Future Event Prediction: If and When

Lukáš Neumann Andrew Zisserman
Andrea Vedaldi
Visual Geometry Group, Department of Engineering Science
University of Oxford
{lukas,az,vedaldi}@robots.ox.ac.uk

Abstract

We consider the problem of future event prediction in video: if *and* when *a future event will occur. To this end, we propose a number of representations and loss functions tailored to this problem. These include several probabilistic formulations that also model the uncertainty of the prediction. We train and evaluate the approach on two entirely different prediction scenarios: if and when a car will stop in the BDD100k car driving dataset; and if and when a player is going to shoot a basketball towards the basket in the NCAA basketball dataset.*

We show that (i) we are able to predict events far in the future, up to 10 seconds before they occur; and (ii) using attention, we can determine which areas of the image sequence are responsible for these predictions, and find that they are meaningful, e.g. traffic lights are picked out for predicting when a vehicle will stop.

1. Introduction

Image understanding is usually concerned with the problem of describing the content of a given image or video. However, in applications such as robotics and autonomous driving this is often not enough: in order to react in a timely manner to external events (such as a pedestrian crossing the street), it may be necessary to predict them *before* they occur or are captured by the imaging device.

Our objective in this paper introducing the problem of time-to-event prediction into computer vision by predicting *future events* in video before they occur. In addition to its practical importance, the problem also has a significant scientific interest. In fact, predicting effectively the future is likely to require an understanding of subtle properties of the world and of its dynamics. Thus, this task can be used as a form of self-supervision to learn about abstract concepts in images and videos.

In this paper, we focus on two key challenges that are of direct interest in many applications: telling (i) *if* a certain event, such as a car stopping, is likely to occur in the near future and, in this case, (ii) *when* the event is going to happen. Our approach is based on performing probabilistic prediction of future events, while accounting for event rarity, which is required due to the fact that most of the data does not in fact contain the event of interest.

Seeking to develop a very general approach, we consider two entirely different testing scenarios. The first scenario is to predict, given a video stream captured from a moving car, whether and when the car is going to stop, in response to traffic conditions and other environmental factors. The second scenario is to predict, in videos of basketball games, whether and if a player is going to throw. We develop and publish two benchmarks for these tasks building on existing public benchmark data. We also develop a new benchmarking protocol based on evaluation metrics that reflect the ability of a model to perform such predictions in a manner that is relevant to applications.

The key design decision for this task is how to represent the prediction, and the associated loss function. In section 3 we discuss a number of possible options and discuss their advantages and disadvantages, introducing models that were not tested before in this context. Our best model is GMMH, a hybrid between a heatmap and a Mixture of Gaussians. We show that this approach works better than may be thought of as the "default" solution for modelling such problems, such as pure classification or the Weibull distribution used in survival analysis.

Compared to alternatives, the main benefit of our approach is its ability to predict events far in the future, up to 10 seconds before they occur. We also show that, as the algorithm learns to predict future events, it induces a visual representation that captures small but important details of image content. For example, in the driving application, we show that the learned neural network pays attention to elements such as traffic and car lights as predictor of slowing traffic.

We also compare our method to human performance, demonstrating that such systems are competitive and in fact

better at performing predictions about the future than average humans. We impute this to the ability of algorithms to better learn a specific domain, such as traffic conditions, and thus be better able to capture subtle cues and tells that may be overlooked by people.

2. Related Work

Early Action Recognition. Several authors have studied the problem of early action recognition, where the task is to predict *what* kind of action is currently happening, using as small number of frames as possible. Hoai et al. [9] use Structured Output SVM to detect facial expressions early. Aliakbarian et al. [3, 2] use context- and action-aware features with a two-stage LSTM to recognize actions from partial video sequences. Similarly, Sigurdsson et al. [18] propose a fully-connected temporal CRF model, which exploits intent information to predict in each frame the action being performed. Dave et al. [5] use a predictive-corrective model to maintain a memory state of the network for per-frame action classification. Ramanathan et al.[16] use a BLSTM to classify and detect events in individual frames of basketball sports videos and Heidarivincheh et al. [8] detect action completion. Wei et al. [27] on the other hand learn to classify the arrow of time in videos.

The crucial difference of all the above methods to our task is that the **event had already started**, i.e. the methods already observe the event of interest unrolling, whereas in our task we observe frames leading towards the start of the event, but never the event itself. The same difference applies to the standard activity recognition datasets [17, 10, 19, 4], which capture many different classes of events, but typically not what leads to them - and even if it was captured, the data would not be very informative for event prediction, as given the nature of the actions in the datasets such as "singing" or "cliff diving", the motivation leading to start the action is mostly intrinsic to the actors and therefore cannot be easily observed in the videos until the action actually starts.

Future Frames Prediction. A lot of work has been done trying to predict future visual appearance. Vondrick et al. [25, 26] use Generative Adversarial Networks (GANs) to explicitly model foreground and background pixels and generate a short video sequence from a single image. Xue et al. [29] also predict future frames from a single image, exploiting cross convolutional networks. Liang et al. [12] then use dual motion GAN to predict future frames given an input video sequence. The prediction horizon in all these methods is however only couple of frames, and therefore it is not applicable for predicting events which are several seconds in the future.

Event Prediction In the machine learning literature, time to event prediction has been extensively studied. Most recently, Martinsson [13] proposed a Recurrent Neural Network model with Weibull distribution for a customer churn prediction. Soleimani et al. [20] use Gaussian processes to predict time to event in medical applications. These models however deal with only low-dimensional sequential data (such as patient temperature measurements), and thus are not directly applicable in computer vision. Where possible, we however try to adapt these models for high-dimensional video data and evaluate them (see section 4.1).

In computer vision, Vondrick et al. [24] exploit AlexNet [11] representations to predict what action will happen in a recording of TV series in exactly 1 second time. Felsen et al. [6] predict which player will hold the ball next, by applying random forests to normalized overhead-view sport videos. Alahi et al. [1] use a LSTM model to predict a 2D heatmap of future pedestrian position in overhead pedestrian videos.

Our work is significantly different in that i) we predict whether the event will happen or not ii) we give time estimate to say when the event is likely to happen and with what probability iii) we predict significantly longer time horizon (up to 10 seconds) and iv) we don't require static and normalized camera views like [1, 6].

3. Method

3.1. Time to event

Consider a short sequence of N past and current visual observations $\mathbf{x}_{t_0-N+1}, \mathbf{x}_{t_0-N+2}, \ldots, \mathbf{x}_{t_0}$. Our aim is to estimate, based on this information, the probability that a certain event of interest will occur in the near future and, if so, when exactly (see fig. 2).

Since the choice of a time origin is immaterial, we simplify the notation by assuming $t_0 = 0$, and denote $X_N = (\mathbf{x}_{-N+1}, \mathbf{x}_{-N+2}, \ldots, \mathbf{x}_0)$ the N observations. Hence, the task can be formulated as estimating the conditional probability density $p(t|X_N)$ of the *time to event* (TTE) $\tau \geq 0$.

More precisely, we define TTE as the time when the observed system *enters first a certain state of interest*. For example, in the car application the state of interest is that the car velocity is zero, and in the sport application the condition is that the ball is flying as a consequence of a throw. This definition has a few important consequences. First, the system may enter and leave the state of interest several times and we are only concerned with the *first* of such occurrences. Second, the system may *already* be in said state at time 0, for example because the car is already stopped, so that strictly speaking τ may be less than 0. In this case we assume instead that $\tau = 0$. Thirdly, the event may not occur at all in the near future. The "no show" condition could be model by allowing t to range in $\mathbb{R}_0^+ \cup \{+\infty\}$. Instead, it is more practical to choose a fixed *prediction horizon* $\Delta_{\text{max}} \gg 0$ and conventionally set $\tau = 2\Delta_{\text{max}}$ whenever the TTE is beyond the horizon.

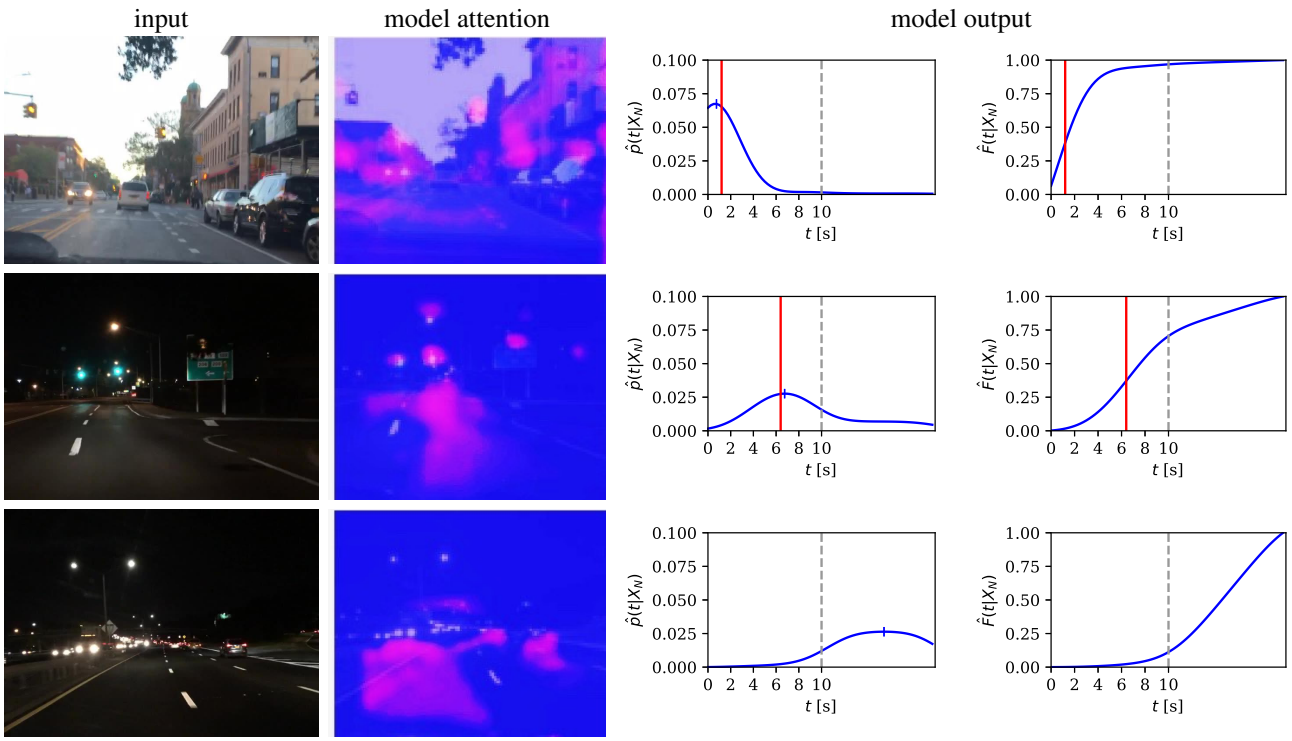

Figure 1. Predicting car stopping in the BDD100k driving dataset. Time to event probability prediction (blue), event occurrence ground truth (red), maximal prediction horizon Δ_{max} (dashed gray). The network has learned to look for various cues (traffic lights, cars in front on the road) to predict the probability if and when the car will stop. It has also learned to assign non-zero probability to time/position corresponding to green traffic lights, as they in fact might turn red by the time the car gets there (middle row). See Supplementary material for the videos.

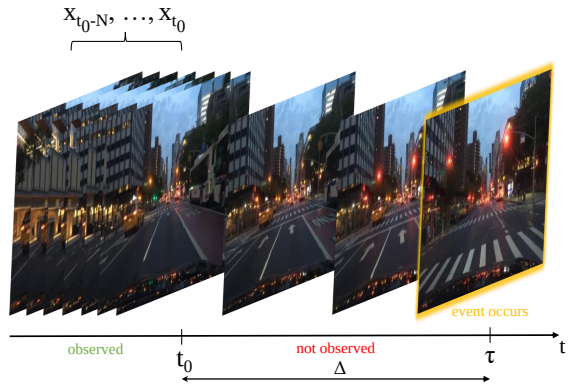

Figure 2. Predicting future event at time τ, having observed N frames up until time t_0, where Δ denotes the time distance between the last observed frame and the event

In the following, rather than discussing density function $p(t|X_N)$ directly, we will make use of their cumulative distribution function (CDF), which has the usual definition $F(u|X_N) = P[t > u|X_N]$. CDFs are more convenient in our discussion as they can represent both continuous and discrete-time distributions. Furthermore, CDFs can capture both non-trivial distributions, used to encode uncertainty, as well as "deterministic ones" that put all the mass on a specific value of t, and can thus be used to encode point estimates (these CDFs are step functions).

3.2. Predicting the TTE

Next, we describe a number of prediction models for the TTE, all implemented as neural networks Φ. These networks take as input a sequence of observations X_N and output an estimate $\hat{F}(t|X_N)$ of the TTE CDF. The networks are learned from example pairs (X_N, τ) via optimization of a suitable expected loss, which depends on the nature of the model. Models differs mainly by whether they use a discrete or continuous representation of time and by whether they predict a non-trivial distribution over possible TTEs or they produce instead point estimates.

3.2.1 Discrete-time models

One-in-many classification. The simplest approach to modelling the CDF $\hat{F}(t|X_N)$ is to quantize time in discrete bins and reduce the problem to a classification one. For simplicity, assume that the quantization interval is of one second and that there are Δ_{max} bins in total. Time $t \geq 0$ is mapped to a discrete index by the following quantizer function:

$$q(t) = \begin{cases} \lfloor t \rfloor + 1, & t < \Delta_{\text{max}}, \\ \Delta_{\text{max}}, & t \geq \Delta_{\text{max}}. \end{cases}$$

Index $q = \Delta_{\text{max}}$ means that the event occurs beyond the prediction horizon.

In order to implement this model, the network $\Phi(X_N)$ is terminated in a softmax layer configured to output a Δ_{max}-dimensional probability vector. The corresponding CDF is given by the cumulative sum

$$\hat{F}(t|X_N) = \sum_{i=1}^{q(t)} \Phi_i(X_N). \quad (1)$$

The model is trained by minimizing its negative log-likelihood $E_{(X_N,t)}[-\log \Phi_{q(t)}(X_N)]$.

Binary classifiers. As a variant of the previous model, we consider using Δ_{max} independent *binary classifiers* $\Phi_i(X_N) \in [0, 1]$ by processing the network output not via a softmax layer, but via a sigmoid. Each such classifier $\Phi_i(X_N)$ is trained via log-likelihood maximization to predict the probability that the event occurs at time i or later. At test time, the event is deemed to occur at the time the first of such classifier fires; formally, the CDF is given by

$$\hat{F}(t|X_N) = \begin{cases} 1, & \exists i \leq t+1 : \ \Phi_i(X_N) \geq \frac{1}{2}, \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

Note that the expression above is valid for values of t strictly less than the horizon Δ_{max}; if all classifiers fail to fire, then the event is deemed to occur beyond the prediction horizon and conventionally predicted to be at time $2\Delta_{\text{max}}$ as explained before.

Heuristic heatmaps. Inspired by the keypoint detection literature [23, 22, 14], we explore predicting the TTE via an heuristic heatmap. To this end, the neural network is configured to output a 1D heat map $\Phi(X_N) = \mathbf{h} \in \mathbb{R}_+^T$ of size $T = 2r\Delta_{\text{max}}$, where r is the temporal resolution (in our experiments, we set $r = 4$). The TTE is decoded as the position of the maximum in the heat map $\mathbf{h}$. For $t < 2\Delta_{\text{max}}$, the TTE CDF is obtained via cumulative summation and normalization:

$$\hat{F}(t; \mathbf{h}) = \frac{1}{\sum_{i=1}^{T} h_i} \sum_{i=1}^{\lfloor \frac{T}{2r\Delta_{\text{max}}} t \rfloor + 1} h_i. \quad (3)$$

This model is trained by minimizing the expected squared L^2 distance $E_{(X_N,t)}[\|\mathbf{g}_t - \Phi(X_N)\|^2]$ between the estimated heatmap and a heatmap prototype $\mathbf{g}_t$ centered at the ground-truth TTE value τ. The prototype is a Gaussian-like kernel $[\mathbf{g}_t]_i = \exp\left[-\frac{1}{2\sigma^2}\left(\frac{2r\Delta_{\text{max}}}{T} i - \tau\right)^2\right]$.

Note that this model allows the heatmap to have non-zero values in a region *beyond* the prediction horizon Δ_{max} up to $2\Delta_{\text{max}}$ so that the "no show" case can be captured. For training the model, "no show" samples are mapped to Gaussian window prototypes $\mathbf{g}_t$ whose center is randomly selected in the interval $[\Delta_{\text{max}}, 2\Delta_{\text{max}}]$. This ensures that data are more balanced in the training process, compared to a representation when one outputs just zero values in the heat map if the event did not occur.

3.2.2 Continuous-time models

A limitation of the previous models is their finite temporal resolution, due to the use of a quantizer function, and no representation of uncertainty of the prediction. We thus evaluate models that produce a continuous estimate of the TTE, and that, apart from Direct Regression, also output prediction uncertainty.

Direct regression. The simplest approach to predict the TTE over a continuous domain is to configure the neural network to output a real number $\Phi(X_N) = \hat{t} \in \mathbb{R}_+$ that approximates the TTE directly (following the convention that, if the event does not occur before the time horizon Δ_{max}, then the model outputs the value $2\Delta_{\text{max}}$).

This model estimates the TTE with infinite resolution, but it does not produce an uncertainty. This fact is captured by a step-wise CDF $\hat{F}(u|\hat{t}) = [u \geq \hat{t}]$. The model is trained by minimizing the expected absolute distance between estimated TTE $\hat{t}$ and ground-truth TTE τ, given by $E_{(X_N,t)}[|\tau - \Phi(X_N)|]$; this is the same as the expected L^1 distance $\|F(\cdot|\hat{t}) - F(\cdot|t)\|_1$ between the estimated step-wise CDF and the step-wise CDF centered at ground-truth TTE.

Gaussian distribution. In this case, the model $\Phi(X_N) = (\mu, \sigma) \in \mathbb{R}_+^2$ outputs two real numbers, the mean μ and standard deviation σ of a 1D Gaussian distribution $\mathcal{N}(t|\mu, \sigma)$, with CDF $F(t|X_N) = \mathcal{N}_{\text{CDF}}(t|\Phi(X_N))$. Again, we use the convention of setting $\tau = 2\Delta_{\text{max}}$ when the event does not occur within the prediction horizon. This model is also trained by minimizing the negative log-likelihood $E_{(X_N,\tau)}[-\log \mathcal{N}(\tau|\Phi(X_N))]$.

Weibull distribution. The Gaussian distribution is not necessarily optimal for modeling TTEs; in fact, several authors have proposed to use the Weibull distribution in similar contexts [13]. In order to experiment with this idea, the network is modified to output the parameters $(\alpha, \beta) \in \mathbb{R}^2$ of a Weibull distribution, whose CDF is given by $\hat{F}(t|\alpha, \beta) = 1 - e^{-(t/\alpha)^\beta}$. The advantage of the Weibull distribution is that it can more explicitly model the case in which the event does not occur within the prediction horizon Δ_{max}. Such a "no show" event is modeled as Type I censoring, denoted with $z = 0$, whereas $z = 1$ means that the event occurs within the window. The model is learned by minimizing the negative log-likelihood $E_{(X_N,t)}[-\log p(\tau|z, \Phi(X_N))]$ where

$$\log p(t|z, \alpha, \beta) = z\left(\log(\beta) + \beta \log \frac{t}{\alpha}\right) - \left(\frac{t}{\alpha}\right)^\beta \quad (4)$$

where, given our convention of mapping events that do not occur we have $z = [\tau > \Delta_{\text{max}}]$.

Gaussian Mixture Model Heatmap (GMMH). Finally, we propose a novel representation based on a Gaussian Mix-

ture Model (GMM), which combines the benefits of the Gaussian distribution and the heuristic heat map models.

The network is configured to output three vectors $\Phi(X_N) = (\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathbf{h}) \in \mathbb{R}^{T\times 3}$ again of dimension $T = 2r\Delta_{\text{max}}$. The first two vectors $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ represent the parameters of T 1D Gaussian distributions, and the third vector $\mathbf{h}$, similar to the heuristic heatmap, is used here as the weighting of the T Gaussians components. The CDF of this model is simply the weighted combination of Gaussian CDFs:

$$\hat{F}(t; \boldsymbol{\mu}, \boldsymbol{\sigma}, \mathbf{h}) = \frac{1}{\langle \mathbf{1}, \mathbf{h} \rangle} \sum_{j=1}^{T} h_j \mathcal{N}_{\text{CDF}}(t; \mu_j, \sigma_j).$$

The loss function is the negative log-likelihood of the GMM regularized by the loss already adopted for the heuristic heatmap:

$$E_{(X_N,t)} \left[-\log \sum_{j=1}^{T} \frac{h_j \mathcal{N}(\tau; \mu_j, \sigma_j)}{\langle \mathbf{1}, \mathbf{h} \rangle} + \lambda \|\mathbf{g}_t - \mathbf{h}\|^2 \right]. \tag{5}$$

3.3. Evaluation Metrics

Next, we present three evaluation metrics of increasing granularity, allowing to compare models' performance in terms of how well they predict whether the event will or will not occur (*Event Prediction Accuracy*), how accurately they predict time to the event (*Time-to-Event Error*) and how accurate is the event probability distribution estimate (*Model Surprise*).

We expect predictions to be more difficult as the event occurs farther in the future, so we break down evaluations based on this parameter. Specifically, we vary $\Delta_{\text{gt}} \in [0, \Delta_{\text{max}}]$ to draw performance curves, as the same test events are observed from greater temporal distance. We also report the average value of the three metrics as Δ_{gt} is swept inside this range.

Event Prediction Accuracy (EPA). EPA measures whether the model can successfully answer the *if* question, namely whether the event will or will not occur within the prediction window. This is a classification problem and EPA is the average classification accuracy. Recall that, in order to predict that the event does not occur, discrete-time models predict a special index/class, whereas continuous-time models predict the event to occur at a time $t > \Delta_{\text{max}}$.

Since most of the episodes X_N do not contain the event of interest (as these are comparatively rare), in order to make metrics comparable between event types and datasets we balance the testing set so that the ratio between sequences with and without the event is 50:50.

Time-to-Event Error (TTEE). TTEE measures whether the model can successfully answer the *if so, when?* question, namely determine when exactly the events will occur. TTEE is the average absolute prediction error $E_{(X_N,t)}[|\tau - \Psi \circ \Phi(X_N)|]$, where Ψ is the operation that maps the output of the neural network Φ to a point estimate $\hat{t} = \Psi \circ \Phi(X_N)$ for the TTE. For discrete-time models, for example, Ψ maps the predicted time index to a continuous time value to allow for a comparison against the ground-truth time. The empirical average is carried over the subset of the test set where the event does occur. If the network predicts incorrectly that the event does not occur, then the TTEE for that sample is set to Δ_{max}.

Model Surprise (MS). For model that outputs a prediction uncertainty in addition to a point estimates, we are also interested in measuring the quality of the predicted uncertainty value. We do so by taking the output distribution $\hat{p}(t|X_N)$ and measuring the expected negative log-likelihood given the ground truth annotations in the test set, defined as $E_{(X_N,\tau)}[-\log \hat{p}(\tau|X_N)]$. This is also known as model "surprise" and is an indication of the quality of the probabilistic output of the model: if the model assigns high probability values to the correct ground truth locations, the resulting "surprise" will be low, and vice-versa.

3.4. Backbone Architecture

In order to implement the neural network Φ, in all experiments we adapt the 3D ResNet-34 [7] architecture (see table 1) and extend it with a soft-attention module [28] to visualize what regions of the video sequence play the key role in the network decision making process. Depending on the model, we also change the output dimension accordingly (see Section 3.2). Most notably, our proposed GMMH has an output dimension of 80×3 ($T = 2r\Delta_{\text{max}} = 2 \times 4 \times 10 = 80$).

We used the vanilla SGD optimizer with Nesterov momentum [21] with an initial learning rate of 10^{-1}, which was decreased by a factor of 10 every time the loss stopped improving, and trained every model until the learning rate fell below 10^{-5}. All models were implemented in PyTorch [15] and all the source code will be released to foster reproducibility.

4. Experiments

We assess our approach in two challenging scenarios: egocentric car stopping and basketball throws.

4.1. Egocentric Car Stopping

In the first experiment, we aim to predict if and when a car is about to stop, using the video stream from a forward-looking camera mounted behind a windshield.

Dataset. We build on the BDD100k dataset [30], which consists of 100,000 video sequences each 40 seconds of length, accompanied with basic sensory data such as GPS, velocity or acceleration. We define the *stopping event* as the

Layer	Spatial Dim.	Time Dim.	Channels	Operation
Input	112×112	16	64	$3 \times 3 \times 3$ conv (stride 2), BN, ReLU
Pooling	56×56	8	64	$3 \times 3 \times 3$ max pool (stride 2)
Residual Block (1)	24×24	8	64	$3 \times 3 \times 3$ conv, BN, ReLU, $3 \times 3 \times 3$ conv, BN, sum
Residual Block (2)	24×24	8	128	$3 \times 3 \times 3$ conv, BN, ReLU, $3 \times 3 \times 3$ conv, BN, sum
Attention Map	24×24	8	1	$3 \times 3 \times 3$ conv, BN, ReLU, $3 \times 3 \times 3$ conv, softmax
Soft Attention	24×24	8	128	element-wise multiply all channels by *Attention Map*
Residual Block (3)	14×14	4	256	$3 \times 3 \times 3$ conv (stride 2), BN, ReLU, $3 \times 3 \times 3$ conv, BN, sum
Residual Block (4)	7×7	2	512	$3 \times 3 \times 3$ conv (stride 2), BN, ReLU, $3 \times 3 \times 3$ conv, BN, sum
Average Pooling	1×1	1	512	$7 \times 7 \times 2$ avg. pool
Output	80×3	—	—	fully connected

Table 1. 3D ResNet architecture

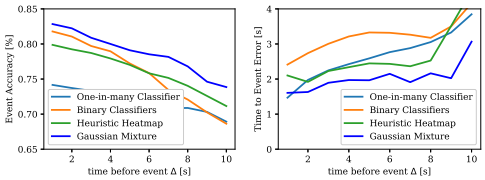

Figure 3. Predicting stopping in the BDD100k driving dataset. EPA (left) and TTEE (right) as functions of the temporal distance Δ before the event

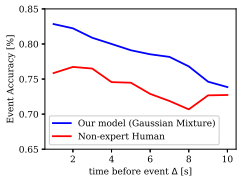

Figure 4. Human accuracy on BDD100k.

# frames	EPA [%]	TTEE [s]	MS
1	60.96	3.84	4.29
2	68.70	3.77	4.19
4	71.41	3.43	4.10
8	73.27	2.90	3.99
16	78.70	2.04	2.69
24	78.75	2.03	2.65

Table 2. Number of input frames vs. event prediction accuracy on BDD100k

Model	EPA [%]	TTEE [s]	MS
One-in-many Classifier	71.87	2.66	N/A
Binary Classifiers	75.90	3.21	N/A
Direct Regression	71.11	3.81	N/A
Gaussian Distribution	66.77	5.77	10.91
Weibull Distribution	55.66	8.31	3.20
Heuristic Heatmap	76.15	2.64	3.88
Gaussian Mixture	**78.70**	**2.04**	**2.69**

Table 3. Car stopping prediction in the BDD100k driving dataset

point in time when the vehicle velocity falls below 5km/h, if previously the velocity was above the threshold for at least 10 seconds (i.e. the vehicle was in motion for at least 10 seconds). We follow the original train/val/test BDD100k dataset split, but since the sensory data are not available for the test subset, we evaluate our model on the val subset only. In total, we used 31k stopping and 21k not-stopping sequences for training, and 4.6k stopping and 3.1k non-stopping sequences for evaluation. Each video sequence was sampled at 5fps. We did not use any of the sequences where the car was not moving at all or where it stopped in less than 10 seconds since the beginning of the video clip in either training or evaluation. We will publish the data split and the stopping events positions to make sure the experiment can be easily reproduced.

Results. We trained all models from scratch, using the same 3D-ResNet architecture (see section 3.4) as the backbone in all experiments. The only difference between them is the the dimensionality/nature of the last layer and loss function accordingly, which vary as explained in section 3.

Method	Output model	EPA [%]
WTTE-RNN [13]	Weibull Distribution	56.01
MS-LSTM [3]	Binary Classifiers	61.17
our method	Gaussian Mixture	**78.70**

Table 4. Car stopping prediction in the BDD100k driving dataset

Table 3 shows GMMH performs the best on this dataset. As shown in fig. 3, all models' prediction accuracy decreases as events more in the future are considered, but GMMH uniformly outperforms all other models.

We also visualize the attention of the network by displaying the values of the *Attention Map* layer, averaged over the N input frames. In fig. 1 we show that the network has for example learned to detect traffic lights, as well as to look for cars ahead on the road, as this is a good indicator or whether the car will shortly stop or not. The model has also learned that *when approaching a green traffic light, there is a still a probability the car might stop* (fig. 1 — middle row), as green might turn red before the car gets there. We note that all this was learned in an unsupervised way, only as a result of the requirement to predict the car stopping.

Comparison to Existing Methods. We also do our best to compare to existing methods in the literature. We exploit the architecture of Aliakbarian et al. [3], which is a LSTM with attention model for early event classification, and we adapt it to our domain by treating the event prediction as a classification problem (see Section 3.2.1). For this method, we use the *Binary Classifiers* output formulation, as this proved to be the best discrete-time model in our experiments. We train the models using the same training data and learning

rate schedule as our model and follow the same evaluation protocol (see Table 4). We show that the method performs worse than our method. This however is in part explained by the fact that this approach uses a different backbone (VGG) as the method also performs worse than 3D-ResNet with the same output formulation (see Table 3). We also adapt WTTE-RNN [13], which is a LSTM-based model used to predict events in low-dimensional data (such as engine failures based on a series of measurements). The model outputs Weibull distribution, which makes it extremely relevant to the domain of time to event prediction. We adapt it for image processing by plugging 3D ResNet-35 as a backbone, in order to generate low-dimensional features for the model, and train it jointly. The method performs only slightly better than the vanilla 3D ResNet-35 using Weibull Distribution (see Table 3), which suggests that the LSTM does not bring much of an additional benefit for the 3D ResNet backbone. It still however is worse than our GMMH model.

Comparison to Human Performance. We set up an experiment to assess the human performance for the task of breaking prediction. We showed 5 second video sequences ($N = 25$ frames) randomly picked from the above validation set to non-expert volunteers and asked them to answer the question "Will the car you are in stop within the next 10 seconds?" by clicking one of the two buttons. For stopping videos, we randomly picked from the interval $\Delta \in (1, 10)$ seconds before the car stopped, so that the last frame of the 5 second sequence was exactly Δ seconds before the stopping, and for non-stopping video we simply picked a random 5-second sequence from the video.

We collected around 1.5k data points from 28 non-expert volunteers (students), by asking them to perform this task for approximately 10 minutes, without giving them any immediate feedback to say if their answers were correct or not. As we show in Figure 4, the human performance decreases as events more in the future are considered, which is expected. More interestingly human performance in our experiment is actually slightly worse than the best model. We think that this is because it is harder for humans to establish a deeper understanding of the currently shown road scene from only a 5 second video sequence, and that humans were generally more defensive in their estimates. Many scenes in the dataset, for example, capture busy traffic situations where cars in front are slowing down/breaking but not actually stopping, but a natural reaction for a human is to expect the car will indeed stop as a result, leading to a biased answer. The network, on the other hand, can learn to exploit this prior in the data.

Ablation Studies. We study the contribution of the vehicle's velocity as a possible source for the stopping prediction. This is interesting, as one might expect that the stopping prediction could be mostly based on velocity information, i.e. the information that the vehicle has started to slow down. First, we train a multi-layer perception classifier, using 16-dimensional velocity vector as the input feature. The vector represents the current velocity in the respective 16 frames of the video sequence. We then compare the prediction performance to our model which uses only images as an input. We also combine both by concatenating the MLP output as the 513th channel for the fully connected layer. We used the one-in-many model in this experiment, so that we can have the same output encoding for all three options, including the standalone MLP.

Model	EPA [%]	TTEE [%]
Image only	71.87	2.66
Velocity only	61.89	6.92
Image + velocity	72.01	2.70

Table 5. Impact of the velocity input

As we show in table 5, the velocity channel on its own is not sufficient for a reliable stopping prediction — predicting whether the car will stop or not (EPA) is 10 percent point worse than when using only the image data, but more importantly the time-to-event prediction error (TTEE) is more than 2 times higher. This suggests that the car slowing down is a reasonably good indicator of that the car about to stop, but the exact time/place of stopping relies heavily on visual information from the scene. On the other hand, adding the velocity information to the image data has very limited impact to the accuracy, which suggests that the network actually already performs some velocity estimation from the image data.

In the second ablation study, we evaluate the impact of the number of observed video frames on the overall accuracy, when using the newly proposed Gaussian Mixture formulation (see table 2). We show that the accuracy is slowly decreased as less input frames are considered, however the difference between the input length of 24 and 16 frames is very small, but comes with almost double the computational cost. In the extreme case of observing a single frame, the accuracy drops significantly, because the network loses any information about the current car movement. We therefore opted to use 16 frames as the input of our model, as a trade-off between accuracy and training/run time.

Due to lack of space, the ablation study of the temporal resolution r is presented in the Supplementary material.

4.2. Basketball Shooting

In the second experiment setting, we aim to predict if and how long till a player is going to shoot a basketball towards the basket, using TV recordings of basketball games. The data is very different from the previous experiment, because the action is observed by an external observer as the camera aims at the field where the action is happening, and the video sequences are generally more challenging as basketball is a

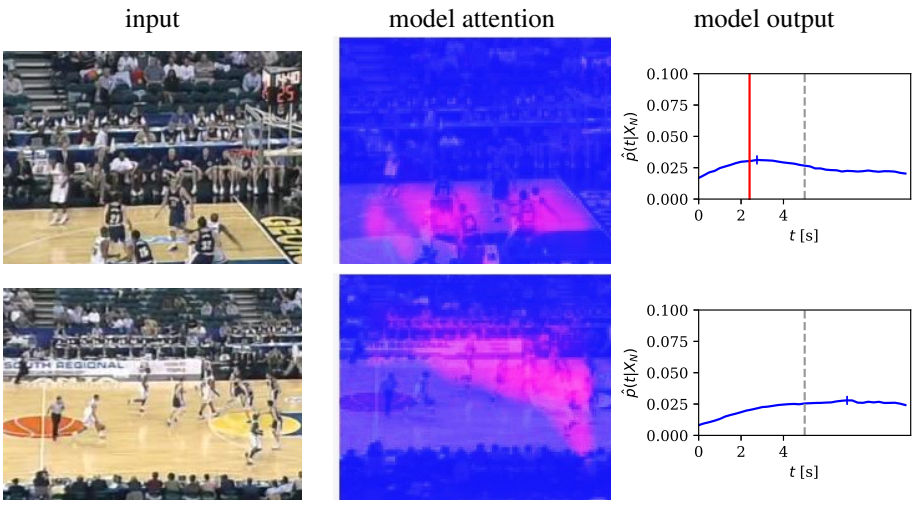

Figure 5. Predicting basketball shooting. Time to event probability prediction (blue), event occurrence ground truth (red), maximal prediction horizon $\Delta_{\max}$ (dashed gray).

model	EPA [%]	TTEE [s]	MS
One-in-many Classifier	53.71	5.00	N/A
Binary Classifiers	73.84	2.30	N/A
Direct Regression	71.87	2.68	N/A
Gaussian Distribution	53.80	4.99	7.09
Heuristic Heatmap	73.64	2.10	3.61
Gaussian Mixture	**76.42**	**1.42**	**3.02**

Table 6. Predicting basketball shooting

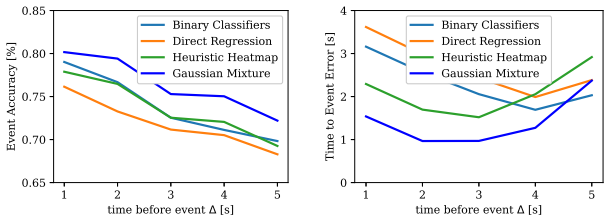

Figure 6. Basketball shooting EA (left) and TTEE (right)

fast-moving game and moreover there are often cuts between different cameras and therefore viewpoints within the video stream.

Dataset. We exploit the NCAA basketball dataset [16], which contains 296 basketball game recordings, each typically 1.5 hours long. The dataset comes with manually created annotations of 11 event types, from which we picked 4 classes (2-point success, 2-point fail, 3-point success and 3-point fail) representing ball leaving player's hands and used that as an annotation for the "ball throw" event used in our experiments. We follow the original data split and used the training & val subsets for training, and the test subset for evaluation. We do not use any of the localization (bounding box) information provided in the dataset.

Since the videos are much longer than in the driving case, we first split them into 30 second sequences and then label each sequence depending whether it contains the ball throw or not. Because the sequences without the ball throw prevail, we randomly subsample them to have a comparable number of sequences with and without the ball throw. As a result, the training set consists of 7k "ball throw" and 9k "no ball throw" sequences, and the testing set contains 1.1k "ball throw" and 1.4k "no ball throw" sequences. We will again publish the exact data split for reproducibility.

Results. We trained all models from scratch, using the same training protocol as in the previous section. In contrast to the previous section, we only train and evaluate the prediction in the interval of $(1, 5)$ seconds, because of generally faster pace of the action happening in the videos and to avoid issues with cuts in the TV stream.

Our GMMH again performs the best (see Table 6), significantly outperforming the other representations in all three metrics. Looking at the attention map, we observe that the network has learned to look for players and the team formation as a cue to predict whether a player is about to shoot the basketball or not (see fig. 5). We also point out that the output probability distribution has generally greater variance (i.e. "the sigmas are larger") than in the previous experiment, which is also reflected in the higher Model Surprise (3.02 for basketball vs. 2.69 for the stopping prediction — note this metric has a logarithmic scale). This suggests that generally there is a higher uncertainty in the underlying dataset.

5. Conclusion

We considered the problem of future event prediction: *if* and *when* an event will occur. We evaluated several possible representations and proposed a novel probabilistic GMMH model, which also outputs uncertainty of the prediction.

By evaluating in two entirely different testing scenarios, we demonstrated that we are able to predict events up to 10 seconds before they occur, and that using attention, we can demonstrate that the network has learned to look for meaningful cues, such as traffic lights. We also showed that in vehicle stopping prediction, our model outperforms an average human, which we contribute to the better ability of neural networks to learn domain specific priors and to capture subtle cues.

Acknowledgement

We are very grateful to Continental Corporation for sponsoring this research.

References

[1] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese. Social LSTM: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–971, 2016. 2

[2] M. S. Aliakbarian, F. Saleh, B. Fernando, M. Salzmann, L. Petersson, and L. Andersson. Deep action-and context-aware sequence learning for activity recognition and anticipation. *arXiv preprint arXiv:1611.05520*, 2016. 2

[3] M. S. Aliakbarian, F. S. Saleh, M. Salzmann, B. Fernando, L. Petersson, and L. Andersson. Encouraging LSTMs to anticipate actions very early. In *IEEE International Conference on Computer Vision (ICCV)*, volume 1, 2017. 2, 6

[4] F. Caba Heilbron, V. Escorcia, B. Ghanem, and J. Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–970, 2015. 2

[5] A. Dave, O. Russakovsky, and D. Ramanan. Predictivecorrective networks for action detection. In *Proceedings of the Computer Vision and Pattern Recognition*, 2017. 2

[6] P. Felsen, P. Agrawal, and J. Malik. What will happen next? forecasting player moves in sports videos. *ICCV, Oct*, 1:2, 2017. 2

[7] K. Hara, H. Kataoka, and Y. Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6546–6555, 2018. 5

[8] F. Heidarivincheh, M. Mirmehdi, and D. Damen. Action completion: A temporal model for moment detection. *arXiv preprint arXiv:1805.06749*, 2018. 2

[9] M. Hoai and F. De la Torre. Max-margin early event detectors. *International Journal of Computer Vision*, 107(2):191–202, 2014. 2

[10] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 3192–3199, 2013. 2

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 2

[12] X. Liang, L. Lee, W. Dai, and E. P. Xing. Dual motion gan for future-flow embedded video prediction. In *IEEE International Conference on Computer Vision (ICCV)*, volume 1, 2017. 2

[13] E. Martinsson. WTTE-RNN : Weibull Time To Event Recurrent Neural Network. Master's thesis, Chalmers University Of Technology, 2016. 2, 4, 6, 7

[14] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*, pages 483–499. Springer, 2016. 4

[15] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017. 5

[16] V. Ramanathan, J. Huang, S. Abu-El-Haija, A. Gorban, K. Murphy, and L. Fei-Fei. Detecting events and key actors in multi-person videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3043–3053, 2016. 2, 8

[17] M. S. Ryoo and J. Aggarwal. Ut-interaction dataset, icpr contest on semantic description of human activities (sdha). In *IEEE International Conference on Pattern Recognition Workshops*, volume 2, page 4, 2010. 2

[18] G. A. Sigurdsson, S. K. Divvala, A. Farhadi, and A. Gupta. Asynchronous temporal fields for action recognition. In *CVPR*, volume 5, page 7, 2017. 2

[19] G. A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, and A. Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, pages 510–526. Springer, 2016. 2

[20] H. Soleimani, J. Hensman, and S. Saria. Scalable joint models for reliable uncertainty-aware event prediction. *IEEE transactions on pattern analysis and machine intelligence*, 40(8):1948–1963, 2018. 2

[21] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013. 5

[22] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler. Efficient object localization using convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–656, 2015. 4

[23] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in neural information processing systems*, pages 1799–1807, 2014. 4

[24] C. Vondrick, H. Pirsiavash, and A. Torralba. Anticipating visual representations from unlabeled video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 98–106, 2016. 2

[25] C. Vondrick, H. Pirsiavash, and A. Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, pages 613–621, 2016. 2

[26] C. Vondrick and A. Torralba. Generating the future with adversarial transformers. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, page 3, 2017. 2

[27] D. Wei, J. Lim, A. Zisserman, and W. T. Freeman. Learning and using the arrow of time. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2

[28] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015. 5

[29] T. Xue, J. Wu, K. Bouman, and B. Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2016. 2

[30] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2018. 5