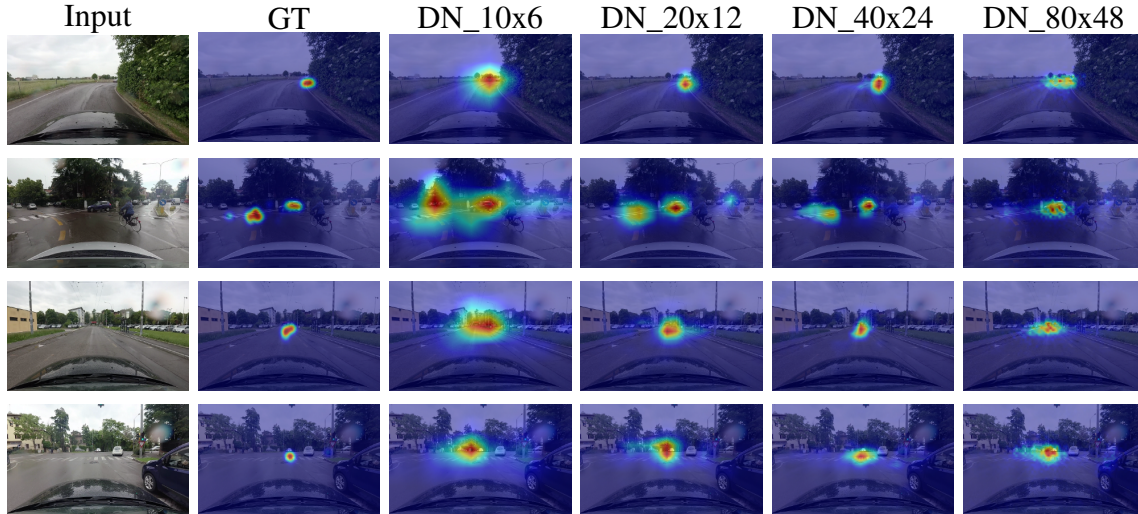


Figure 5-5: Visualization of performance of DN-CaffeNet with predictions of different sizes. The suffix of the model name is the size of the prediction. We can see that smaller predictions are usually just as well localized, but with a much wider radius due to the higher level of upsampling needed to scale the prediction back up to the original size of the input (320x192 in our case).

increasing prediction size does not buy extra performance.

5.4.3 Pathway Visualization

We wish to understand what each branch in our network has actually learned. We hypothesized back in Chapter 4 that the static branch, which only takes in frame I_t as input, would learn color and luminance based saliency, while the dynamic branch, which takes in I and I_{t-1} as inputs, would learn motion based saliency. To evaluate our hypothesis, we visualize the final $M \times N$ saliency maps generated by each branch along with the final predicted output (columns 4, 5, and 6), shown in Fig. 5-6. Column 2 contains an overlay between frames I and I_{t-1} as well as an arrow indicating the car's direction of travel, which more easily allows us to get a sense of flow between the two frames.

From the visualization, we can see that the first half of our hypothesis was more or less correct. The static branch does seem to learn all areas that may be of salience to a driver. This includes the road ahead, other vehicles, and other joining roads. However, instead of learning salient regions from motion outside of the vehicle, the dynamic branch actually seems to be contributing mostly by learning saliency from the motion of the car itself! For

example, if the car is turning left, the dynamic branch will learn that from frames I and I_{t-1} and adjust its saliency map prediction to only include road to the left. Because the static branch is not fed with any temporal information, it does not learn the motion of the vehicle and will select salient regions across the entire frame.

This phenomenon is well illustrated by Fig. 5-6. For example, in the 3rd row, we can see from the flow overlay that the car is moving approximately forward. The flow saliency reflects this by generating a wide saliency map for all road that's approximately in the forward direction. Meanwhile, the static saliency branch has learned about the road straight ahead, the road to the right, and a parked car to the right. Combining their contributions together, the flow saliency map cancels out the predictions to the right because the car is not moving in that direction, leaving only the road straight ahead.

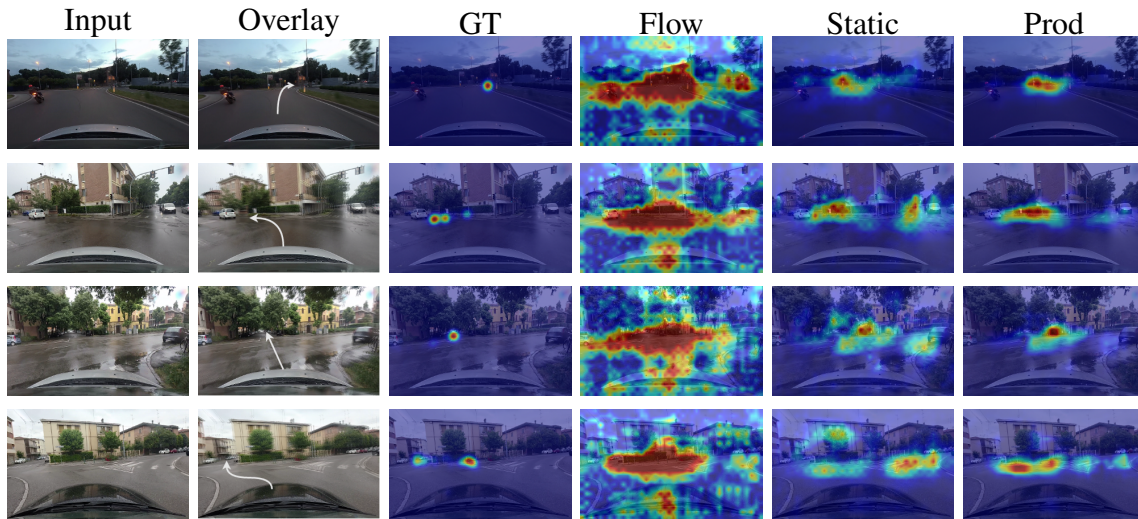


Figure 5-6: Visualization of the contributions of both the static and dynamic branches to the overall saliency prediction. The arrow on the images in Column 2 indicate the direction the car is traveling. This figure illustrates that the static branch has learned to spot salient objects to drivers (the road ahead, other vehicles, joining roads), while the dynamic branch has learned to predict a saliency map based on the direction the vehicle is travelling.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 6

Conclusion and Future Work

Analyzing and modeling driver attention is the first step in understanding how humans perform a task as complex as driving using only vision. In this thesis, we begin by quantitatively establishing some general trends regarding how speed, driving maneuver, and objects in the environment affect drive gaze. As per conventional wisdom, driving fast causes a marked decrease in gaze variation, turning at intersections results in the most significant gaze shifts, and vehicles, roads, pedestrians, bikes, and traffic signs are the most important classes of objects to drivers on the road.

We continue by presenting a generalized model for driving saliency that shows encouraging results when compared to other saliency models, especially on more difficult driving situations. We qualitatively show that our model learned roads, road splits, and other vehicles well but performs poorly on traffic lights and predicting salient turns in advance. We also show that our model has incorporated both static (by learning salient roads and vehicles) and dynamic (by learning the movement of the vehicle) information into the final prediction.

On a small scale, there is still much to do regarding our current model. For one, we need to perform an actual comparison with the DrEyeVe model so we can accurately evaluate which strategy is superior. For another, our model has no concept of temporal consistency between saliency maps – we should investigate whether incorporating an LSTM or using some form of spatio-temporally consistent loss function would improve the results. Finally, we should investigate the performance of our model on video saliency datasets such as

Hollywood2 [34]. Our model does not use any handcrafted features or inputs, so there's no reason it can't be retrained on a new dataset.

On a larger scale, actually understanding and modeling driver saliency well has the potential to have widespread consequences both for how we interact with our vehicles and for how the next generation of autonomous vehicles will be programmed. As we mentioned in the Introduction, ADAS can be significantly improved through being able to selectively warn user of hazards on the road, and also more accurately detecting driver inattention. Autonomous vehicles can use the model to learn to drive more like a human, rather than requiring hundreds of thousands of dollars worth of sensors to do the same task.

We hope that our work is just the beginning of many further studies into driver attention. The future with truly intelligent ADAS and autonomous vehicles is just too exciting to let pass us by.

Bibliography

- [1] Stefano Alletto, Andrea Palazzi, Francesco Solera, Simone Calderara, and Rita Cucchiara. Dr (eye) ve: A dataset for attention-based tasks with applications to autonomous and assisted driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 54–60, 2016.
- [2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015.
- [3] Karlene Ball, Cynthia Owsley, Michael E Sloane, Daniel L Roenker, and John R Bruni. Visual attention problems as a predictor of vehicle crashes in older drivers. *Investigative ophthalmology & visual science*, 34(11):3110–3123, 1993.
- [4] Loris Bazzani, Hugo Larochelle, and Lorenzo Torresani. Recurrent mixture density network for spatiotemporal visual attention. *arXiv preprint arXiv:1603.08199*, 2016.
- [5] Ali Borji, Ming-Ming Cheng, Huaizu Jiang, and Jia Li. Salient object detection: A benchmark. *IEEE Transactions on Image Processing*, 24(12):5706–5722, 2015.
- [6] Ali Borji and Laurent Itti. Cat2000: A large scale fixation dataset for boosting saliency research. *CVPR 2015 workshop on "Future of Datasets"*, 2015. *arXiv preprint arXiv:1505.03581*.
- [7] Gabriel J Brostow, Julien Fauqueur, and Roberto Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, 30(2):88–97, 2009.
- [8] Zoya Bylinskii, Phillip Isola, Constance Bainbridge, Antonio Torralba, and Aude Oliva. Intrinsic and extrinsic effects on image memorability. *Vision research*, 116:165–178, 2015.
- [9] Zoya Bylinskii, Tilke Judd, Ali Borji, Laurent Itti, Frédo Durand, Aude Oliva, and Antonio Torralba. Mit saliency benchmark.
- [10] Zoya Bylinskii, Tilke Judd, Aude Oliva, Antonio Torralba, and Frédo Durand. What do different evaluation metrics tell us about saliency models? *arXiv preprint arXiv:1604.03605*, 2016.

- [11] Souad Chaabouni, Jenny Benois-Pineau, Ofer Hadar, and Chokri Ben Amar. Deep learning for saliency prediction in natural video. *arXiv preprint arXiv:1604.08010*, 2016.
- [12] Marcella Cornia, Lorenzo Baraldi, Giuseppe Serra, and Rita Cucchiara. Predicting human eye fixations via an lstm-based saliency attentive model. *arXiv preprint arXiv:1611.09571*, 2016.
- [13] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015.
- [14] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.
- [15] Insurance Institute for Highway Safety. Insurance institute for highway safety - fatality facts. <http://www.iihs.org/iihs/topics/t/general-statistics/fatalityfacts/overview-of-fatality-facts>. Accessed: 2017-08-11.
- [16] Lex Fridman, Philipp Langhans, Joonbum Lee, and Bryan Reimer. Driver gaze region estimation without use of eye movement. *IEEE Intelligent Systems*, 31(3):49–56, 2016.
- [17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645. Springer, 2016.
- [18] William J Horrey, Christopher D Wickens, and Kyle P Consalus. Modeling drivers’ visual attention allocation while interacting with in-vehicle technologies. *Journal of Experimental Psychology: Applied*, 12(2):67, 2006.
- [19] Xun Huang, Chengyao Shen, Xavier Boix, and Qi Zhao. Salicon: Reducing the semantic gap in saliency prediction by adapting deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 262–270, 2015.
- [20] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 20(11):1254–1259, 1998.
- [21] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.
- [22] Ming Jiang, Shengsheng Huang, Juanyong Duan, and Qi Zhao. Salicon: Saliency in context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1072–1080, 2015.

- [23] Tilke Judd, Krista Ehinger, Frédo Durand, and Antonio Torralba. Learning to predict where humans look. In *Computer Vision, 2009 IEEE 12th international conference on*, pages 2106–2113. IEEE, 2009.
- [24] Panos Konstantopoulos, Peter Chapman, and David Crundall. Driver’s visual attention as a function of driving experience and visibility. using a driving simulator to explore drivers’s eye movements in day, night and rain driving. *Accident Analysis & Prevention*, 42(3):827–834, 2010.
- [25] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [26] Srinivas SS Kruthiventi, Kumar Ayush, and Radhakrishnan Venkatesh Babu. Deep-fix: A fully convolutional neural network for predicting human eye fixations. *IEEE Transactions on Image Processing*, 2017.
- [27] Matthias Kümmerer, Thomas SA Wallis, and Matthias Bethge. Deepgaze ii: Reading fixations from deep features trained on object recognition. *arXiv preprint arXiv:1610.01563*, 2016.
- [28] Michael F Land and David N Lee. Where we look when we steer. *Nature*, 369(6483):742–744, 1994.
- [29] Guanbin Li and Yizhou Yu. Deep contrast learning for salient object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 478–487, 2016.
- [30] Nian Liu and Junwei Han. A deep spatial contextual long-term recurrent convolutional network for saliency detection. *arXiv preprint arXiv:1610.01708*, 2016.
- [31] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.
- [32] Yu-Fei Ma, Xian-Sheng Hua, Lie Lu, and Hong-Jiang Zhang. A generic framework of user attention model and its application in video summarization. *IEEE transactions on multimedia*, 7(5):907–919, 2005.
- [33] Sophie Marat, T Ho Phuoc, Lionel Granjon, Nathalie Guyader, Denis Pellerin, and Anne Guérin-Dugué. Spatio-temporal saliency model to predict eye movements in video free viewing. In *Signal Processing Conference, 2008 16th European*, pages 1–5. IEEE, 2008.
- [34] Stefan Mathe and Cristian Sminchisescu. Dynamic eye movement datasets and learnt saliency models for visual action recognition. *Computer Vision–ECCV 2012*, pages 842–856, 2012.

- [35] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Advances in neural information processing systems*, pages 2204–2212, 2014.
- [36] Andrea Palazzi, Davide Abati, Simone Calderara, Francesco Solera, and Rita Cucchiara. Predicting the driver’s focus of attention: the dr(eye)ve project. *CoRR*, abs/1705.03854, 2017.
- [37] Junting Pan, Elisa Sayrol, Xavier Giro-i Nieto, Kevin McGuinness, and Noel E O’Connor. Shallow and deep convolutional networks for saliency prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 598–606, 2016.
- [38] Robert J Peters and Laurent Itti. Applying computational tools to predict gaze direction in interactive visual environments. *ACM Transactions on Applied Perception (TAP)*, 5(2):9, 2008.
- [39] Jay Pratt, Petre V Radulescu, Ruo Mu Guo, and Richard A Abrams. It’s alive! animate motion captures visual attention. *Psychological Science*, 21(11):1724–1730, 2010.
- [40] Adria Recasens, Aditya Khosla, Carl Vondrick, and Antonio Torralba. Where are they looking? In *Advances in Neural Information Processing Systems*, pages 199–207, 2015.
- [41] Dmitry Rudoy, Dan B Goldman, Eli Shechtman, and Lihi Zelnik-Manor. Learning video saliency from human gaze using candidate selection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1147–1154, 2013.
- [42] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.
- [43] Evan Shelhamer. CaffeNet. https://github.com/BVLC/caffe/tree/master/models/bvlc_reference_caffenet, 2014.
- [44] Ludovic Simon, Jean-Philippe Tarel, and Roland Brémond. Alerting the drivers about road signs with poor visual saliency. In *Intelligent Vehicles Symposium, 2009 IEEE*, pages 48–53. IEEE, 2009.
- [45] Christopher Thomas. Opensalicon: An open source implementation of the salicon saliency model. *arXiv preprint arXiv:1606.00110*, 2016.
- [46] Antonio Torralba, Aude Oliva, Monica S Castelano, and John M Henderson. Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search. *Psychological review*, 113(4):766, 2006.

- [47] Geoffrey Underwood. Visual attention and the transition from novice to advanced driver. *Ergonomics*, 50(8):1235–1249, 2007.
- [48] Wenguan Wang, Jianbing Shen, and Ling Shao. Consistent video saliency using local gradient flow optimization and global refinement. *IEEE Transactions on Image Processing*, 24(11):4185–4196, 2015.
- [49] Chuan Yang, Lihe Zhang, Huchuan Lu, Xiang Ruan, and Ming-Hsuan Yang. Saliency detection via graph-based manifold ranking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3166–3173, 2013.
- [50] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.
- [51] Jianming Zhang and Stan Sclaroff. Saliency detection: A boolean map approach. In *Proceedings of the IEEE international conference on computer vision*, pages 153–160, 2013.