Discrimination of Different Basic Eye Movement Types

Florian Strohm
University of Stuttgart
Stuttgart, Germany
st109838@stud.uni-stuttgart.de

ABSTRACT

The segmentation of eye tracking data into specific types of eye movements is often realized using algorithms which rely on carefully chosen parameters. Recent work showed that the utilization of deep learning models can significantly outperform traditional algorithms. In this work we further explore the capabilities of deep learning models in this domain by using an attention convolutional neural network that classifies raw gaze samples as fixations, saccades, smooth pursuit or noise. The results show that our model outperforms current state-of-the-art end-to-end performance on the GazeCom data set. Since the available labeled data sets are each restricted to a specific domain or a small amount of subjects, the generalization capabilities of deep learning models are limited. Therefore, we additionally investigate the impact of unsupervised pre-training on the predictive performance and generalizability of deep learning models. For this we train an autoencoder using eye tracking recordings of the Hollywood2 data set and use the learned weights in a deeper neural network architecture for possibly better generalization and reduced overfitting.

KEYWORDS

eye movement; eye tracking; deep learning; fixation; saccade; smooth pursuit

ACM Reference Format:
Florian Strohm. 2018. Discrimination of Different Basic Eye Movement Types. In *Proceedings of Fachpraktikum Interaktive Systeme (FIS'18).* ACM, New York, NY, USA, 7 pages.

1 INTRODUCTION

In the last years eye trackers have become smaller, cheaper and easier to use. Currently more and more consumers are using eye trackers, especially in the video entertainment area. With the increasing popularity of eye trackers, research in this area becomes more relevant as the spread of such devices allows for more different use-cases. There are many gaze applications that require an automated eye movement detection, especially in human-computer interaction and experimental psychology. Some examples of gaze applications in this fields are:

- activity recognition [18] [3]
- predicting search target [15] [14]
- cognitive workload detection [19]
- predicting visual memory recall [4]
- predicting personality traits [8] [13]

For those applications specific eye movement events have to be detected within the gaze data, but the output of an eye tracker are just x and y positions of the users gaze in a calibrated coordinate system. This information is not sufficient as the raw gaze data does not directly imply which eye movement a user performed. For example, if a user is looking from one point to another, the raw data may suggest that the user looked at many points in between, whereas in reality the user performed two fixations with a saccade in between. There are many different methods to extract such eye movement events from the gaze data, but most of them rely on handcrafted rules and thresholds which have to be selected by the user and do not generalise very well. Recent work showed that machine learning models can outperform these algorithms and do not need any adjustments by the user ([17], [7], [21], [22]) . Therefore, in this work we are utilizing deep learning methods to automatically identify different eye movement events in the raw gaze data, namely fixations, saccades and smooth pursuits. We are using an attention convolutional neural networks (ACNN) and show that it is superior to classical CNN architectures and current state of the art performance by Startsev et al. [17]. Additionally, we are comparing the performances of deeper architectures and regularise them by training the feature extractor layers unsupervised on a different data set.[1]

2 RELATED WORK

The first work that utilized deep learning models for automated eye movement detection was by Hoppe and Bulling [7]. In their work they used a CNN architecture with one

[1]All the relevant code is accessible online in the folloing repository: https://github.com/StrohmFn/eye_movement_detection

convolution and max pooling layer followed by a dense and the output layer. The network was trained to detect fixations, saccades and smooth pursuits. They do not input the raw x and y gaze coordinates but apply a fast Fourier transform (FFT) first. The frequency domain of the gaze data should help the network to distinguish between the different classes more easily. They also use a sliding window approach with window size of 30. As the used data set was recorded using a 300Hz eye-tracker, this window size corresponds to a tenth of a second. Our approach is similar but we are using a larger context window size and eliminate the FFT step to create a fully end-to-end system.

In the work of Zemblys et al. [21] they use a neural network to predict fixations, saccades and post-saccadic oscillations. Their network consists of two convolution layers followed by three gated recurrent layers and the output layer. Instead of using a hand labeled data set for training the classification network, they use the data to train a generator network. This generator network outputs a mixture of Gaussians from which the next gaze sample position and the corresponding label can be drawn. Using this generator they created a synthetic data set which is 450 times larger than the data used to train the generator. This data is then used to train the actual network which shows superior performance than other approaches.

Startsev et al. [17] use a CNN-BLSTM network architecture to predict gaze samples as either fixations, saccades or smooth pursuits. Unlike the other approaches they do not only classify a single sample per forward pass but classify all samples in a certain window at once. To do so they use a sequence to sequence BLSTM unit with three convolution layers upfront, that prepare the input for the BLSTM. They train and evaluate their network on the GazeCom [5] [16] data set and show superior performance compared to many other algorithms.

3 METHODOLOGY

The disadvantage in the approach from Startsev et al. is that they have very different context windows for each input sample. For example, the network has no information about the past for the leftmost sample in the input and no information about the future for the rightmost sample. In this work we are using a sliding window approach for the input to all neural network architectures. With this approach we are just classifying the sample in the center of the window, while every other sample only serves as context information for the network. This ensures equal context information for each to be classified sample. A re-implementation of the network by Startsev et al. with this windowing approach turned out to be infeasible for real time applications, because the network is too slow. Therefore we are using convolutional neural networks only, as they are substantially faster than LSTM networks.

Attention Convolutional Neural Network

Attention networks showed a performance gain in different time-series prediction domains like emotion recognition from speech [12]. We want to transfer this approach to the eye movement detection task and therefore, the main architecture in this work is an attention convolutional neural network as can be seen in Figure 1.

The window size of the network is 251 samples with two channels, one for x and one for y gaze position. This corresponds to about one second of context information as we are using 250 hertz eye tracking recordings. Therefore the network receives half a second of information about past and future gaze locations. This also means that the classification of each sample is delayed by half a second to obtain the future gaze information. The intuition behind the attention network is that it learns to focus on important samples in the input as most likely not all 251 samples are equally relevant for classifying the center sample.

The first layer of the network is a convolution layer with 16 filters of kernel size 15. The large kernel size enables the network to learn features that require a larger temporal context. Startsev et al. showed that speed and direction of eye movements are valuable features for classification. In theory, this features can be calculated using two samples but this estimate can be very noisy due to the eye tracker sampling rate and eye movement jitter. A kernel size of 15 should therefore enable a more robust estimate of these features. Furthermore, the x and y gaze positions are blended together after the first layer, therefore we want a large receptive field while they are still separated. The convolution layer is followed by a max pooling layer with kernel size two in order to reduce dimensionality, while also introducing some invariances. The output is then passed through another convolution layer similar to the first one but with softmax activation. This layer is the main part of the attention mechanism because the softmax activation function forces the network to focus on specific parts of the input. The output of this layer is as a probability distribution over its input. As this probability distribution is subsequently multiplied with the previous output of the max pooling layer, the semantic meaning of this distribution can be interpreted as the relevancy of a weight for the classification task. The attention mechanism is followed by a dense layer with 32 neurons and finally the output layer which classifies the input either as fixation, saccade, smooth pursuit or noise.

Unsupervised Pre-Training with an Autoencoder

The previously described ACNN is not very deep but increasingly overfits the training data if more layers are added. The

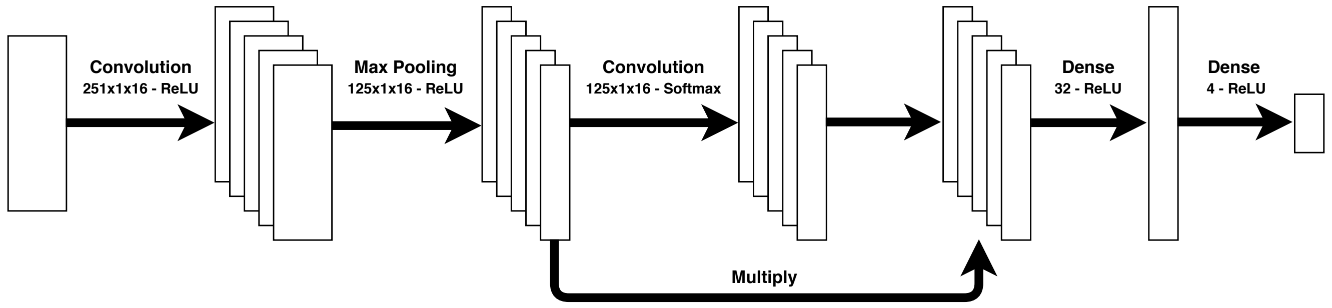

Figure 1: Architecture of the ACNN.

amount of annotated training data is limited which also limits the size of the network. Besides annotated data sets there are also large data sets without annotations, which we can use to train an autoencoder. The idea is that the autoencoder is trained unsupervised on an large data set and learns to extract the most relevant features of the input samples. After the autoencoder is trained we copy the learned weights and use them in an deeper CNN network. We train the CNN network using a labeled data set but freeze the pre-trained values so the network can only change parameters in the dense layers. This approach enables the usage of deeper models but limits the amount of trainable parameters and thus also limits the degree of overfitting. The autoencoder is trained via backpropagation like a normal neural network but the target output is equal to the input of the network. The architecture of our autoencoder network is shown in Figure 2.

Similar to the previous ACNN, the sliding window contains 251 gaze samples, but is padded with zeros to reach a size of 256, as even tensor sizes are needed for later upsampling processes in the decoder. At first the input is passed through three convolution and max pooling blocks which transform the input into a latent feature space representation. The first convolution layer has a kernel size of 15 while the following layers have kernel size three. Instead of decoding already we additionally add two dense layers to further reduce the dimensionality and finally reach a bottleneck of size 64. Consequently, the autoencoder has to fit the most relevant information into this 64 neurons to then subsequently reconstruct the input. After the bottleneck layer the network is basically mirrored but traversed in reversed order as visualized in Figure 2. Instead of max pooling we are using upsampling layers in the decoder part which repeats each temporal step twice along the time axis, doubling the size of the tensor. We are not using an attention mechanism in the autoencoder architecture as their goals are at odds. The attention mechanism forces the network to focus on the most relevant segments of the input while the autoencoder tries to reconstruct the input as a whole and thus needs to consider all samples. Additionally to the architecture shown in Figure 2, we are also using an even deeper model with a similar design but with ten convolution layers and no dense layers.

4 DATA SETS

For the training and testing of our models we use three different data sets. The GazeCom data set is used to train the different classifier networks in a supervised manner. The Hollywood2 data set is used to pre-train the classifier network using the autoencoder architecture. Finally, we are using the lund2013 data set to test the generalizability of the different models on completely new data. The hypothesis is that the pre-trained models generalize better to different data than the fully supervised trained models.

GazeCom

The GazeCom data set was created by Dorr et al. [5]. They used an HD video camera to record 18 videos in real-world scenarios, each around 20 seconds long. The data set also contains stop motion videos, movie trailers and static images. Unfortunately, no annotations are available for these stimuli which is why we are using the HD video data only. To create the gaze data they used a stationary 250Hz eye-tracker. The data set contains eye-tracking information from a total of 54 subjects that watched all 18 videos. There was no specific task given to the subjects other than just watching the video. Recordings that contained too many low confidence gaze samples were discarded, leaving between 37 and 52 recordings per video. Later Agtzidis et al. [1] manually annotated this data set with the labels fixation, saccade, smooth pursuit and noise. Sample wise, the data set contains 3.132.536 fixation, 454.787 saccade, 475.817 smooth pursuit and 254.916 noise samples. Since we have a large sample imbalance towards fixations the models would overfit to predict mostly this class. We therefore use data augmentation to obtain a more balanced data set. We used different augmentation techniques, each having different assumptions on the eye-movement behaviour. The most harmless augmentations are

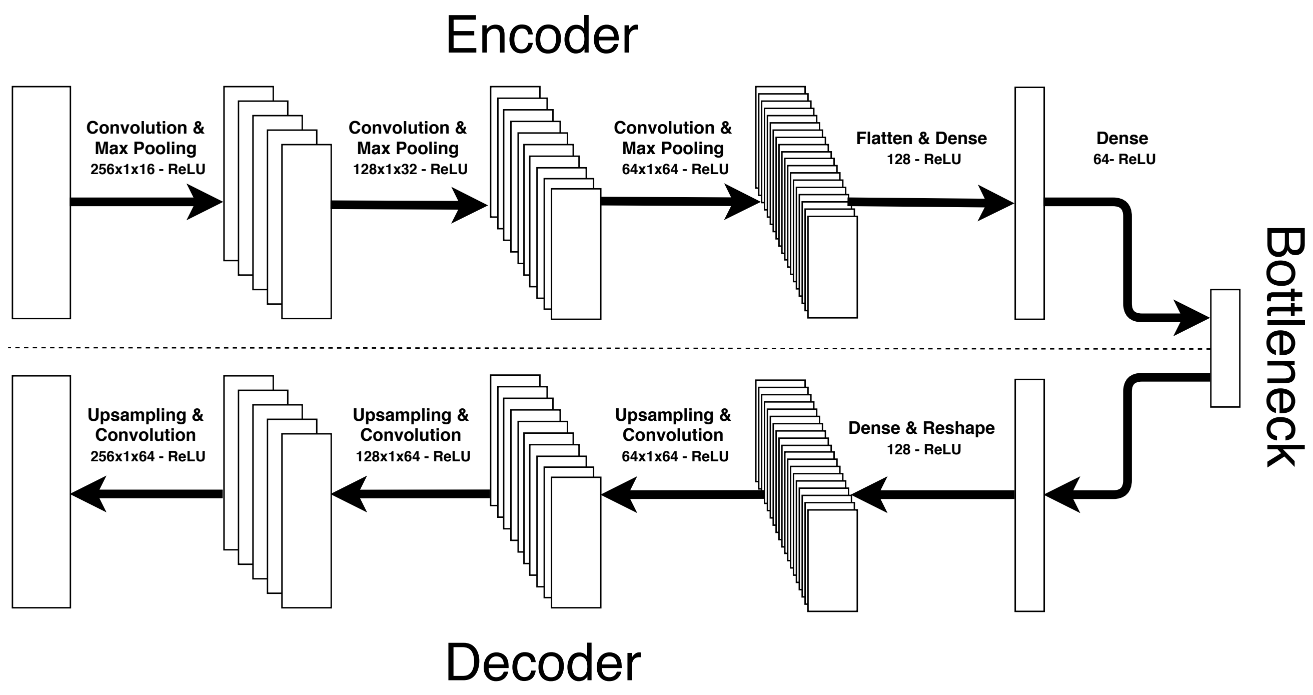
Encoder
Convolution & Max Pooling 256x1x16 - ReLU
Convolution & Max Pooling 128x1x32 - ReLU
Convolution & Max Pooling 64x1x64 - ReLU
Flatten & Dense 128 - ReLU
Dense 64- ReLU
Bottleneck
Dense & Reshape 128 - ReLU
Upsampling & Convolution 64x1x64 - ReLU
Upsampling & Convolution 128x1x64 - ReLU
Upsampling & Convolution 256x1x64 - ReLU
Decoder

Figure 2: Architecture of the autoencoder network.

probably mirroring a window of samples on the x-axis and/or on the y-axis. The assumption on this type of augmentation is that eyes behave similar from left to right as they do from right to left and analogously for vertical movements; this assumption seems realistic. As the data is still unbalanced using this augmentation, we further use the eye-movements in reversed order. The assumptions made for this augmentations are certainly not true because the acceleration and deceleration of eye-movements do not behave equally. Besides that, eye-movements like post-saccadic oscillations occur, as the name suggests, after a saccade, but by reversing the gaze data PSOs occur before each saccade. However, for our considered eye-movement types this assumption seems not too harmful and helps to finally balance the data set. We also experimented with rotating the gaze data, but this resulted in a significant loss in classification accuracy. If we rotate for example by 90 degrees, we assume that vertical and horizontal movements are equal. Since the field of view of a human eye is very wide this assumption is not feasible. In the end the augmentations we used are:

- sequence mirrored on x-axis
- sequence mirrored on y-axis
- sequence mirrored on x- and y-axis
- sequence reversed
- sequence mirrored on x-axis and reversed
- sequence mirrored on y-axis and reversed

We applied this augmentations on all sequences that are labeled as either saccade, smooth pursuit or noise. This results in a much more balanced data set now containing 3.132.536 fixation, 3.183.509 saccade, 3.330.719 smooth pursuit and 1.784.412 noise samples.

Hollywood2

The Hollywood2 data set [11] contains almost six hours of video data collected from 69 different Hollywood movies split into 1707 small clips. The data set was created for the task of action recognition, wherefore the clips show different actions like running, eating or driving. Vig et al. [20] conducted a gaze data collection study for the Hollywood2 data set. Five subjects watched all of the 1707 clips while their gaze was recorded via an eye-tracker at 1000Hz. The task of the subjects was to identify the action performed in the clip. Recordings with too much noise were again discarded, leaving between 1554 and 1706 recordings per subject. We downsampled the data to 250Hz by skipping three samples for each time step. We use this data set without any augmentation to train our autoencoder networks.

Lund2013

To deeper evaluate and compare the performance of the different models we use the lund2013 data set by Larsson et al. [10] [9] [2]. This data set contains gaze data for three different stimuli: moving dots, images and videos. We are only using the gaze data for the video stimuli as our models are trained on this domain. The gaze was recorded at 500Hz from 33 different participants watching several video clips, however, only 28 recordings were annotated by two different raters. The gaze samples are annotated with the labels fixation, saccade, post-saccadic oscillation, smooth pursuit, blink and undefined. For our evaluation we ignore samples labeled as post-saccadic oscillation, blink or undefined. We downsample the data to 250Hz by skipping every second sample for each time step and did not perform any data augmentation. The final data set contains 66.807 fixation, 8.429 saccade and 109.752 smooth pursuit samples. The high number of smooth pursuits is due to the fact that the subjects were explicitly instructed to follow moving objects.

5 EXPERIMENTS & RESULTS

For all experiments where we train a model using the GazeCom data set we are using leave one video out (LOVO) cross validation. Since the data set contains 18 videos, we are training 18 different models and report the average F_1 score for each experiment. The results of all our experiments are shown in Table 1.

	GazeCom			
Model	avg. F_1	Fixation	Saccade	SP
ACNN	<u>0.784</u>	<u>0.937</u>	0.830	**0.584**
ACNN-half-size	0.761	0.930	0.809	0.545
ACNN-real-time	0.677	0.917	0.776	0.337
CNN-1-layer	0.753	0.931	0.830	0.499
CNN-3-layer-pre	**0.789**	**0.938**	<u>0.854</u>	<u>0.573</u>
CNN-10-layer-pre	0.745	0.917	0.801	0.515
1D CNN-BLSTM	0.762	0.913	**0.855**	0.517
1D CNN-BLSTM speed and direction	0.830	0.939	0.893	0.703

Table 1: Comparison of eye-movement detectors on the GazeCom data set. The results of the best models are highlighted in boldface, while the second best models are underlined.

The model *1D CNN-BLSTM* is the model by Startsev et al. [17]. Note that this is their model that also only uses the raw x and y gaze positions. Their best performing model is not end-to-end as it uses the precomputed features speed and direction and is therefore not directly comparable with our results. As the table shows, our *ACNN* model slightly outperforms their model by 2.2% in average F_1. The main difference is present in smooth pursuit detection with a 6.7% increase in F_1.

Since our standard ACNN model requires 125 samples after the to be predicted sample, the prediction is delayed by half a second. For some applications this delay may not be feasible which is why we further investigate the performance of smaller windows. The model *ACNN-half-size* uses a context window size of 125; 62 samples from the past and 62 samples from the future. This reduces the classification delay to one fourth of a second but also causes some performance loss, especially for smooth pursuit.

To further understand the importance of the future samples we trained the ACNN model with no future samples at all, called *ACNN-real-time*. As the table shows, we have a sharp performance drop for smooth pursuit detection that shows how important future samples are for this class. This observation is in line with the results from Startsev et al. [17] and might also be a reason why our ACNN model outperforms their model. Our model has a constant amount of future information whereas the CNN-BLSTM model has to classify some samples with little to no future information.

The model *CNN-1-layer* is similar to the ACNN model but without the attention mechanism. We can clearly see that the attention mechanism significantly increases the smooth pursuit detection performance with an increase of 8.5% in F_1 score. However, the attention mechanism seems to have little to no impact for fixation and saccade prediction.

If we create models with significantly more parameters, they increasingly overfit and the cross-validation performance drops. In the next two experiments we therefore try to counteract this overfitting effect by pre-training the models *CNN-3-layer-pre* and *CNN-10-layer-pre*. As described in section 3 we train the networks unsupervised with an autoencoder architecture on the Hollywood2 data set and then use the learned weights in the actual model. We append a trainable dense and classification layer to the freezed convolution layers. As the results show, the pre-training procedure helps to prevent overfitting, but no significant performance gains can be achieved in comparison to the *ACNN* model. The *CNN-3-layer-pre* model is on a par with the *ACNN* model, while the *CNN-10-layer-pre* model achieves a lower F_1 score overall. The reason to this may be that it slightly overfitts the Hollywood2 data set while unsupervised training. It is difficult to train a good autoencoder, as the only observable metric is training loss. This metric is not helpful for determining if the model learns something useful. More research in this area could definitely lead to better results for pre-trained models.

For the last experiment we evaluate the performance of our previously trained models on the independent data set lund2013. For each network architecture we evaluate all 18 models and report the mean F_1 score in Table 2. It shows that

	Lund2013			
Model	avg. F_1	Fixation	Saccade	SP
ACNN	0.272	0.352	0.0514	**0.412**
CNN-1-layer	0.283	0.440	**0.054**	0.355
CNN-3-layer-pre	**0.297**	<u>0.445</u>	<u>0.052</u>	<u>0.395</u>
CNN-10-layer-pre	<u>0.292</u>	**0.454**	**0.054**	0.369

Table 2: Comparison of eye-movement detectors on the Lund2013 data set. The results of the best models are highlighted in boldface, while the second best models are underlined.

the generalisability of all models is pretty bad, especially for saccade detection. This experiment confirms that the attention mechanism is helpful for smooth pursuit detection because the *ACNN* model again achieves the highest F_1 score for this class. Furthermore, the results do not show that pre-trained models generalized better since their F_1 score is similar to the not pre-trained *CNN-1-layer* model.

6 SUMMARY AND OUTLOOK

We have proposed a simple neural network architecture to detect fixations, saccades and smooth pursuits in raw gaze data. Our attention convolutional neural network slightly outperforms the currently best performing end-to-end model by Startsev et al. [17]. The results indicate that the attention mechanism is very helpful to detect smooth pursuits because the ACNN achieved a noticeable higher F_1 score for this class in all tests. Furthermore, we tried to train deeper models by pre-training them with a different data set. The results show that pre-training is certainly possible but do not show significant improvements yet. A much larger and diverse data set might help for better generalization of deeper models and since pre-training does not require any labels, creating such a data set seems reasonable.

It would be interesting to experiment with a deep pre-trained model with an attention mechanism to benefit from its superior smooth pursuit detection capability. Unfortunately, the most reasonable position for the attention mechanism is at the first layer, which makes it difficult to train the mechanism without training the whole network again.

Future work could also use the precomputed features *speed* and *direction* which allows for a direct comparison with the current best performing model on the GazeCom data set. However, the focus of this work was to create a fully end-to-end gaze classifier.

REFERENCES

[1] Ioannis Agtzidis, Mikhail Startsev, and Michael Dorr. 2016. In the pursuit of (ground) truth: A hand-labelling tool for eye movements recorded during dynamic scene viewing. In *2016 IEEE second workshop on eye tracking and visualization (ETVIS)*. IEEE, 65–68.

[2] Richard Andersson, Linnea Larsson, Kenneth Holmqvist, Martin Stridh, and Marcus Nyström. 2017. One algorithm to rule them all? An evaluation and discussion of ten eye movement event-detection algorithms. *Behavior research methods* 49, 2 (2017), 616–637.

[3] Ali Borji and Laurent Itti. 2014. Defending Yarbus: Eye movements reveal observers' task. *Journal of vision* 14, 3 (2014), 29–29.

[4] Andreas Bulling and Daniel Roggen. 2011. Recognition of visual memory recall processes using eye movement analysis. In *Proceedings of the 13th international conference on Ubiquitous computing*. ACM, 455–464.

[5] Michael Dorr, Thomas Martinetz, Karl R Gegenfurtner, and Erhardt Barth. 2010. Variability of eye movements when viewing dynamic natural scenes. *Journal of vision* 10, 10 (2010), 28–28.

[6] John M Henderson, Svetlana V Shinkareva, Jing Wang, Steven G Luke, and Jenn Olejarczyk. 2013. Predicting cognitive state from eye movements. *PloS one* 8, 5 (2013), e64937.

[7] Sabrina Hoppe and Andreas Bulling. 2016. End-to-end eye movement detection using convolutional neural networks. *arXiv preprint arXiv:1609.02452* (2016).

[8] Sabrina Hoppe, Tobias Loetscher, Stephanie Morey, and Andreas Bulling. 2015. Recognition of curiosity using eye movement analysis. In *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers*. ACM, 185–188.

[9] Linnéa Larsson, Marcus Nyström, Richard Andersson, and Martin Stridh. 2015. Detection of fixations and smooth pursuit movements in high-speed eye-tracking data. *Biomedical Signal Processing and Control* 18 (2015), 145–152.

[10] Linnéa Larsson, Marcus Nyström, and Martin Stridh. 2013. Detection of saccades and postsaccadic oscillations in the presence of smooth pursuit. *IEEE Transactions on biomedical engineering* 60, 9 (2013), 2484–2493.

[11] Marcin Marszałek, Ivan Laptev, and Cordelia Schmid. 2009. Actions in context. In *CVPR 2009-IEEE Conference on Computer Vision & Pattern Recognition*. IEEE Computer Society, 2929–2936.

[12] Michael Neumann and Ngoc Thang Vu. 2017. Attentive Convolutional Neural Network Based Speech Emotion Recognition: A Study on the Impact of Input Features, Signal Length, and Acted Speech. *Proc. Interspeech 2017* (2017), 1263–1267.

[13] Evan F Risko, Nicola C Anderson, Sophie Lanthier, and Alan Kingstone. 2012. Curious eyes: Individual differences in personality predict eye movement behavior in scene-viewing. *Cognition* 122, 1 (2012), 86–90.

[14] Hosnieh Sattar, Andreas Bulling, and Mario Fritz. 2017. Predicting the category and attributes of visual search targets using deep gaze pooling. In *Proceedings of the IEEE International Conference on Computer Vision*. 2740–2748.

[15] Hosnieh Sattar, Sabine Muller, Mario Fritz, and Andreas Bulling. 2015. Prediction of search targets from fixations in open-world settings. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 981–990.

[16] Mikhail Startsev, Ioannis Agtzidis, and Michael Dorr. 2016. Smooth Pursuit. http://michaeldorr.de/smoothpursuit/.

[17] Mikhail Startsev, Ioannis Agtzidis, and Michael Dorr. 2018. 1D CNN with BLSTM for automated classification of fixations, saccades, and smooth pursuits. *Behavior Research Methods* (08 Nov 2018). https://doi.org/10.3758/s13428-018-1144-2

[18] Julian Steil and Andreas Bulling. 2015. Discovery of everyday human activities from long-term visual behaviour using topic models. In *Proceedings of the 2015 acm international joint conference on pervasive and ubiquitous computing*. ACM, 75–85.

[19] Els Stuyven, Koen Van der Goten, André Vandierendonck, Kristl Claeys, and Luc Crevits. 2000. The effect of cognitive load on saccadic eye movements. *Acta psychologica* 104, 1 (2000), 69–85.

[20] Eleonora Vig, Michael Dorr, and David Cox. 2012. Space-variant descriptor sampling for action recognition based on saliency and eye movements. In *European conference on computer vision*. Springer, 84–97.

[21] Raimondas Zemblys, Diederick C Niehorster, and Kenneth Holmqvist. 2018. gazeNet: End-to-end eye-movement event detection with deep neural networks. *Behavior research methods* (2018), 1–25.

[22] Raimondas Zemblys, Diederick C Niehorster, Oleg Komogortsev, and Kenneth Holmqvist. 2018. Using machine learning to detect events in eye-tracking data. *Behavior research methods* 50, 1 (2018), 160–181.