Estimating Drivers' Required Attention Levels via Dashcam Videos

Gary Corcoran

Department of Electrical and Computer Engineering

McGill University, Montreal

April 2018

A thesis submitted to McGill University in partial fulfillment
of the requirements of the dgreew of Master of Engineering.

Abstract

The problem being addressed in this research is estimating the driver's required attention level from outward-facing dashcam videos. This research takes a different approach to driver attention; assessing the driver's needs as opposed to the driver's performance. The input data consists of outward-facing dashcam videos in various driving situations. A two-stream recurrent convolutional neural network approach is used where one stream, the spatial stream, analyses individual video frames and computes high-level appearance features. The other temporal stream analyses optical flow between adjacent frames and computes high-level motion features. Both spatial and temporal features are then fed into a Recurrent Neural Network (RNN) that explicitly models the sequence of features in time. The dataset being used is a selection of 1750 dashcam videos collected by Chan *et al.* [3]. The dataset was previously annotated with positive and negative instances of vehicle accidents, however, for this work a novel annotation is used. Each video is newly annotated with the required attention level of the driver, *i.e.* the amount of attention required to drive in each situation. These videos were collectively annotated among a group of participants who were asked to view the video from the perspective of the driver and provide a label corresponding to what they perceived to be the driver's required attention level. The average score across all participants was taken as the final attention label. The four categories of attention are as follows: low attention, medium attention, high attention, and very high attention. The complete system runs in real-time; approximately ten frames per second.

Résumé

Le problème de cette recherche est d'estimer les niveaux d'attention perçus du conducteur vidéos de cam dash. Cette recherche examine une approche différente de l'attention du conducteur; le besoin au lieu de la performance du conducteur. Les données d'entrée sont constituées de vidéos de caméras de tableau de bord situations. Un double flux approché est utilisé où un flux spatial regarde le brut cadres individuels et porte des informations sur les caractéristiques d'apparence, tandis que l'autre flux regarde le flux optique entre les images et capture des informations de mouvement. L'ensemble modèle est construit sur deux réseaux neuronaux convolutifs (CNN) et un neuronal récurrent final Réseaux (RNN) qui combine les deux flux ensemble dans le temps. L'ensemble de données utilisé est un collection de 1750 vidéos de cam dash collectées par Chan et al. [1]. L'ensemble de données est annoté avec des exemples positifs et négatifs d'accidents. Au lieu d'utiliser ces accidents binaires catégories, chaque vidéo est nouvellement annotée avec un niveau d'attention perçu du conducteur, c.-à-d. la quantité d'attention requise par le conducteur dans chaque situation. Les vidéos sont collectivement annoté parmi un groupe de sujets qui ont été invités à utiliser toutes les informations dans le vidéo pour annoter les «meilleurs» niveaux d'attention perçus. Le score moyen pris comme étiquette d'attention finale. Les quatre catégories de niveaux d'attention requis sont les suivantes: faible attention, attention moyenne, grande attention et très grande attention. De ces attentions niveaux d'une architecture de réseau de neurones multiples multiples est utilisé pour estimer le conducteur requis attention dans diverses situations. Le système complet fonctionne en temps réel; approximativement dix images par seconde.

Acknowledgements

This work is supported by NSERC. Additionally, the author gratefully acknowledges Brisk Synergies for providing funding for this research.

The author would like to thank his advisor Dr. James Clark for all his motivation, guidance, and patience throughout this process.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Even for the most experienced drivers, driving any vehicle is a difficult task. This difficulty can be shown by the high number of collisions in Canada alone. In 2015, there were 118,404 collisions that were either fatal or involved a personal injury [2]. In about 84% of all accidents the cause was traced back to driver error [9].

Although the majority of vehicles are currently equipped with passive safety systems, *i.e.* systems to help reduce the outcome of an accident, such as seat belts, airbags, *etc.*, there are still a high number of serious accidents. Newer intelligent car models are becoming equipped with active safety systems that utilize an understanding of the vehicle's state to avoid and minimize the effects of a crash. Some of these systems include collision warning and adaptive cruise control. Research into these active safety systems have expanded into applications that work with or for the driver. This new generation of driver-assistance systems go beyond automated control systems by attempting to work in combination with a driver. These advanced safety systems include predicting driver intent [20], warning drivers of lane departures [16], *etc.* Although these active systems have many benefits, they are difficult to implement as they require knowledge about the driver, the vehicle, or the environment.

Furthermore, as these new technologies becoming embedded into users' everyday life,

drivers need to make sure they are paying adequate attention to their current driving environment. To this end, there has been numerous works on modeling and monitoring drivers' attentiveness. Many of these works attempt to directly correlate driver attention to other measurements such as drowsiness, head movement and position, or alertness [12, 27, 18, 19]. Many of these systems, often referred to as driver attention monitoring systems, work sufficiently well leading to newer vehicles becoming equipped with them. These vehicle safety systems were first introduced by Toyota in 2006 for its latest Lexus models [26]. Their system uses infrared sensors to monitor the driver's attention level. Specifically, these driver monitoring systems include a camera placed on the steering column which is capable of eye tracking via infrared LED detectors. In the case that the driver is not paying adequate attention to the road ahead, and a dangerous situation is detected, the system will warn the driver by flashing lights or providing warning sounds. If no action is taken by the driver, the vehicle will enter automation mode and apply the brakes (a warning alarm will sound followed by a brief automatic application of the braking system). After Toyota entered the market, various other companies began to follow their lead.

BMW started to offer an Active Driving Assistant with Attention model that analyses the user's driving behavior and, if necessary, advises the driver to rest. The alert system notifies the driver as to when to take a break in the form of graphic symbols shown on the control display [1]. Similarly, Bosch offers a driver drowsiness detection system that takes input from a combination of steering angle sensors, a front-mounted lane assisting camera, vehicle speed, and turn signal information [7]. Using this information, a driver drowsiness level is computed. Additionally, Ford, Hyundai, and Kia all come fully installed with their respective driver attention warning systems. These systems were first debuted by Ford with their 2011 Ford Focus [6], Hyundai on their 2017 i30, and lastly, Kia with their 2018 Stinger.

Although these systems can provide a broad understanding of the driver's attentiveness, they address the driver's performance as opposed to the driver's need. Additionally, many of these systems require a difficult installation and calibration process. To this end, the proposed research aims to gather knowledge related to the driver's required attention level

from a single outward-facing dashcam. To accomplish this, dashcam videos are processed by a two-stream recurrent convolutional architecture to produce a label corresponding to the driver's required attention level in each driving situation. The attention levels are divided into four categories: low attention, medium attention, high attention, and very high attention.

The proposed two-stream recurrent convolutional architecture is based on a two-stream approach that models both spatial and temporal features in separate convolutional neural network (CNN) streams. These streams are then combined and fed into a Recurrent Neural Network (RNN) that models the new stream of features as hidden units in time. Both streams are built upon a pre-trained VGG [24] network that is fine-tuned with the newly annotated attention dataset. The spatial stream takes still video frames as inputs and outputs a stream of high-level appearance features. The temporal stream utilizes simple flow fields extracted from pairs of adjacent frames and outputs high-level motion features. The final output features of both streams are then combined and fed into a RNN using Long Short-Term Memory (LSTM) units [10]. Since the proposed algorithm assigns a single attention level to the full video sequence, only the final hidden state of the RNN model is used. This last hidden state is then feed into a Softmax classifier to produce the final output of the network, *i.e.* the driver's required attention level. The complete process runs in real-time (ten frames per second) on a laptop GPU.

In particular, the main contributions of this paper are as follows: 1) create a novel attention dataset that addresses the driver's need as opposed to the driver's performance, 2) propose a two-stream recurrent convolutional architecture that estimates the driver's required attention level via outward-facing dashcam videos in real-time, and 3) help improve driver safety by proposing a system that can analyze the road ahead and provide intuitive feedback to the driver.

The rest of this paper is organized as follows. Section 2 presents related work on driver attention that employs various methods use as gaze following using convnets. The explanation

of the structure and training scheme for the two-stream recurrent convolutional architecture is presented in Section 3 and 4 respectively. Section 5 outlines the experiments and presents the prediction performance of the proposed model. Future works and concluding remarks are provided in Section 6.

1.1 Objectives and Scope

The main objectives of this research are:

- Determine a multiple neural network architecture that can learn to estimate a driver's required attention levels in various driving situations.
- Train the network on a newly labelled dataset containing annotations of the driver's perceived attention levels for each driving situation.
- Run the algorithm in real-time using only a single, outward facing dashcam as input.

1.2 Overview

This thesis consists of (insert number of chapters) chapters. A short summary of each chapter is provided in the following section. The chapter order reflects the manner in which this research program was evolved.

Chapter 2 describes the fundamental framework used to develop this research program. Among the topics presented are recent video classification algorithms, the extension into vehicles for vehicle intent, and lastly models for driver attention.

Chapter 3 describes...

Chapter 2

Literature Review

The following literature review will be separated into three sections. The first section describes a general approach to video classification, the second section addresses video classification with respect to advanced driver assistance systems, and the last section considers the existing research in the area of driver attention.

2.1 Video Classification

Video classification in real-world environments find applications in a variety of domains including video surveillance, human action recognition, and intelligent vehicles. Although substantial process has been made in this area, it is still a very challenging problem due to a number of reasons including cluttered backgrounds, occlusions, ambiguity, *etc.* To address these issues, most approaches make strict assumptions about the circumstances under which the video was taken. Additionally, a two-step paradigm is generally followed: (1) compute complex handcrafted features from raw video frames, and (2) learn a classifier based on these obtained features. The main disadvantages with these approaches are most assumptions do not hold true in real-world environments and it is very difficult to know which features are

important for a given task.

Deep learning architectures are a class of models that address these problems by learning specific hierarchy features for a given task, and thus, automating the process of feature construction. The follow sections will introduce recent deep learning approaches in video classification.

2.1.1 Three-Dimension Convolutional Neural Networks (CNNs)

One popular deep learning model for understanding image content are convolutional neural networks (CNNs). These models are capable of automatically learning complex features required for vision while yielding state-of-the-art results in image recognition, segmentation, detection, and retrieval. Encouraged by the results in the domain of images, CNN approaches have recently extended into video classification tasks. Compared to still images, video classification tasks require methods to capture additional motion information encoded in adjacent frames. The following section will describe a CNN model that incorporates both spatial and temporal features.

A simple approach to applying CNNs to video classification is to treat each individual frame as a still image and apply a CNN at the individual frame level. Since this approach is applied to two-dimensional images, it does not consider motion information encoded in multiple adjacent frames. To incorporate this information, the work proposed by Ji *et al.* [13] introduces a three-dimensional CNN model applied particularly to action recognition. This model performs three-dimensional convolutions in the convolutional layers of the CNN such that both spatial and temporal dimensions are captured. The full architecture is initialized by generating multiple channels of information from adjacent video frames, *e.g.* gradient and flow information. The model then performs convolution and pooling separately on each information channel. The final feature vector is obtained by combining information from all channels.

In two-dimensional CNNs, convolutions are performed at each convolutional layer to extract features from a local neighborhood on a set of feature maps from the previous layer. An additive bias is then applied and the result is passed through a nonlinear function. Formally, the unit at position (x, y) in the j^{th} feature map in the i^{th} layer, denoted as v_{ij}^{xy}, is given by:

$$v_{ij}^{xy} = tanh\left(b_{ij} + \sum_{m}\sum_{p=0}^{P_i-1}\sum_{q=0}^{Q_i-1} w_{ijm}^{pq} v_{(i-1)m}^{(x+p)(y+q)}\right) \tag{2.1}$$

where $tanh(.)$ is the hyperbolic tangent function, b_{ij} is the bias for the feature map, m indexes over the set of feature maps in the $(i-1)^{th}$ layer, w_{ijk}^{pq} is the value at the position (p, q) of the kernel connected to the k^{th} feature map, and P_i and Q_i are the respective height and width of the kernel [13]. The resolution of the feature maps is reduced in the subsampling layers by pooling over a local neighborhood of features. The functionality of these layers is to reduce the spatial size of the feature maps, the number of parameters, and the computation in the network, and thus, controlling overfitting. A CNN architecture is generally constructed by stacking multiple convolutional and subsampling layers in an alternating fashion.

As previously mentioned, these two-dimensional CNNs only compute features from the spatial dimensions. To capture motion information, three-dimensional convolutions are used in the convolutional stages of the CNNs. The 3D convolution is achieved by stacking multiple adjacent frames together. By this construction, the feature maps in the convolutional layer are connected to multiple contiguous frames in the previous layer, thereby capturing motion information. Formally, the value at position (x, y, z) on the j^{th} feature map in the i^{th} layer is given by:

$$v_{ij}^{xyz} = tanh\left(b_{ij} + \sum_{m}\sum_{p=0}^{P_i-1}\sum_{q=0}^{Q_i-1}\sum_{r=0}^{R_i-1} w_{ijm}^{pqr} v_{(i-1)m}^{(x+p)(y+q)(z+r)}\right) \tag{2.2}$$

where R_i is the size of the 3D kernel along the temporal dimension, w_{ijm}^{pqr} is the $(p, q, r)^{th}$ value of the kernel connected to the m^{th} feature map in the previous layer [13].

The complete 3D-CNN architecture considers several contiguous frames as input. The model is initialized by applying a set of hardwired kernels to the given input sequence to generate multiple channels of information. These five channels consist of the following infor-

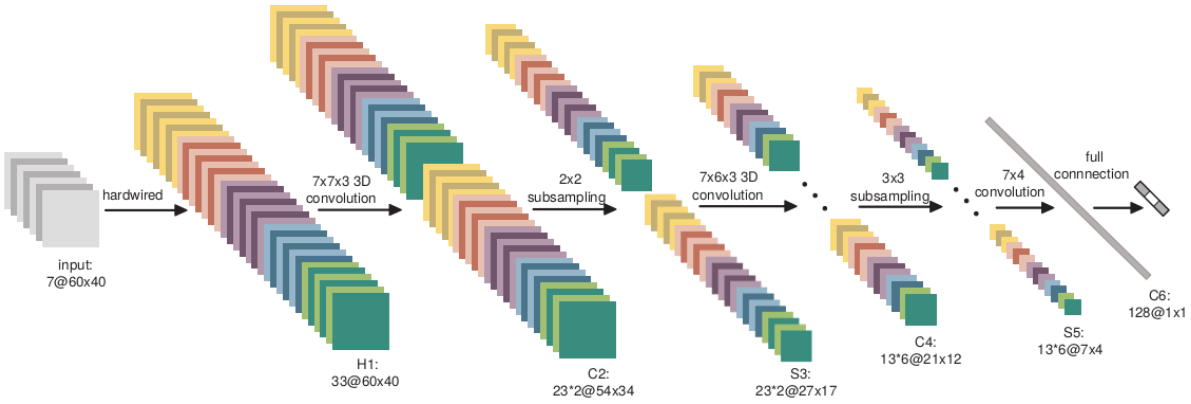

Figure 2.1: 3D CNN architecture. This architecture consists of one hardwired layer, three convolutional layers, two subsampling layers, and a fully connected layer [13].

mation: 1) grayscale information where each frame in the input sequence is first converted to a grayscale image, 2) gradient information in both vertical and horizontal directions, and similarly, 3) optical flow information in both vertical and horizontal directions. This hardwired layer is used to encode prior knowledge on the computed features.

Once the five channels of information are computed, the 3D-CNN architecture applies 3D convolutions to each channel respectively, where each 3D convolutional kernel size has an extra temporal dimension T. To increase the number of feature maps, two sets of distinct convolutions are applied at each location. These two channels are then passed through alternating pooling and convolutional layers until the last fully-connected layer. The last fully-connected layer incorporates all previous information captured throughout the network, and thus, successfully capturing both spatial and motion information. The full 3D-CNN architecture, along with the respective kernel and feature sizes, is shown in figure 2.1.

The complete architecture was evaluated on the TRECVID 2008 and KTH datasets [22]. The TRECVID dataset consists of 49-hours of video captured at the London Gatwick Airport using five different cameras. The architecture focused on the recognition of three action classes in a one-against-rest manner. Additionally, a simple frame-based 2D-CNN architecture was used in comparison. The average precision and recall for both the 3D and 2D-CNN

architectures are as follows: the 3D-CNN received an average precision and recall accuracy of 71.37% and 2.30% respectively, while the 2D-CNN architecture received accuracies of 60.85% and 1.55%. This comparison demonstrates the benefits of incorporating motion information in the convolutional layers of the CNN architecture. The KTH dataset consists of six action classes performed by 25 subjects. Using a similar evaluation process to the TRECVID dataset, the 3D-CNN model was compared to other common action recognition models [17, 11, 21]. The 3D-CNN architecture achieved an overall accuracy of 90.2%, demonstrating competitive results to the compared methods with accuracies of 83.3%, 91.7%, and 92.7% respectively.

2.1.2 Fusion-Based Convolutional Neural Networks

Unlike images which can be cropped and rescaled to a fixed size, videos vary widely in temporal extent and cannot be easily processed with a fixed-size architecture. The fusion-based CNN architecture proposed by Karpathy *et al.* [14] offers to treat every video as a bag of short, fixed-sized clips. Since each clip contains several contiguous frames in time, the connectivity of the CNN model is extended in the time domain in order to learn spatio-temporal features.

To learn these spatio-temporal features, three proposed fusion methods are evaluated: late fusion, early fusion, and slow fusion. One difficulty with extending CNN models in the time domain is the difficulty of training grows considerably, *i.e.* the network must now process several frames of the video at a time. To address this issue, the proposed architectures are modified to contain two separate streams of processing: 1) a context stream that learns features on low-resolution frames, and 2) a high-resolution fovea stream that only operates on the middle portion of the frame. The single-frame CNN model along with its extension in time via fusion is described below and demonstrated in figure 2.2.

- The single-frame model is similar to the ImageNet challenge winning model [15] as

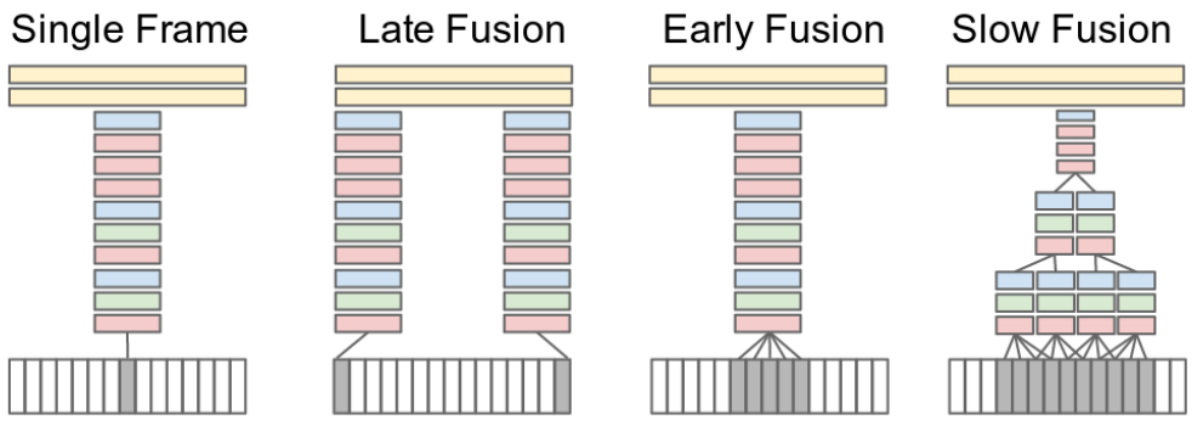

Figure 2.2: Fusing information over temporal dimensions. Red, green, and blue boxes indicate the convolutional, normalization, and pooling layers, respectively [14].

it looks at each frame individually and extracts only spatial information as it passes through the CNN model. The final fully-connected layers contain no motion information from the input frames.

- The late fusion model places two separate single-frame networks with shared parameters a distance of n frames apart. This model passes spatial information through the two separate single-frame networks and merges both streams in the first fully-connected layer. Using this approach, neither single-frame network alone can detect any motion but the first fully-connected layer can compute global motion characteristics by comparing the outputs of both towers.

- The early fusion model extends on the single-frame fusion model as it combines information across an entire window immediately on the pixel level. To incorporate this information, the filters in the first convolutional layer of the single-frame model are extended by T pixels, where T is some temporal extend. This new spatio-temporal information is then passed forward through the complete network.

- The slow fusion model is a balanced mix between the two approaches. This model slowly fuses temporal information throughout the network such that the higher layers receive progressively more global information in both spatial and temporal dimensions.

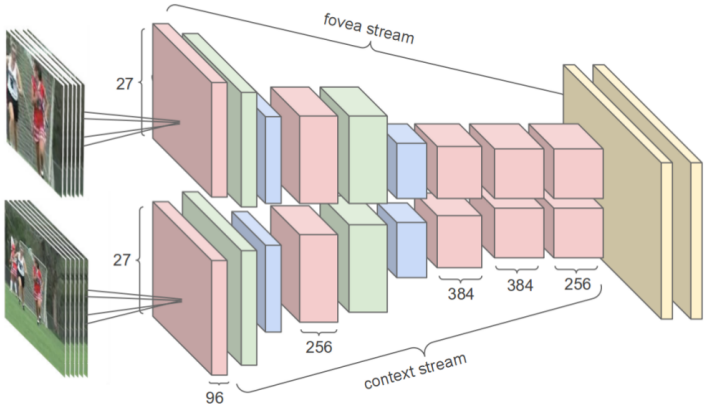

Figure 2.3: Multi-resolution CNN architecture. Both streams consist of convolution (red), normalization (green), and pooling (blue) layers. The streams converge via two fully-connected layers (yellow) [14].

> This model is implemented by extending the connectivity of all convolutional layers in time and carrying out temporal convolutions in addition to spatial convolutions to activations.

To train the fusion-based models takes considerably more time than training the single-frame approach, and thus, a multi-resolution architecture is used. This architecture aims to balance the speed of low-resolution training with the accuracy of high-resolution training. The low-resolution context stream receives the downsampled frames at half the original spatial resolution, while the fovea stream receives the center region at the original resolution. Using the two-stream approach, the total input dimensionality is reduced by half. The complete architecture processes both streams by the same networks described above, with the small adjustment of starting at half the original input resolution. The final activations from both streams are then concatenated and fed into the first fully connected layer of the network. Figure 2.3 demonstrates the final CNN architecture.

The proposed architecture was evaluated on the Sports-1M dataset consisting of one-million YouTube videos annotated with 487 classes [14]. Using this dataset, the hit-1 and hit-5 (correct prediction in top k predictions) accuracies were computed for both the single-frame approach and the early, late, and slow fusion approaches. Table 2.1 demonstrate these

results. It is evident that incorporating additional motion information via late and slow fusion provides additional insight into the video classification task.

Additional, transfer learning was conducted on the UCF-101 Activity Recognition dataset [25]. This dataset consists of 13,320 videos belonging to 101 categories. The same slow fusion network that was trained on the Sports-1M dataset was then evaluated on the UCF-101 dataset scoring a hit-1 accuracy of 65.4% when fine-tuning the top three full-connected layers.

Model	**Hit-1**	**Hit-5**
Single-Frame	59.3%	77.7%
Early Fusion	57.7%	76.8%
Late Fusion	59.3%	78.7%
Slow Fusion	60.9%	80.2%
CNN Average (Single+Early+Late+Slow)	63.9%	82.4%

Table 2.1: Accuracies for the correct prediction in the top k predictions on the Sports-1M dataset via fusion-based CNN architectures [14].

Two-Stream Convolutional Networks

The previous methods described above attempt to model both spatial and temporal features within the same network. The two-stream approach, proposed by [23], offers a two-stream convolutional neural network architecture that incorporates both spatial and temporal information in separate streams. The spatial stream, in the form of individual frame appearance, carries information about scenes and objects depicted in the video while the temporal stream, in the form of motion across the frames, conveys the movement of the observer (the camera) and the objects [23]. The two streams are then combined in the later layers via late fusion. The proposed architecture is related to the two-stream hypothesis in which the human visual cortex contains two pathways: 1) a ventral stream that performs objection recognition, and 2) a dorsal stream that recognizes motion [8]. Additionally, decoupling of the spatial and temporal networks allow the spatial network to improve its training time by pre-training on

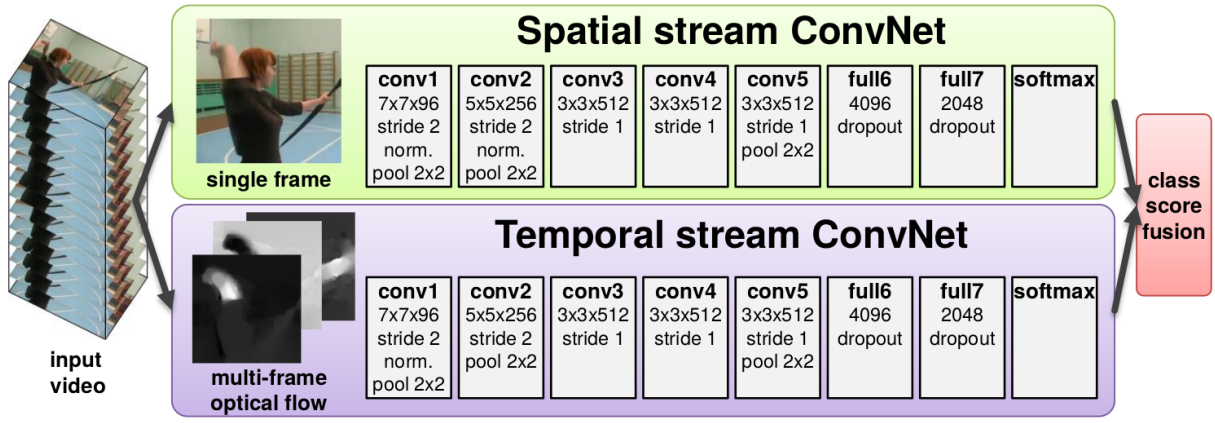

Figure 2.4: Two-stream architecture for video classification [23].

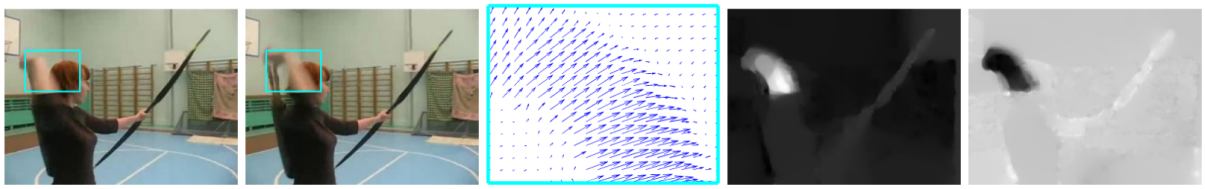

Figure 2.5: Optical Flow. (a)(b): a pair of consecutive frames with the area above the moving hand outlined. (c): a close-up of the dense optical flow in the outlined area. (d)(e): horizontal and vertical component of the displacement field, respectively. Higher intensity corresponds to positive values, while lower intensity to negative values [23].

large amounts of labelled object data.

Figure 2.4 demonstrates the complete two-stream architecture. As mentioned previously, the spatial ConvNet stream operates on each individual frame in the video, effectively performing action recognition from still images. These appearance features are useful to the complete architecture as some actions are strongly associated to a particular object. The spatial stream in this sense computes the same appearance features as an image classification architecture, and thus, the spatial stream can take advantage of pre-training on a large image classification dataset, *i.e.* the ImageNet dataset [15]. The temporal stream, which recognizes motion between consecutive frames, operates on optical flow fields. These flow fields explicitly describe the motion between consecutive frames and provides improvement

to the training phase as the network does not need to estimate motion implicitly from raw image frames.

The input to the temporal stream is formed by stacking optical flow displacement fields between several consecutive frames. The dense optical flow fields can be seen in figure 2.5. The dense optical flow can be represented as set of displacement vector fields d_t between a pair of consecutive frames at t and $t+1$. Using this notation, $d_t(u,v)$ denotes the displacement vector at point (u,v) in frame t that moves to its end location in frame $t+1$. The horizontal and vectors components of the vector field are d_t^x and d_t^y respectively. To represent motion across a sequence of frames, the flow channels of L consecutive frames are stacked, represented by $d_t^{x,y}$, to form a total of $2L$ input channels [23]. In general, it is beneficial to perform zero-centering of the network input, as it allows the model to better fit the non-linearities. In the case of optical flow, the displacement between a pair of given frames can be dominated by a particular direction, generally caused by camera movement. It is important to compensate for this camera motion, and various methods have attempted to estimate a global displacement vector and subtracting it. The approach used in this architecture is a simple mean subtraction; from each displacement field d, the mean vector (d_x, d_y) is subtracted. Once the final compensated inputs are feed into the complete architecture, the output of both CNN streams are a set of softmax scores for each class label. The last layer combines the two sets of softmax scores using a late fusion method based on a multi-class linear support vector machine (SVM) [4].

The complete architecture was evaluated on the UCF-101 dataset [25]. Individually, the spatial and temporal streams received a hit-1 accuracy of 72.7% and 81.2% respectively. Using both streams, the final hit-1 accuracy on the UCF-101 dataset was 88.0%.

2.1.3 Long-Term Recurrent Convolutional Neural Networks

Recurrent Neural Networks (RNNs) are capable of learning complex temporal dynamics by mapping a given input sequences to a sequence of hidden states, and hidden states to a sequence of outputs via the following recurrence equations:

$$h_t = g(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{2.3}$$

$$z_t = g(W_{hz}h_t + b_z) \tag{2.4}$$

where g is an element-wise non-linearity, such as a sigmoid or hyperbolic tangent, x_t is the input, $h_t \in \mathbb{R}^N$ is the hidden state with N hidden units, and y_t is the output at time t [5]. Although RNNs have proven successful in various fields, long-term dependencies are difficult to capture due to the well-known vanishing/exploding gradients problem [10]. This problem results from propagating the gradients down through the many layers of the recurrent network, each corresponding to particular timestep. Long short-term memory units (LSTMs) [10] provide a solution to this problem by incorporating memory units that allow the network to learn when to forget previous hidden states and when to update hidden states given new information. The equations for the LSTM are as follows:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \tag{2.5}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \tag{2.6}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \tag{2.7}$$

$$g_t = \phi(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{2.8}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{2.9}$$

$$h_t = o_t \odot \phi(c_t) \tag{2.10}$$

where $\sigma(x) = (1+e^{-x})^{-1}$ is the sigmoid non-linearity, $\phi(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ is the hyperbolic tangent non-linearity. In addition to the hidden unit $h_t \in \mathbb{R}^N$, the LSTM includes an input gate $i_t \in \mathbb{R}^N$, forget gate $f_t \in \mathbb{R}^N$, output gate $o_t \in \mathbb{R}^N$, input modulation gate $g_t \in \mathbb{R}^N$, and memory cell $c_t \in \mathbb{R}^N$ [5].

Recently, LSTMs have achieved impressive results in the domains of speech recognition,

machine translation, *etc.* Analogous the CNNs, LSTMs are attractive because they allow end-to-end fine-tuning. The advantages of LSTMs for modeling sequential data in vision problems are twofold. First, when integrated with current vision systems, LSTM models are straightforward to fine-tune end-to-end, and secondly, LSTMs are not confined to fixed length inputs or outputs, allowing simple modeling for sequential data of varying lengths, such as text or video. The following sections will demonstrate how LSTMs are used for video classification.

Contrast to the models described above which explore two extrema of perceptual time-series learning, *i.e.* either learning a fully-general time-varying weighting, or applying simple temporal pooling, the proposed Long-term Recurrent Convolutional Network (LRCN) [5] is "doubly deep" in that it can be compositional in both the spatial and temporal "layers". Such models have advantages when target concepts are complex and/or training data are limited. Following the same motivation for current deep convolutional models, these models are also deep over the temporal dimension, *i.e.* temporal recurrence of latent variables. The proposed recurrent long-term models are directly connected to modern visual convolutional networks and can be jointly trained to simultaneously learn temporal dynamics and convolutional perceptual representations.

The complete architecture is built on-top of a deep hierarchical visual feature extractor (CNN) with a model that can learn to recognize and synthesize temporal dynamics for tasks involving sequential data. The system works by passing in each visual input, *e.g.* video frame, represented as v_t, though a feature transformation $\phi_v(v_t)$ parameterized by V producing a fixed-length vector representation $\phi_t \in \mathbb{R}^d$. Having computed the feature-space representation of the visual input sequence $(\phi_1, \phi_2, \ldots, \phi_T)$, the sequence model takes over. The sequence model parameterized by W maps as input x_t and a previous timestep hidden state h_{t-1} to an output z_t and updated hidden state h_t. Inference must be ran sequentially, *i.e.* from top to bottom, by computing in order: $h_1 = f_W(x_1, h_0)$, then $h_2 = f_W(x_2, h_1)$, up to h_T. The final step is computing the prediction distribution $P(y_t)$ at time-step t. To do this, the softmax over outputs z_t of the sequential model, producing a distribution over the

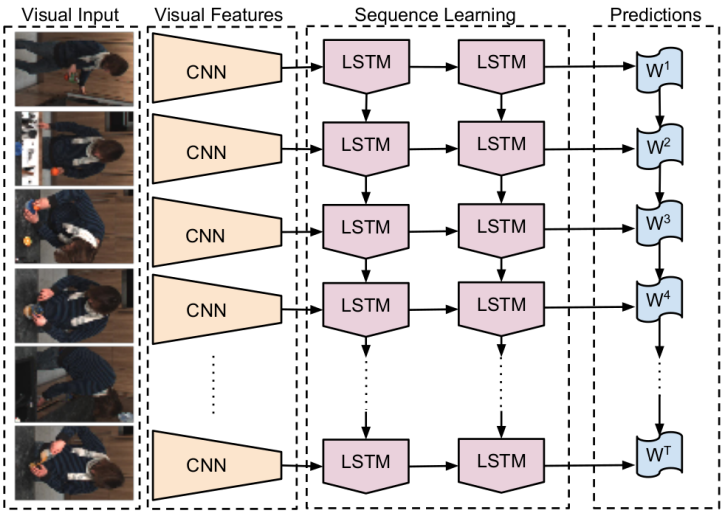

Figure 2.6: Proposed LRCN architecture. The LRCN processes the variable-length visual inputs (left) with a CNN (middle-left), whose outputs are fed into a stack of recurrent sequence models (LSTMs, middle-right), which finally produce a variable-length prediction (right) [5].

space of C possible per-timestep outputs:

$$P(y_t = c) = \frac{exp(W_{zc} z_{t,c} + b_c)}{\sum_{c' \in C} exp(W_{zc} z_{t,c'} + b_c)} \tag{2.11}$$

The visual feature transformation, ϕ, corresponds to activations in some layer of a deep CNN model. The complete model is shown in figure 2.6.

The final model takes as input T video frames $(x_1, x_2, \ldots, x_T)$ and outputs T labels $(y_1, y_2, \ldots, y_T)$. A fusion approached is used to merge per-timestep predictions $(y_1, y_2, \ldots, y_T)$ into a single prediction y for the full sequence. The T individual frames are inputs into T convolutional networks which are connected to a single-layer LSTM with 256 hidden units. The complete model considers both RGB and "flow-images". The "flow-images" are created by transforming the computed optical flow by centering the x and y values around 128 and multiplying by a scalar such that the flow values fall between 0 and 255. The third channel is created by computing the flow magnitude.

The architecture was evaluated on the UCF-101 dataset [25] which consists of 12'000 video categorized into 101 human action classes. A single-frame baseline architecture was used in comparison to the proposed LRCN network. Table 2.2 demonstrates the results.

The LRCN model consistently outperforms the baseline model and incorporating RGB and flow helps improve more. This also shows a substantial increase compared to the previous method.

	Single Input Type		**Weighted Average**	
	RGB	**Flow**	**1/2 1/2**	**1/3 2/3**
Single-Frame	67.70%	72.19%	75.87%	78.84%
LRCN	68.19%	77.46%	80.62%	82.66%

Table 2.2: Comparing single-frame models to LRCN networks for activity recognition on the UCF-101 dataset [25], with both RGB and flow inputs [5].

Beyond Short Snippets

2.2 Vehicle Video Classification

2.3 Driver Attention Levels

Chapter 3

Conclusion

This paper presented the problem of estimating the driver's required attention via outward-facing dashcam videos. The proposed multiple neural network architecture uses appearance and motion features and recurrent neural network. We achieve a real-time system that helps alert the driver when a high or very high attention level is required with an accuracy twice that of ¡¡insert accuracy¿¿. Our final experimental results demonstrate the effectiveness of using a simple outward facing dashcam. Moving forward, we would like to continue this research to explore how including other fast processing features could help improve the accuracy while still maintaining results in real-time.

Bibliography

[1] *BMW Model Upgrade Measures Taking Effect from the Summer of 2013*. https://www.press.bmwgroup.com/global/article/detail/T0141144EN/bmw-model-upgrade-measures-taking-effect-from-the/summer-of-2013. Accessed: 2018-02-08. 2013.

[2] *Canadian Motor Vehicle Traffic Collision Statistics: 2015*. https://www.tc.gc.ca/eng/motorvehiclesafety/tp-tp3322-2015-1487.html. Accessed: 2018-02-08. 2017.

[3] Fu-Hsiang Chan et al. "Anticipating Accidents in Dashcam Videos". In: *Computer Vision – ACCV 2016*. Ed. by Shang-Hong Lai et al. Cham: Springer International Publishing, 2017, pp. 136–153. ISBN: 978-3-319-54190-7.

[4] Koby Crammer and Yoram Singer. "On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines". In: *J. Mach. Learn. Res.* 2 (Mar. 2002), pp. 265–292. ISSN: 1532-4435. URL: http://dl.acm.org/citation.cfm?id=944790.944813.

[5] J. Donahue et al. "Long-Term Recurrent Convolutional Networks for Visual Recognition and Description". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.4 (Apr. 2017), pp. 677–691. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2016.2599174.

[6] *Driver Drowsiness*. https://web.archive.org/web/20110513232258. Accessed: 2018-02-08. 2011.

[7] Robert Bosch GmbH. *Driver Drowsiness*. http://www.bosch-mobility-solutions.com/en/. Accessed: 2018-02-08. 2013.

[8] Melvyn Goodale and A.David Milner. "Separate Visual Pathways for Perception and Action". In: 15 (Feb. 1992), pp. 20–5.

[9] Kerstin Haring, Marco Ragni, and Lars Konieczny. "A Cognitive Model of Drivers Attention". In: *Proceedings of the 11th International Conference on Cognitive Modeling, ICCM 2012*. Jan. 2012.

[10] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Comput.* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[11] H. Jhuang et al. "A Biologically Inspired System for Action Recognition". In: *2007 IEEE 11th International Conference on Computer Vision*. Sept. 2007, pp. 1–8. DOI: 10.1109/ICCV.2007.4408988.

[12] Qiang Ji and Xiaojie Yang. "Real Time Visual Cues Extraction for Monitoring Driver Vigilance". In: *Computer Vision Systems*. Ed. by Bernt Schiele and Gerhard Sagerer. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 107–124. ISBN: 978-3-540-48222-2.

[13] S. Ji et al. "3D Convolutional Neural Networks for Human Action Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.1 (Jan. 2013), pp. 221–231. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2012.59.

[14] Andrej Karpathy et al. "Large-Scale Video Classification with Convolutional Neural Networks". In: *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*. CVPR '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 1725–1732. ISBN: 978-1-4799-5118-5. DOI: 10.1109/CVPR.2014.223. URL: http://dx.doi.org/10.1109/CVPR.2014.223.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'12. Lake Tahoe, Nevada: Curran Associates Inc., 2012, pp. 1097–1105. URL: http://dl.acm.org/citation.cfm?id=2999134.2999257.

[16] Woong Kwon and Sukhan Lee. "Performance evaluation of decision making strategies for an embedded lane departure warning system". In: *Journal of Robotic Systems* 19.10 (2002), pp. 499–509. ISSN: 1097-4563. DOI: 10.1002/rob.10056. URL: http://dx.doi.org/10.1002/rob.10056.

[17] Juan Carlos Niebles, Hongcheng Wang, and Li Fei-Fei. "Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words". In: *International Journal of Computer Vision* 79.3 (Sept. 2008), pp. 299–318. ISSN: 1573-1405. DOI: 10.1007/s11263-007-0122-4. URL: https://doi.org/10.1007/s11263-007-0122-4.

[18] R. Onken. "DAISY, an adaptive, knowledge-based driver monitoring and warning system". In: *Intelligent Vehicles '94 Symposium, Proceedings of the*. Oct. 1994, pp. 544–549. DOI: 10.1109/IVS.1994.639576.

[19] Arun Sahayadhas and Kenneth Sundaraj. "Detecting Driver Drowsiness Based on Sensors: A Review". In: *Sensors* 12.12 (2012), pp. 16937–16953. ISSN: 1424-8220. DOI: 10.3390/s121216937. URL: http://www.mdpi.com/1424-8220/12/12/16937.

[20] Dario D. Salvucci. "Inferring Driver Intent: A Case Study in Lane-Change Detection". In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 48.19 (2004), pp. 2228–2231. DOI: 10.1177/154193120404801905. eprint: https://doi.org/10.1177/154193120404801905. URL: https://doi.org/10.1177/154193120404801905.

[21] K. Schindler and L. van Gool. "Action snippets: How many frames does human action recognition require?" In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*. June 2008, pp. 1–8. DOI: 10.1109/CVPR.2008.4587730.

[22] C. Schuldt, I. Laptev, and B. Caputo. "Recognizing human actions: a local SVM approach". In: *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* Vol. 3. Aug. 2004, 32–36 Vol.3. DOI: 10.1109/ICPR.2004.1334462.

[23] Karen Simonyan and Andrew Zisserman. "Two-Stream Convolutional Networks for Action Recognition in Videos". In: *CoRR* abs/1406.2199 (2014). arXiv: 1406.2199. URL: http://arxiv.org/abs/1406.2199.

[24] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *CoRR* abs/1409.1556 (2014). arXiv: 1409.1556. URL: http://arxiv.org/abs/1409.1556.

[25] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. "UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild". In: *CoRR* abs/1212.0402 (2012).

[26] *Toyota Enhances Pre-crash Safety System With Driver-monitoring Function*. https://newsroom.toyota.co.jp/en/detail/248128. Accessed: 2018-02-08. 2015.

[27] Jian-Da Wu and Tuo-Rung Chen. "Development of a drowsiness warning system based on the fuzzy logic images analysis". In: *Expert Systems with Applications* 34.2 (2008), pp. 1556–1561. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2007.01.019. URL: http://www.sciencedirect.com/science/article/pii/S0957417407000401.