Efficient Deep Learning Methods for Vehicle Incident Prediction

A THESIS PRESENTED BY

Monica D. Song

TO

The Department Of Computer Science

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF Bachelor of Arts
IN THE SUBJECT OF Computer Science

Harvard University
Cambridge, Massachusetts

March 2019

Thesis advisor: Aude Oliva Monica D. Song

Efficient Deep Learning Methods for Vehicle Incident Prediction

Abstract

The purpose of this thesis is to detail a step-by-step process of leveraging deep learning to solve an issue in the autonomous driving industry. We aim to predict the occurrence of traffic-related incidents using solely the visual information available in dashcam videos and deep learning models that prioritize efficiency. We propose a bottom-up approach, starting with the simplest models and building up from there. We begin with the most common deep-learning visual recognition system, the Convolutional Neural Network (CNN), and assess its ability to predict and identify incidents from single images. This method of single-frame prediction is surprisingly able to capture relationships between frames despite assuming independence between frames. As a logical next step, we add temporal structure to the model to capture the motion dynamics of our video data. We consider two approaches: 1) Linear Layer Feature Concatenation (LLFC) and 2) Recurrent Neural Network (RNN), and compare their performances side-by-side. We investigate the effect of changing sequence length, frame rate, and type of feature vector on the accuracy and efficiency of these two models. We find that the two models have comparable accuracies, but different evaluation times and feature vector preferences. Lastly, we explore our solution's ability to generalize to a vaster scale of data and real-life settings.

Table of Contents

List of Tables

List of Figures

Acknowledgments

I want to express my gratitude to the people who have helped me along my journey in computer science and machine learning research. First, I would like to thank my advisor Aude Oliva for being so willing to accept me to her lab and for providing me the resources necessary to carry out my research. She has been an excellent role model.

I would like to thank Mathew Monfort for his academic and technical guidance. He has been a great source of new ideas, always encouraging about every research direction I was going to pursue. I would like to thank the Toyota Research Institute (TRI) for giving me the opportunity to attend my first academic conference and Ruth Rosenholtz for taking me under her wing at the TRI workshop. Also, thank you to Ben Wolfe for collecting the data, which no doubt saved me months of effort, and taking the time to explain the intricacies of the data annotation process to me.

I am grateful for the team at Perceptive Automata for allowing me the freedom to mess around with their GPUs during my summer there and for teaching me how deep learning is done "in the wild." Moreover, it was a Friday lunch journal club in which we read a paper on scene understanding that serendipitously led me to Aude Oliva's work at MIT CSAIL.

Last but not least, I would like to thank my parents for their unwavering support throughout my entire life. This thesis would not have been possible without them.

1

Introduction

Imagine it is a Friday afternoon and you are behind the wheel of a Tesla Model 3, driving your normal commute back home on the 210 Freeway. All of a sudden, you hear a high monotone beep and the steering wheel slides beneath your grip, your vehicle swerving to the left and and your shoulder bumping into the window. You look to your right, and you realize that while thinking about your weekend plans, you failed to notice an errant driver merging into your lane, narrowly missing a collision.

This situation, albeit over-dramatic, captures that awe-inspiring "magic" of how machine learning can foresee abnormal behavior before humans can and possibly rescue us

from peril. While certainly not as theatrical, our approach attempts to emulate and supercede human levels of accident prediction by way of a small corpus of YouTube videos and a few deep learning network architectures.

Motivation

One of the key practices of safe and defensive driving is looking ahead and expecting the unexpected. Human drivers are very good at this [1]. We are able to comprehend the details of a scene and classify scenes in just a couple hundred milliseconds [2], [3], a skill that is imperative to avoiding accidents. In this work, we aim to simulate this almost superhuman ability by developing deep-learning models to predict vehicle-related incidents using only video data.

We have an important goal of prioritizing our model's speed and efficiency, which we achieve via two techniques: minimizing the amount of layers in our model and minimizing the amount of information we give to the model. This requirement will allow us to run on our networks on mobile devices with limited computational power in a future project.

There has been a trend among researchers of using very deep, many-layered models [4]–[6] , which are difficult to apply to online prediction due to their hefty size. Some work has been done on making deep learning models more efficient: Zhang et al. propose methods of model compression [7] and Hettinger propose a greedy, single-layer method of training [8]. Additionally, several people have been able to train very shallow

networks for tasks such as text classification and object recognition [9]–[11]. In addition to faster evaluation times, another advantage of shallow models is that they are easier to train, significantly lowering the difficulty of network optimization.

Secondly, our method differs from the vast majority of deep learning work in the autonomous driving space in that we only have one source of information about the vehicle's surroundings, while most self-driving technology uses a larger number of sensors including LIDAR and GPS in addition to video data [12]. Nonetheless, this constraint helps us to achieve network efficiency, presenting a very unique opportunity of development in the field of autonomous vehicles.

1.1 Related Work

The problem of predicting collisions and other hazardous traffic situations has attracted much attention over the past decade as the race to perfect the first self-driving car continues. A range of commercials interests from Aurora and Argo AI to Waymo and Zoox have been working on developing autonomous vehicle technology powered by predictive deep learning [13]. In addition to industry-backed research, there has been a wide range of academic work devoted to this topic [14]–[16]. With regard to the prediction of traffic accidents, several people have modeled the risk of vehicle collision [17], [18] and done direct work to predict traffic accidents [19]–[21]. While Chen [19] and Yuan [21] use multiple modes of information such as GPS and inter-vehicle

communication as well as other human-annotated details such as road, weather, time, and traffic factors, Chan [20] only uses dashcam video (as do we).

Chan uses an RNN with LSTM units and a dynamic-spatial-attention mechanism to tailor their model to focus on vehicles, pedestrians, and other objects in the scene. However, their approach uses an object-detection algorithm Faster-RCNN in order to identify the important objects in the image. While highly accurate, object-detection algorithms like Faster-RCNN come at a computational cost, requiring a GPU to to be able to process images fast enough for online prediction [22]. Furthermore, in our Class Activation Mappings in Section 3, we show that detecting objects and creating bounding boxes may not be necessary.

Overview

The main contribution of this thesis is a bottom-up construction of an end-to-end machine learning model that can make predictions about dangerous driving scenarios from video data alone. We adopt a systematic process of developing our model, assessing the effectiveness of the model in a series of experiments, described in Section 2.3, to see how we can improve it further at each stage.

In Stage One (Section 3), we begin by tuning an off-the-shelf Resnet-18. To correct for the ResNet's overfitting, we also design a shallower convolutional network BasicNet that is one-tenth the size of the ResNet. These two Convolutional Neural Networks (CNNs) in Stage One set the stage for our next set of models. In Stage Two

(Section 4), we create two models that incorporate time using the feature vectors learned by the CNNs in Stage One. The first time-dependent model is very simple; by concatenating the feature vectors, we see if we can capture changes between frames. The second model is an RNN with LSTM units meant to learn the dynamics as well as any long-term dependencies that exist between feature vectors of consecutive frames. We tune several hyperparameters to evaluate how well these two models can form and store memories of previous frames. We compare these models to each other, as well as to a modified version of our models from Stage One. Through this process of iteration, we hope to demonstrate good practices in machine learning as part of a scientifically rigorous approach to this discipline.

To conclude, we move the discussion beyond analysis of our models and explore whether it makes sense to use deep learning in the context of this problem. We consider our work a practical project, the goal of which is to demonstrate a step-by-step process of applying machine learning to solve a salient problem in the autonomous driving industry. Thus, it is important that we examine the potential of our results to translate to circumstances outside the scope of this work.

2

Problem Description

We define the problem as follows: we want to create a model that is able to predict vehicle-related incidents from dashcam video. We benchmark the model's performance with that of three human annotators. We are subject to two constraints: we only have the video as our one source of information, and we want to avoid extremely deep networks, for the future purpose of creating efficient networks that are able to run on machines with low-computational power.

2.1 Dataset

The dataset consists of 279 videos, all filmed on dashboard cameras and downloaded from YouTube. The videos contain a diversity of "**incidents**" i.e. potentially hazardous behavior from other actors that would cause a human driver to pay extra attention to the road and take some additional action such as swerve, slow down, or stop.

Incident Cue

Each video also has a human-annotated "incident cue" specifying the time (the average of three different annotators) in the video when the annotator *anticipates* that an incident will occur.

Data Splits

We split the videos into three categories: 181 videos in train set, 48 in the validation set, 50 in the test set. From each video, we sample around around 50 clips, also known as sequences. We label frames before the accident cue with label = 0 and all frames after and include the incident with label = 1. Therefore, we formulate our problem as a supervised-learning **binary classification task,** in which we train models to classify images as positive or negative for containing an incident.

2.2 Models

We train several binary classifiers. In determining the design of our models, we begin with the basic building blocks of computer vision and layer on additional elements in stages. Our approach to the problem thus follows a natural progression of increasing complexity. The models are summarized in Table 2.1.

Model	Input Dimension	No. Layers	No. Parameters	Recurrent?
ResNet-18	3 x 224 x 224	18	11.7 million	No
BasicNet	3 x 224 x 224	5	140,000	No
LLFC	1 x (Seq. Len. x Vector Len.)	1	(Seq. Len. x Vector Len.)	No
RNN	(Seq. Len) x (Vector Len.)	2	116,000; 181,000	Yes

Table 2.1: Model summaries

2.2.1 Convolutional Neural Network

Working with image data makes training a feedforward[1] CNN, a deep neural network most commonly applied to analyzing visual imagery [23], a natural first step. We test two different CNNs.

ResNet-18

We begin with the ResNet-18 (Figure 2.1), due to its convenient availability as a pre-trained neural network in PyTorch. ResNets are a family of networks known for their state-of-the-art performance on ImageNet in classifying objects into 1000 categories [24]. We fine-tune an 18-layer ResNet pretrained on ImageNet on our much smaller dataset to perform this binary classification task, a procedure known as "transfer learning" whose benefits have been well-documented [25], [26]. However, given the reality that our train

[1] Feedforward means means that information flows through the model in one direction only, from input to output. There are no feedback connections in which outputs of the model are fed back into itself.

set contains just under 7000 images and the ResNet-18 has over 11 million parameters, the ResNet-18 (the shallowest of the ResNet family) is still too deep. Its quick convergence on the train set, achieving 100% after one epoch, confirms this fact; thus, we look to create a shallower CNN.

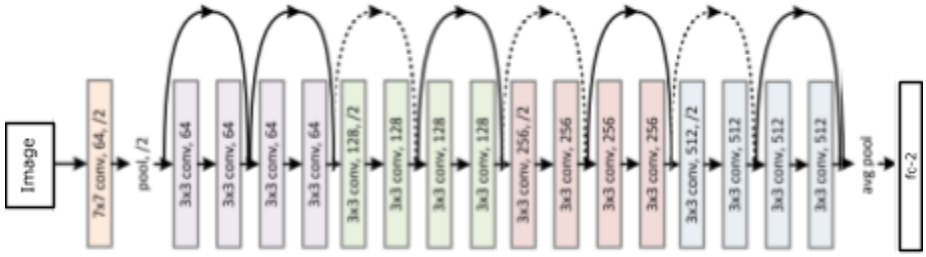


Figure 2.1: ResNet-18 architecture, fine-tuned to two output classes

BasicNet

To avoid issues with overfitting, we design and train a custom five-layer CNN, called BasicNet (Figure 2.2), that contains 140,000 parameters. The small number of layers in BasicNet keep us on track to achieving our goal of maximizing efficiency.

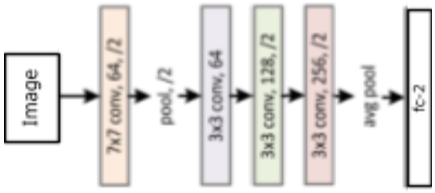


Figure 2.2: BasicNet architecture

2.2.2 Linear Layer Feature Concatenation

Noticeably missing from CNNs is the ability to incorporate temporal dependency between examples in our data, i.e. CNNs ignore our data's dimension of time. A simple way to incorporate temporal information into our model is to concatenate the feature vectors[2] of consecutive frames from the CNN into one long feature vector, allowing for the model to take into account more than one frame at a time. Thus, we implement a layer that performs a linear transformation on the concatenated feature vector, resulting in a single probability output (Figure 2.3).

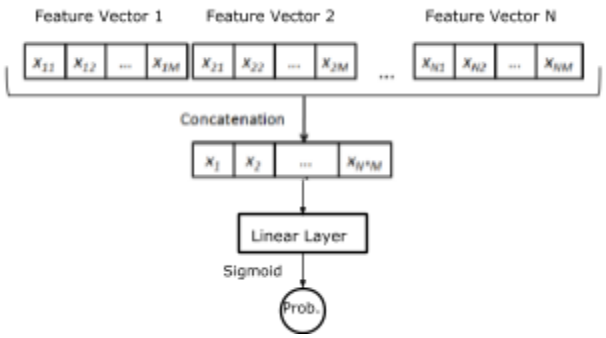


Figure 2.3: Linear Layer Feature Concatenation (LLFC)

2.2.3 Recurrent Neural Network

[2] A feature vector is a vector of numeric values that describe the image, compressing the amount of values from the RGB image (3 x 224 x 224 = 150528) into a vector that is a fraction of the original size of the image.

A second way to incorporate time is with a Recurrent Neural Network that is able to keep track of scores from previous frames. RNNs use their internal state to process sequences of inputs. In our work, we use an RNN with a Long Short-Term Memory (LSTM) unit (Figure 2.4) containing two hidden layers, each of length 64. LSTMs provide a solution by incorporating memory units that explicitly allow the network to learn when to "forget" previous hidden states and when to update hidden states given new information [27].

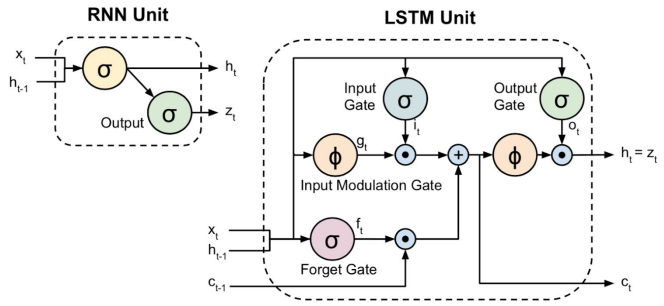


Figure 2.4: Traditional RNN unit (left), LSTM unit (right) [27]

The input of the RNN is a sequence of feature vectors, each the result of feeding its associated image into the CNN (Figure 2.5). The RNN processes this sequence of feature vectors, outputting a probability of incident.

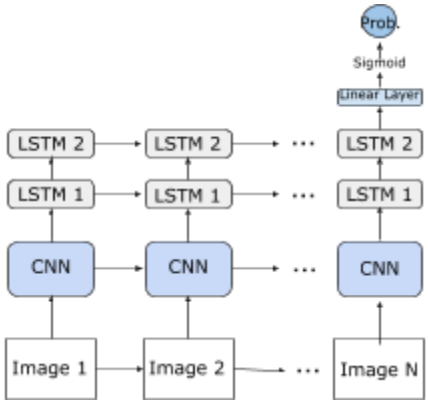


Figure 2.5 RNN architecture

2.3 Experiments

We divide our experiments into two cases based on whether the model takes into account the temporal dimension of our data:

- Stage 1: Single Frame Prediction (Section 3)
- Stage 2: Multi-Frame Prediction (Section 4)

In Single Frame Prediction, we assess the effectiveness of using only the feedforward ResNet-18 and BasicNet, where the model is to asked to predict the occurence of an incident using only single frames. In Multi-Frame Prediction, we give our two models, LLFC and RNN, a sequence of images and ask whether the sequence contains an incident. We test several variables: type of CNN used, sequence duration, and frame rate. In order to validate our assumption that adding temporal information will improve scores, we test against a "baseline" model that simply averages scores of single frames. This GitHub repository contains the code for all the experiments.

3

Single-Frame Prediction

In Single-Frame Prediction, we evaluate the effectiveness of BasicNet and ResNet-18 on single frames extracted from the video, in which no *information is shared between frames*.

We begin with feeding individual frames into a Convolutional Neural Network that only uses information in the single static frame to classify the frame as being negative or positive for containing an incident. To emphasize again, this approach does not take into account the information from neighboring frames; thus the model is unable to learn motion or physics cues.

Analysis

The models have limited success on this binary classification task. The ResNet-18 achieves an accuracy of 56% on the test set (Table 3.1), and BasicNet achieves an accuracy of 54% (Table 3.2). On one hand, given the almost impossible nature of this task, doing better than the random on a balanced dataset is noteworthy, yet a 56% accuracy rate leaves much to be desired.

Furthermore, in comparing the two models' performances, we learn that the feature vectors from ResNet-18 are not significantly more representative of the images than our much shallower CNN variant BasicNet. Since we are classifying the images into two categories, the more complex design of the ResNet that allow it to achieve top scores on ImageNet may not be necessary for us.

ResNet-18

	Actual: +	Actual: -	Accuracy: 0.56 Precision: 0.60 Recall: 0.56 F-Score: 0.58
Predicted: +	1430	1085	
Predicted: -	1129	1474	

Table 3.1: ResNet-18 classification scores for single frames

BasicNet

	Actual: +	Actual: -	Accuracy: 0.54 Precision: 0.55 Recall: 0.54 F-Score: 0.55
Predicted: +	1408	1151	
Predicted: -	1193	1366	

Table 3.2: BasicNet classification scores for single frames

Additionally, for some videos, the networks are able to achieve some semblance of "time dependency" between frames, assigning consecutive frames similar probabilities. Our ideal model would output a smooth sigmoidal curve of probabilities over the course of the frame sequence, with a sharp increase in the output around the human incident cue. The sigmoidal curve represents the ultimate goal that we are aiming for (Figure 3.1).

On certain sequences, both the ResNet-18 and BasicNet are able to achieve this desired pattern when we average the scores over ten frames (Figure 3.2). A closer analysis reveals that these sequences tend to contain objects moving perpendicular to the driver's plane of view.

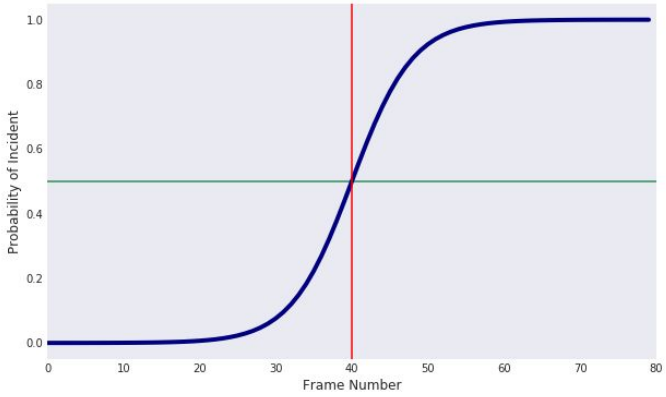


Figure 3.1: The Goal Curve. The desired pattern follows a sigmoidal curve such that the model's predictions begin increasing prior to the incident cue, hitting 0.5 around the incident cue with a sharp increase shortly after.

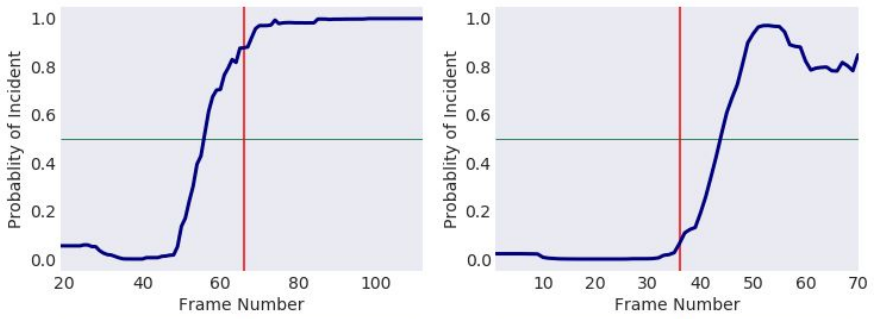


Figure 3.2. Strong performance on single-frame prediction task from ResNet-18. The model's scores are smoothed over 10 frames.

However, examples where the models exhibit the sigmoidal pattern only comprise about 20% of the videos in the test set. The other 80% of the videos set have random, erratic spikes in the model outputs and do not follow such a smooth pattern (Figure 3.3).

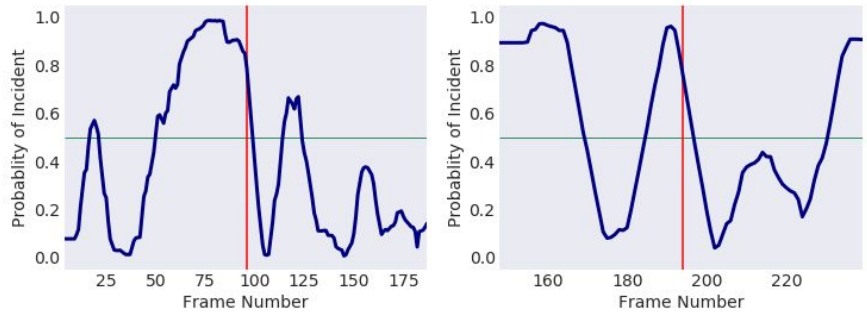


Figure 3.3. Weak performance on single-frame prediction task from ResNet-18. Note the erratic spikes in the model's probabilities.

Class Activation Mapping

While both ResNet-18 and BasicNet seem equally lackluster, we can confirm that the ResNet-18 localizes objects significantly better than BasicNet using a method called Class Activation Mapping (CAM). CAM uses the global average pooling layer in CNNs to visualize the discriminative regions [28]. ResNet-18 highlights (in red) the turning vehicle (Figure 3.4) and pedestrian (Figure 3.5), while BasicNet fails to pick up on these regions.

Figure 3.4: ResNet (top) vs BasicNet (bottom) CAM on turning vehicle sequence

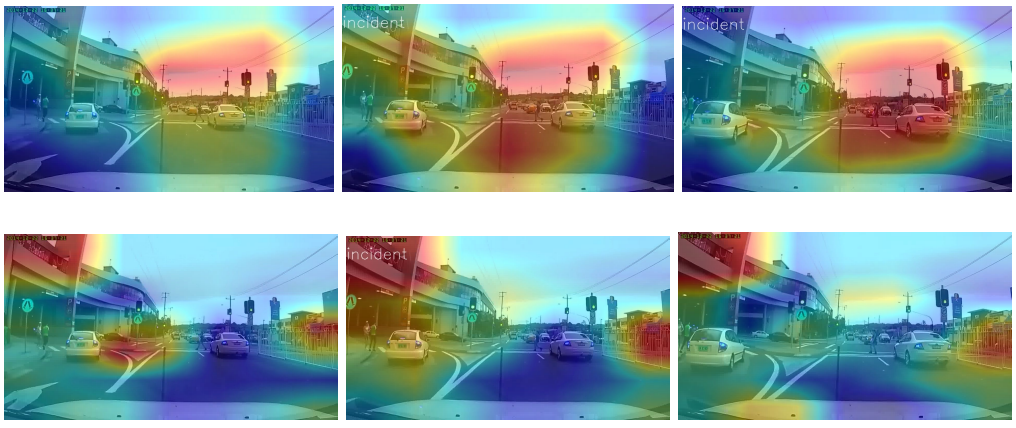


Figure 3.5: ResNet (top) vs BasicNet (bottom) CAM on crossing pedestrian sequence

Thus, while the ResNet-18 only produces a marginal gain in accuracy, it has an incredible ability to localize objects in the image.

With a hit-or-miss performance on identifying the time of the incident, single frame prediction with ResNet-18 and BasicNet leaves much room for improvement but provides a solid foundation for future models that incorporate temporal information. This experiment suggests that the feature vectors from the convolutional networks are useful in compressing and representing the scene details of the images in our dataset.

4

Multi-Frame Prediction

In Stage Two: Multi-Frame Prediction, we incorporate temporal dynamic behavior such that we can retain information from previous frames. We develop two new models that incorporate time: Linear Layer Feature Concatenation (LLFC) and Recurrent Neural Network (RNN) with LSTM. We compare them to two "Baseline" models, in which we modify BasicNet and ResNet-18 from Stage One by averaging their scores over frames in a sequence to get their predictions. We measure the effect of several different variables (detailed in Table 4.1) on the LFFC and RNN's performance on the validation set.

Model	Feature Vector, Length	Sequence Duration	Frame Rate
LLFC RNN	ResNet-18, 512 BasicNet, 256	0.4 secs 0.6 secs 0.8 secs 1.0 secs 1.2 secs	10 frames/sec 5 frames/sec

Table 4.1: Multi-Frame prediction variables for Stage Two models. We test 2 new models, 2 feature vectors, 5 sequence durations, and 2 frame rates, resulting in 40 experiments total.

4.1 Comparison to Baseline

Accuracy

We evaluate the Baseline models on sequences of varying length. While these Baseline models achieved accuracies of 54-56% on single-frame prediction, these percentages dramatically increase by 15 to 20 points once we average scores over multiple frames (Table 4.2) showing that inter-frame changes are incredibly informative. Nonetheless, both Stage Two models outperform Baseline models by around nine percent (Table 4.2).

Sequence Length	Linear Layer FC		RNN		Baseline	
	BasicNet	ResNet	BasicNet	ResNet	BasicNet	ResNet
4	0.780	0.764	0.776	0.833	0.744	0.743
6	0.783	0.765	0.754	0.796	0.725	0.738
8	0.768	0.758	0.793	0.812	0.708	0.685
10	0.795	0.756	0.795	0.810	0.699	0.690

12	0.751	0.758	0.744	0.803	0.669	0.687

Table 4.2: Accuracy on validation set of the LLFC and RNN with two variants of feature vectors from BasicNet and ResNet-18, compared to Baseline models

Time

We evaluate the Baseline models on a sequence of 50 frames, averaged over 100 trials. We find that the ResNet-18 Baseline takes slightly longer by on average 141 milliseconds than BasicNet Baseline (Table 4.3), which is reasonable given its additional depth.

Baseline Model	Time
BasicNet	1.355 sec
ResNet-18	1.496 sec

Table 4.3: Time to extract 50 feature vectors for Baseline models

We note that by construction of our Stage Two models LLFC and RNN, the Baseline models will always be faster since Stage Two models rely on the output of the Baseline to calculate their scores. These Stage Two models take on average around 62 milliseconds of additional computational time to process a sequence of 50 frames (Table 4.4), which amounts to around a 4.2% increase in computation time.

Thus, in using more complex temporal techniques, we achieve a nine percent increase in accuracy at a cost of a four percent time increase.

Sequence Length	LLFC		RNN	
	BasicNet	ResNet-18	BasicNet	ResNet-18
4	38.4 msec	38.9 msec	65.0 msec	76.8 msec
6	39.8 msec	40.5 msec	71.7 msec	85.6 msec
8	41.3 msec	42.9 msec	76.7 msec	87.6 msec
10	43.5 msec	44.7 msec	82.7 msec	93.7 msec
12	46.8 msec	43.5 msec	88.9 msec	107.8 msec

Table 4.4: Time to evaluate a frame sequence of 50 images for Stage Two models. These times are measured after the extraction of feature vectors by the CNN.

4.2 Varying Sequence Length

We are interested in seeing how the amount of prior information the model receives factors into its ability to accurately anticipate and classify the arrival of a hazardous incident.

Accuracy

We note that for both Stage Two models, as sequence length increases, the accuracy of the model does not increase monotonously, as one might expect (Table 4.2). Instead, for the LLFC, the accuracy decreases as sequence length increases. This is likely due to the act of feature concatenation, which results in one long feature vector that ignores the distinction between frames. Thus, concatenated vectors that are extremely long may confuse the linear layer if it fails to learn that these long inputs should be segmented. The RNN interestingly achieves its lowest accuracy scores at the longest sequence length. However, it

is also able to achieve some of its highest scores on sequences eight and ten frames in length. The RNN processes feature vectors one by one; thus feeding it more feature vectors should in theory provide it more information about inter-frame dynamics.

Additionally, the LLFC achieves its optimal performance using BasicNet feature vectors, while the RNN always does better with the ResNet-18 features (Table 4.2). As a result, we have an interesting pairing in our hands: the LLFC performs better with BasicNet while the RNN better with ResNet-18. The extra hidden layer in the RNN may be responsible for this as it is likely learning a non-linear relationship.

Effect on Time

As expected, the evaluation time increases monotonously with sequence length (Table 4.4). We examine the relationship between time and accuracy as a function of sequence length to see if there is a trade-off. First, we note that the RNN achieves higher accuracies than the LLFC in seven out of ten cases (Table 4.2). However, the RNN without fail takes a longer time than the LLFC to evaluate sequences (Table 4.4), a consequence of its tenfold increase in parameters. We reason that the RNN's hidden layers calculate some useful transformation of the data, but at a computational cost. Within models, we do not observe such a strong correlation between accuracy and time (Figure 4.1). This could be because acquiring additional information, which necessitates more computational resources, actually has the harmful effect of confusing the model.

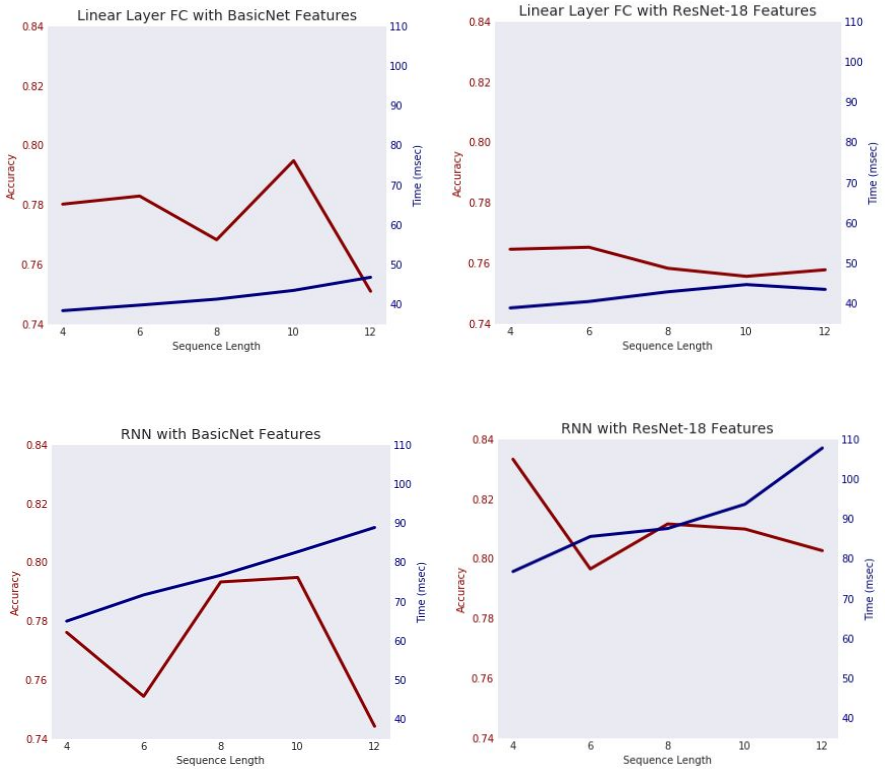


Figure 4.1: Comparison of accuracy (red) and evaluation time (blue) for LLFC (top) and RNN (bottom). There does not appear to be any strong relationships between accuracy and time.

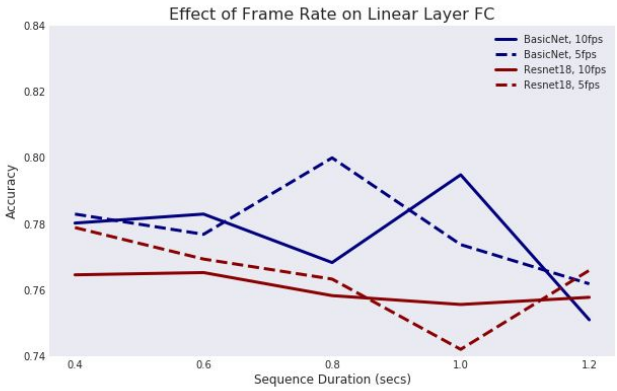
4.3 Varying Frame Rate

We compare two values of frame rates: the default rate of 10 fps and a 50% slower rate of 5 fps. This experiment is important for future applications in which it may be impossible for the model's hardware to extract and analyze frames at the desired speed of 10 fps or greater. Thus it would be beneficial for us to know how these models perform in

resource-constrained environments, in which there may be a trade-off between the speed of information processing and model accuracy.

Analysis

Interestingly, we observe no such compromise. On the contrary, for both the LLFC and RNN, the models perform better with the slower frame rate on BasicNet vectors and perform around the same level for ResNet-18 vectors (Figure 4.2). We attribute this finding to the fact that the greater change in the scenery between frames with the slower frame rate helps the model learn the dynamics of these videos better. This result shows us that time-dynamic models may be well-suited for resource-constrained environments that suffer from information scarcity.



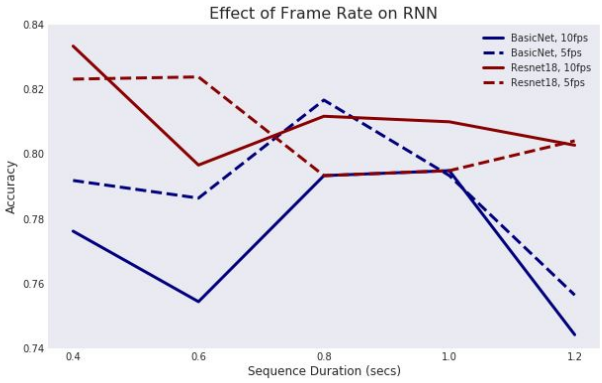


Figure 4.2: Accuracy of the models (top: LLFC, bottom: RNN) with frame rates of 10 and 5 frames per second

Through testing these variables, we discover several interesting relationships. First, we find that the LLFC and RNN outperform the Baseline models by about nine percent at a four percent increase in computational time. In our various experiments, we find that the LLFC performs better with BasicNet vectors while the RNN with ResNet-18 vectors, a relationship we attribute to the nonlinearity in the hidden layers of the RNN. We also observe that the LLFC prefers short sequences of feature vectors, while the RNN prefers short to medium length sequences. While computation time increases with sequence length, we note that accuracy does not necessarily increase as well, which tells us that going beyond the memory of four frames is not very helpful. Our last experiment with halving the frame rate confirms this idea that more information does not always equal better.

5

Evaluation on Test Set

We now evaluate our models on the held-out test set in order to investigate how well they translate to the "real world." We select three variants of models based on their performance on the validation set in the previous section to capture the diversity of parameters that we have tested in our experiments:

- RNN with ResNet-18 vectors with sequence length 4, which had the **highest accuracy**
- LLFC with BasicNet vectors with sequence length 4, which had the **fastest time**
- BasicNet Baseline with scores averaged over 4 frames, for comparison

We use the default frame rate of 10fps to make comparing these three models easier.

Quantitative Analysis

We find that the RNN has a higher classification accuracy than the LLFC and Baseline (Table 5.1). However, we note that the LLFC and Baseline both have higher recall rates, i.e. the ability to correctly identify sequences containing an incident. This tells us that these two models tend to to over-classify sequences as containing an incident, which may be desirable for a real-life accident prevention system where safety is the number one concern.

Model	Accuracy	Recall	Precision	F1
RNN with ResNet-18	0.749	0.547	0.735	0.627
LLFC with BasicNet	0.723	0.584	0.670	0.620
Baseline	0.679	0.572	0.666	0.615

Table 5.1: Classification scores of models on test set

Qualitative Analysis

We plot the model's probabilities on a *randomly selected* 21 out of the 50 videos in the test set (Figure 5.1). The RNN is able to achieve a significantly closer resemblance of the desired sigmoidal pattern (Figure 3.1) than the LLFC and Baseline. In particular, the RNN's output on videos 2, 4, 5, 8, 9, 11, 18, 20, and 21 is quite ideal. On these sequences, the LLFC and Baseline are also able to achieve similar sigmoidal patterns on these videos

but with a greater amount of noise. We discuss the consequences of these results in the next section. The results of our models' predictions can be viewed here.

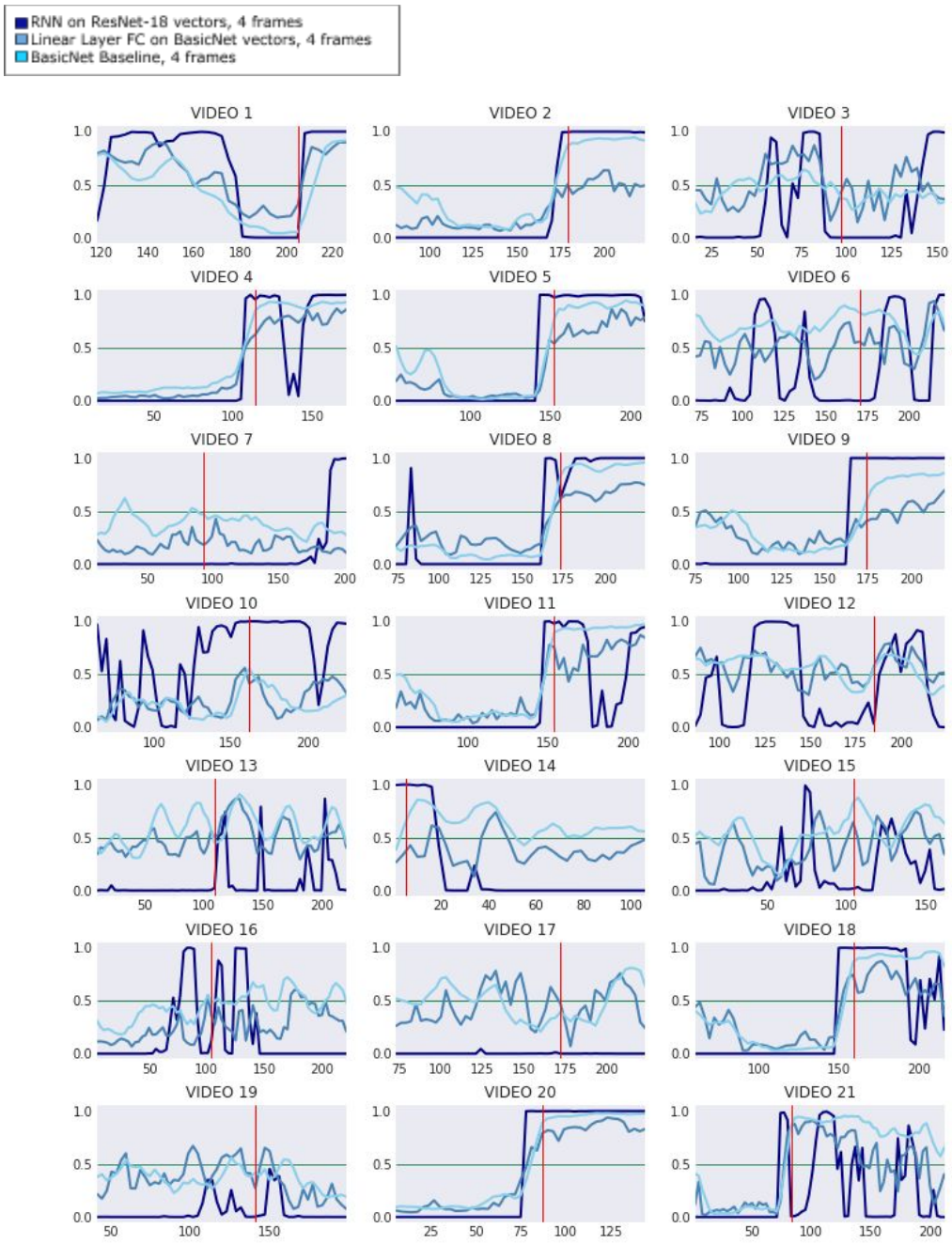


Figure 5.1. Comparison of three models on test set. The RNN most closely emulates the desired sigmoidal pattern, while the LLFC and Baseline are more unstable.

6

Discussion

The applied nature of this work calls for discussion of our models' generalizability to a larger corpus of data, as well as of how appropriate deep-learning is for solving the problem at hand.

6.1 Limitations

From the outset, we assumed that this problem of incident prediction could be tractably solved by deep learning. Here we analyze this assumption.

Size and Diversity of Dataset

First, we note that the dataset only contains 279 videos, which is relatively small in the arena of deep learning. Our train, validation, and test splits each contain several thousand examples. Thus, we are unable to harness the advantage of deep learning in its ability to scale to "big data" in this setting.

However, our dataset also contains a huge diversity of traffic-related incidents, such as unsafe lane merges, trucks flipping over, cars colliding, pedestrians jaywalking, motorcycles skidding. Thus, despite the small size of this dataset, its variety of activities forces the model to learn features that are not specific to one type of incident. Additionally, given the diverse nature of our data, it is difficult to summarize with human engineered features; thus deep-learning models may learn more useful features.

Generalizability

One of the advantages of machine learning is the ability of models to generalize to unseen data. The performance of our models on the validation and test sets of 70 - 75% accuracy are evidence that the models were able learn some key predictors of incidents. We highlight some specific cases of success where the model achieved a goal sigmoidal curve (Figure 5.1):

- Video 2: A falling block of snow crashes on the driver's window.
- Video 5: An oncoming vehicle makes an unsafe left turn.
- Video 8: A sedan careens into the side of a large truck.
- Video 9: A vehicle in the right lane makes an unsafe lane merge.
- Video 11: The vehicle ahead makes a U-turn.
- Video 18: A vehicle proceeds unsafely at a T-intersection.

- Video 20: A vehicle makes an unsafe lane merge to the right.
- Video 21: A vehicle makes an unsafe lane merge to the left.

It appears that the model is able to pick up on other vehicles that enter the driver's field of view and come dangerously close to the driver. However, there is still a large fraction of sequences where the model gives very wrong predictions, either not detecting an incident at all or spiking well before the incident. We highlight some specific cases of failure:

- Video 3: Premature detection. The driver loses control of his own vehicle due to icy road conditions.
- Video 6: Premature detection. The appearance of pedestrians in the scene before the incident confuses the model.
- Video 7: Late detection. A vehicle starts unsafely merging into the driver's lane from the right but then stops.
- Video 10: Premature detection. There are many cars parallel parked on the narrow residential street, confusing the model.
- Video 17: No detection. Driver fails to heed to a crossing pedestrian, almost hitting him.
- Video 19: No detection. A passenger exits the vehicle in left lane at a red light, unsafely opening the door.

These failure cases *tend to contain humans*. It is a well-documented phenomenon in the self-driving industry that humans are unpredictable actors, and much academic and industry research has been dedicated to predicting the intent of pedestrians [29]–[31]. We recognize that our problem of incident prediction becomes exponentially more difficult when dealing with human agents.

Thus, we see that our models have learned to predict the hazard of other vehicles interfering with the driver's own trajectory, while there is still much work to do when dealing with the complex dynamics of human actors.

Hyper Parameter Tuning

We performed a cursory hyperparameter tuning in the spirit of binary search to find the best set of learning rates, optimizers, and loss functions to use. However, it was by no means exhaustive. As evidenced by the high training error (30%) for Linear Layer FC (Figure 6.1), this set of hyperparameters will need to be further refined for different models.

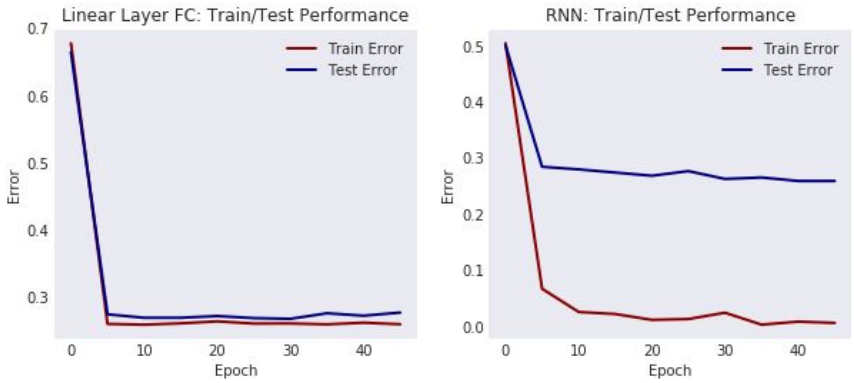


Figure 6.1: Train/test performance for Stage Two models over 50 epochs

6.2 Future Work

There are many novel directions that we can explore further. This work excites us because given the small scale of our dataset, we have already achieved presentable results. Coupled

with the influx of more labeled data, the following more complex variants of our model will no doubt achieve better performance.

Many-to-Many RNN

A Recurrent Neural Network that takes as input a sequence of feature vectors and outputs a sequence of labels will allow us to more adequately capture the desired sigmoidal function. The many-to-many RNN will give another another dimension of analysis in that we can also study how the scores change over a single sequence.

Hybrid RNN-CNN

This hybrid model, in which we tune the weights of the CNN and RNN concurrently, will be better optimized to multi-frame prediction. This model is highly similar to the Long-term Recurrent Convolutional Network (LRCN) Model that Donahue et al. propose [27]. We expect this model to have intermediary feature vectors that will more precisely capture the subtle changes between frames.

7

Conclusion

We have presented an efficient, end-to-end deep learning system for vehicle incident prediction. Starting with single-frame prediction, we build up to multi-frame prediction, achieving solid results that make us excited for future improvements and applications.

In pursuing our goal of developing an accurate and efficient model on our small and heterogeneous dataset, we have demonstrated notable progress. We find in Stage One that single image prediction does poorly, which we explain by the lack of information shared between frames. However, in Stage Two, we find that the Baseline models and the two additional models we train have significantly higher accuracies. The surprising success

of the BasicNet Baseline, with a meager five layers, is a testament to the effectiveness of Convolutional Neural Networks for image data. We also find that the Linear Layer Feature Concatenation and the Recurrent Neural Network achieve around nine percent higher accuracy rates at a four percent increase in computational expense. Thus, increasing the temporal complexity of our models has a net benefit on their ability to forecast and detect vehicle-related incidents. Finally, we evaluate three models on the test set to assess their ability to generalize to unseen data. We conclude that they predict the risk associated with other vehicles fairly well but face difficulties when it comes to pedestrians.

We have followed a narrative of iteration and analysis, taking incremental steps to develop our model in a methodical fashion. Laying a rigorously established foundation allows us to precisely analyze the amount of improvement that more complex models achieve over their predecessors. We hope that by having the foresight to truly understand our foundational models, we will better navigate future developments in our research and contribute to the greater machine learning community.

Epilogue

My desire to write this thesis arose out of a curiosity to figure out what deep learning is and to understand why today machine learning is at the heart of every technology out there, a buzzword in media, a must-take class in every computer science student's curriculum, and such a pervasive term in today's day and age. With the vast amount of infrastructure, free courseware, open-source research, publically available datasets that are online today, one can become a machine learning expert quite easily. The machine learning ecosystem has a low barrier of entry, and for that I am thankful. Now, having conquered the many trials and tribulations that this thesis has thrown at me, I have removed some of the mysticism that machine learning and deep learning once held in my mind, walking away with greater confidence in myself as a computer scientist.

A native Californian, I have a vision of a future where I can drive down the Pacific Coast Highway while being able to fully embrace the view of the great blue ocean, eyes and hands off the wheel, wind blowing through my hair. But as we have seen for the past decade, so many kinks must be worked out before such a dream can be realized. With an amorphous, constantly delayed timeline, the advent of self-driving cars seems increasingly far off in the distance. Nonetheless, it is this carefree image of salt and sun that transcends my concerns with the industry, inspiring me to study this problem for the past year.

I end this thesis with a greater appreciation for the developers who spend their time perfecting open-source tools and admiration for the researchers who work on the cutting-edge of this rapidly evolving field.

References

[1] B. Wolfe *et al.*, "Perceiving the Roadway in the Blink of an Eye-Rapid Perception of the Road Environment and Prediction of Events," in *Proceedings of the 9th International Driving Symposium on Human Factors in Driver Assessment, Training, and Vehicle Design: driving assessment 2017*, Manchester Village, Vermont, USA, 2017, pp. 207–213.

[2] M. Potter, "Meaning in visual search," *Science*, vol. 187, no. 4180, pp. 965–966, Mar. 1975.

[3] J. M. Henderson and A. Hollingworth, "HIGH-LEVEL SCENE PERCEPTION," *Annu. Rev. Psychol.*, vol. 50, no. 1, pp. 243–271, Feb. 1999.

[4] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *ArXiv14091556 Cs*, Sep. 2014.

[5] Yuanjun Xiong, Kai Zhu, Dahua Lin, and X. Tang, "Recognize complex events from static images by fusing deep channels," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, 2015, pp. 1600–1609.

[6] J. Carreira and A. Zisserman, "Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset," *ArXiv170507750 Cs*, May 2017.

[7] Q. Zhang *et al.*, "Efficient Deep Learning Inference Based on Model Compression," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Salt Lake City, UT, USA, 2018, pp. 1776–17767.

[8] C. Hettinger, T. Christensen, B. Ehlert, J. Humpherys, T. Jarvis, and S. Wade, "Forward Thinking: Building and Training Neural Networks One Layer at a Time," *ArXiv170602480 Cs Stat*, Jun. 2017.

[9] H. T. Le, C. Cerisara, and A. Denis, "Do Convolutional Networks Need to Be Deep for Text Classification?," p. 8.

[10] K. Ashraf, B. Wu, F. N. Iandola, M. W. Moskewicz, and K. Keutzer, "Shallow Networks for High-Accuracy Road Object-Detection," *ArXiv160601561 Cs*, Jun. 2016.

[11] M. D. McDonnell and T. Vladusich, "Enhanced Image Classification With a Fast-Learning Shallow Convolutional Neural Network," *ArXiv150304596 Cs*, Mar. 2015.

[12] J. Levinson *et al.*, "Towards fully autonomous driving: Systems and algorithms," in *2011 IEEE Intelligent Vehicles Symposium (IV)*, Baden-Baden, Germany, 2011, pp. 163–168.

[13] "Self-Driving Cars: The Complete Guide | WIRED." [Online]. Available: https://www.wired.com/story/guide-self-driving-cars/. [Accessed: 26-Mar-2019].

[14] L. Fridman *et al.*, "MIT Autonomous Vehicle Technology Study: Large-Scale Deep Learning Based Analysis of Driver Behavior and Interaction with Automation," *ArXiv171106976 Cs*, Nov. 2017.

[15] M. Bojarski *et al.*, "End to End Learning for Self-Driving Cars," *ArXiv160407316 Cs*, Apr. 2016.
[16] H. Xu, Y. Gao, F. Yu, and T. Darrell, "End-to-End Learning of Driving Models from Large-Scale Video Datasets," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 2017, pp. 3530–3538.
[17] M. Strickland, G. Fainekos, and H. B. Amor, "Deep Predictive Models for Collision Risk Assessment in Autonomous Driving," *ArXiv171110453 Cs*, Nov. 2017.
[18] H. Ren, Y. Song, J. Liu, Y. Hu, and J. Lei, "A Deep Learning Approach to the Prediction of Short-term Traffic Accident Risk," p. 9.
[19] C. Chen, H. Xiang, T. Qiu, C. Wang, Y. Zhou, and V. Chang, "A rear-end collision prediction scheme based on deep learning in the Internet of Vehicles," *J. Parallel Distrib. Comput.*, vol. 117, pp. 192–204, Jul. 2018.
[20] F.-H. Chan, Y.-T. Chen, Y. Xiang, and M. Sun, "Anticipating Accidents in Dashcam Videos," in *Computer Vision – ACCV 2016*, vol. 10114, S.-H. Lai, V. Lepetit, K. Nishino, and Y. Sato, Eds. Cham: Springer International Publishing, 2017, pp. 136–153.
[21] Z. Yuan, X. Zhou, T. Yang, J. Tamerius, and R. Mantilla, "Predicting Traffic Accidents Through Heterogeneous Urban Data: A Case Study," p. 9.
[22] J. Huang *et al.*, "Speed/accuracy trade-offs for modern convolutional object detectors," *ArXiv161110012 Cs*, Nov. 2016.
[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.
[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," p. 9.
[25] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, USA, 2014, pp. 1717–1724.
[26] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 3320–3328.
[27] J. Donahue *et al.*, "Long-term Recurrent Convolutional Networks for Visual Recognition and Description," *ArXiv14114389 Cs*, Nov. 2014.
[28] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning Deep Features for Discriminative Localization," p. 9.
[29] G. Habibi, N. Jaipuria, and J. P. How, "Context-Aware Pedestrian Motion Prediction In Urban Intersections," *ArXiv180609453 Cs Stat*, Jun. 2018.
[30] E. Ohn-Bar and M. M. Trivedi, "Looking at Humans in the Age of Self-Driving and Highly Automated Vehicles," *IEEE Trans. Intell. Veh.*, vol. 1, no. 1, pp. 90–104,

Mar. 2016.
[31] S. Anthony, “Introducing Perceptive Automata: Human Intuition for Self-Driving Cars,” *Medium*, 10-Jul-2018. .