1D CNN with BLSTM for automated classification of fixations, saccades, and smooth pursuits

Mikhail Startsev¹ · Ioannis Agtzidis¹ · Michael Dorr¹

Published online: 8 November 2018
© Psychonomic Society, Inc. 2018

Abstract

Deep learning approaches have achieved breakthrough performance in various domains. However, the segmentation of raw eye-movement data into discrete events is still done predominantly either by hand or by algorithms that use hand-picked parameters and thresholds. We propose and make publicly available a small 1D-CNN in conjunction with a bidirectional long short-term memory network that classifies gaze samples as fixations, saccades, smooth pursuit, or noise, simultaneously assigning labels in windows of up to 1 s. In addition to unprocessed gaze coordinates, our approach uses different combinations of the speed of gaze, its direction, and acceleration, all computed at different temporal scales, as input features. Its performance was evaluated on a large-scale hand-labeled ground truth data set (GazeCom) and against 12 reference algorithms. Furthermore, we introduced a novel pipeline and metric for event detection in eye-tracking recordings, which enforce stricter criteria on the algorithmically produced events in order to consider them as potentially correct detections. Results show that our deep approach outperforms all others, including the state-of-the-art multi-observer smooth pursuit detector. We additionally test our best model on an independent set of recordings, where our approach stays highly competitive compared to literature methods.

Keywords Eye-movement classification · Deep learning · Smooth pursuit

Introduction

Eye-movement event detection is important for many eye-tracking applications as well as the understanding of perceptual processes. Automatically detecting different eye movements has been attempted for multiple decades by now, but evaluating the approaches for this task is challenging, not least because of the diversity of the data and the amount of manual labeling required for a meaningful evaluation. To compound this problem, even manual annotations suffer from individual biases and implicitly used thresholds and rules, especially if experts from different sub-areas are involved (Hooze, Niehorster,

Nyström, Andersson, & Hessels, 2017). For smooth pursuit (SP), even detecting episodes¹ by hand is not entirely trivial (i.e., requires additional information) when the information about their targets is missing. Especially when pursuit speed is low, it may be confused with drifts or oculomotor noise (Yarbus, 1967).

Therefore, most algorithms to date are based on hand-tuned thresholds and criteria (Larsson, Nyström, Andersson, & Stridh, 2015; Berg, Boehnke, Marino, Munoz, & Itti, 2009; Komogortsev, Gobert, Jayarathna, Koh, & Gowda, 2010). The few data-driven methods in the literature either do not consider smooth pursuit (Zemblys, Niehorster, Komogortsev, & Holmqvist, 2017), operate on data produced with low-variability synthetic stimuli (Vidal, Bulling, & Gellersen, 2012), or are not yet publicly available (Hoppe & Bulling, 2016).

Here, we propose and make publicly available a neural network architecture that differentiates between the three major eye-movement classes (fixation, saccade, and smooth

✉ Mikhail Startsev
mikhail.startsev@tum.de

Ioannis Agtzidis
ioannis.agtzidis@tum.de

Michael Dorr
michael.dorr@tum.de

¹ Technical University of Munich, Institute for Human-Machine Communication, Arcisstr. 21, Munich, 80333, Germany

¹We use the terms “event” and “episode” interchangeably, both referring to an uninterrupted interval, where all recorded gaze samples have been assigned the same respective label, be that in the ground truth or in algorithmic labels.

pursuit) while also taking potential noise (e.g., blinks or lost tracking) into account. Our approach learns from data (simple features) to assign sequences of labels to sequences of data points with a compact one-dimensional convolutional neural network (1D-CNN) and bidirectional long short-term memory block (BLSTM) combination. Evaluated on a fully annotated² (Startsev, Agtzidis, & Dorr, 2016) GazeCom data set of complex natural movies (Dorr, Martinetz, Gegenfurtner, & Barth, 2010), our approach outperforms all 12 evaluated reference models, for both sample- and event-level detection. We additionally test our method's generalization ability on the data set of Andersson, Larsson, Holmqvist, Stridh, and Nyström (2017).

Related work

Data sets

Despite the important role of smooth pursuit in our perception and everyday life (Land, 2006), its detection in free-viewing scenarios has been somewhat neglected. At the very least, it should be considered in event detectors to avoid false detections of other eye-movement types (Andersson et al., 2017). Even when taking into account information about gaze patterns of dozens of observers at once (Agtzidis, Startsev, & Dorr, 2016b), there is a dramatic gap between the performance of detecting saccades or fixations, and detecting smooth pursuits (Startsev et al., 2016).

We will, therefore, use the largest publicly available manually annotated eye-tracking data set that accounts for smooth pursuit to train and validate our models: GazeCom (Dorr et al., 2010; Startsev et al., 2016) (over 4 h of 250-Hz recordings with SR Research EyeLink II). Its data files contain labels of four classes, with noise labels (e.g., blinks and tracking loss) alongside fixations, saccades, and smooth pursuits. We maintain the same labeling scheme in our problem setting (including the introduced, yet unused, “unknown” label).

We additionally evaluate our approach on a small high-frequency data set that was originally introduced by Larsson, Nyström, and Stridh (2013) (data available via (Nyström, 2015); ca. 3.5 min of 500-Hz recordings with SensoMotoric Instruments Hi-Speed 1250), also recently used in a larger review of the state of the art by Andersson et al. (2017). This data set considers postsaccadic oscillations in manual annotation and algorithmic analysis, which is not common yet for eye-tracking research.

Another publicly available data set that includes smooth pursuit, but has low temporal resolution, accompanies the work of Santini, Fuhl, Kübler, and Kasneci (2016) (available at Santini, 2016; ca. 15 min of 30-Hz recordings with a Dikablis Pro eye tracker). This work, however, operates in a different data domain: pupil coordinates on raw eye videos. Because it was not necessary for the algorithm, no eye tracker calibration was performed, and therefore no coordinates are provided with respect to the scene camera. Post hoc calibration is possible, but it is recording-dependent. Nevertheless, the approach of Santini et al. (2016) presents an interesting ternary (fixations, saccades, smooth pursuit) probabilistic classifier.

Automatic detection

Many eye-movement detection algorithms have been developed over the years. A simple, yet versatile toolbox for eye-movement detection is provided by Komogortsev (2014). It contains Matlab implementations for a diverse set of approaches introduced by different authors. Out of the eight included algorithms, five (namely, I-VT and I-DT (Salvucci & Goldberg, 2000), I-HMM (Salvucci & Anderson, 1998), I-MST (Goldberg & Schryver, 1995; Salvucci & Goldberg, 2000), and I-KF Sauter, Martin, Di Renzo, & Vomscheid, 1991) detect fixations and saccades only (cf. Komogortsev et al. 2010 for details).

I-VVT, I-VMP, and I-VDT, however, detect the three eye-movement types (fixations, saccades, smooth pursuit) at once. I-VVT (Komogortsev & Karpov, 2013) is a modification of the I-VT algorithm, which introduces a second (lower) speed³ threshold. The samples with speeds between the high and the low thresholds are classified as smooth pursuit. The I-VMP algorithm (San Agustin, 2010) keeps the high speed threshold of the previous algorithm for saccade detection, and uses window-based scoring of movement patterns (such as pair-wise magnitude and direction of movement) for further differentiation. When the score threshold is exceeded, the respective sample is labeled as belonging to a smooth pursuit. I-VDT (Komogortsev & Karpov, 2013) uses a high speed threshold for saccade detection, too. It then employs a modified version of I-DT to separate pursuit from fixations.

Dorr et al. (2010) use two speed thresholds for saccade detection alone: The high threshold is used to detect the peak-speed parts in the middle of saccades. Such detections are then extended in time as long as the speed stays above the low threshold. This helps filter out tracking noise and other artifacts that could be mistaken for a saccade, if only

²The recordings were algorithmically pre-labeled to speed up the hand-labeling process, after which three manual annotators have verified and adjusted the labeled intervals in all of the files.

³Unfortunately, in the eye-movement literature, the term “velocity” (in physics, a vector) often is used to refer to “speed” (the scalar magnitude of velocity). Here, we try to be consistent and avoid using “velocity” when it is not justified.

the low threshold was applied. Fixations are determined while trying to avoid episode contamination with smooth pursuit samples. The approach uses a sliding window inside intersaccadic intervals, and the borders of fixations are determined via a combination of modified dispersion and speed thresholds.

Similarly, the algorithm proposed by Berg et al. (2009) was specifically designed for dynamic stimuli. Here, however, the focus is on distinguishing saccades from pursuit. After an initial low-pass filtering, the subsequent classification is based on the speed of gaze and principal component analysis (PCA) of the gaze traces. If the ratio of explained variances is near zero, the gaze follows an almost straight line. Then, the window is labeled either as a saccade or smooth pursuit, depending on speed. By combining information from several temporal scales, the algorithm is more robust at distinguishing saccades from pursuit. The samples that are neither saccade nor pursuit are labeled as fixations. The implementation is part of a toolbox (Walther & Koch, 2006).

An algorithm specifically designed to distinguish between fixations and pursuit was proposed by Larsson et al. (2015), and its re-implementation is provided by Startsev et al. (2016). It requires a set of already detected saccades, as it operates within intersaccadic intervals. Every such interval is split into overlapping windows that are classified based on four criteria. If all criteria are fulfilled, the window is marked as smooth pursuit. If none are fulfilled, the fixation label is assigned. Windows with one to three fulfilled criteria are labeled based on their similarity to already-labeled windows.

Several machine learning approaches have been proposed as well. Vidal et al. (2012) focus solely on pursuit detection. They utilize shape features computed via a sliding window, to then use k -NN based classification. The reported accuracy of detecting pursuit is over 90%, but the diversity of the data set is clearly limited (only purely vertical and horizontal pursuit), and reporting accuracy without information about class balance is difficult to interpret.

Hoppe and Bulling (2016) propose using convolutional neural networks to assign eye-movement classes. Their approach, too, operates as a sliding window. For each window, the Fourier-transformed data are fed into a network, which contains one convolutional, one pooling, and one fully connected layer. The latter estimates the probabilities of the central sample in the window belonging to a fixation, a saccade, or a pursuit. The network used in this work is rather small, but the collected data seems diverse and promising. The reported scores are fairly low (average F1 score for detecting the three eye movements of 0.55), but without the availability of the test data set, it is impossible to assess the relative performance of this approach.

An approach by Anantrasirichai, Gilchrist, and Bull (2016) is to identify fixations in mobile eye tracking via an

SVM, while everything else is attributed to the class “saccades and noise”. The model is trained with gaze trace features, as well as image features locally extracted by a small 2D CNN. The approach is interesting, but the description of the data set and evaluation procedure lacks details.

A recently published work by Zemblys et al. (2017) uses Random Forests with features extracted in 100–200-ms windows that are centered around respective samples. It aims to detect fixations, saccades, and postsaccadic oscillations. This work also employs data augmentation to simulate various tracking frequencies and different levels of noise in data, which adds to the algorithm’s robustness.

Unfortunately, neither of these machine learning approaches are publicly available, at least in such a form that allows out-of-the-box usage (e.g., the implementation of Zemblys et al. (2017) lacks a pre-trained classifier).

All the algorithms so far process gaze traces individually. Agtzidis et al. (2016b) (toolbox available at Startsev et al., 2016) detect saccades and fixations in the same fashion, but use inter-observer similarities for the more challenging task of smooth pursuit detection. The samples remaining after saccade and fixation detection are clustered in the 3D space of time and spatial coordinates. Smooth pursuit by definition requires a target, so all pursuits on one scene can only be in a limited number of areas. Since no video information is used, inter-observer clustering of candidate gaze samples is used as a proxy for object detection: If many participants’ gaze traces follow similar paths, the chance of this being caused by tracking errors or noise is much lower. To take advantage of this effect, only those gaze samples that were part of some cluster are labeled as smooth pursuit. Those identified as outliers during clustering are marked as noise. This way, however, many pursuit-like events can be labeled as noise due to insufficient inter-observer similarity, and multiple recordings for each clip are required in order to achieve reliable results.

Our approach

We here propose using a bidirectional long short-term memory network (one-layer, in our case), which follows the processing of the input feature space by a series of 1D convolutional layers (see Fig. 1). Unlike (Hoppe & Bulling, 2016) and the vast majority of other literature approaches, we step away from single sample classification in favor of simultaneous window-based labeling, in order to capture temporal correlations in the data. Our network receives and outputs a window of samples-level features and labels, respectively. Unlike most of the methods in the literature, we also assign the “noise” label, which does not force our model to choose only between the meaningful classes, when this is not sensible.

Fig. 1 The architecture of our 1D CNN-BLSTM network. BN stands for “batch normalization”, FC – for “fully connected”. In this figure, the input is assumed to contain the five different-scale speed features, and the context window size that is available to the network is just above 1 s

Features for classification

We used both the raw x and y coordinates of the gaze and simple pre-computed features in our final models. To avoid overfitting for the absolute gaze location when using the xy -coordinates, we used positions relative to the first gaze location in each processed data window. Our initial experiments showed that a small architecture such as ours noticeably benefits from feature extraction on various temporal scales prior to passing the sequence to the model. This is especially prominent for smooth pursuit detection. With a limited number of small-kernel convolutional layers, network-extracted features are influenced only by a small area in the input-space data (i.e., the feature extracting sub-network has a small receptive field, seven samples, or 28 ms, in the case of our network). With this architecture it would thus be impossible to learn motion features on coarser temporal scales, which are important, e.g., for detecting the relatively persistent motion patterns, which characterize smooth pursuits. To overcome this, we decided to use precomputed features; specifically, we included speed, acceleration, and direction of gaze, all computed at five temporal scales in order to capture larger movement patterns on the feature level already.

The speed of gaze is an obvious and popular choice (Sauter et al., 1991; Salvucci & Goldberg, 2000; Komogortsev et al., 2010; Komogortsev & Karpov, 2013). Acceleration could aid saccade detection, as it is also sometimes used in the literature (Collewyn & Tamminga, 1984; Nyström & Holmqvist, 2010; Behrens, MacKeben, & Schröder-Preikschat, 2010; Larsson et al., 2013) as well as in SR Research’s software for the EyeLink trackers.

The effect of using the direction of gaze is less obvious: Horizontal smooth pursuit, for instance, is more natural to our visual system (Rottach, Zivotofsky, Das, Averbuch-Heller, Discenna, Poonyathalang, & Leigh, 1996). The drifts that occur due to tracking artifacts are, however, more pronounced along the vertical axis (Kyoung Ko, Snodderly, & Poletti, 2016).

We consider five different temporal scales for feature extraction: 4, 8, 16, 32, and 64 ms. The speed (in $^{\circ}/s$)

and direction (in radians, relative to the horizontal vector from left to right) of gaze for each sample were computed via calculating the displacement vector of gaze position on the screen from the beginning to the end of the temporal window of the respective size, centered around the sample. Acceleration (in $^{\circ}/s^2$) was computed from the speed values of the current and the preceding samples on the respective scale (i.e., numerical differentiation; acceleration for the first sample of each sequence is set to 0). If a sample was near a prolonged period of lost tracking or either end of a recording (i.e., if gaze data in a part of the temporal window was missing), a respectively shorter window was used.

We additionally conducted experiments on feature combinations, concatenating feature vectors of different groups, in order to further enhance performance.

Data sets

GazeCom

We used the GazeCom (Dorr et al., 2010; Startsev et al., 2016) recordings for both training and testing (manual annotation in Agtzigis, Startsev, & Dorr, 2016a), with a strict cross-validation procedure. This data set consists of 18 clips, around 20 s each, with an average of 47 observers per clip (total viewing time over 4.5 h). The total number of individual labels is about 4.3 million (including the samples still recorded after a video has finished; 72.5, 10.5, 11, and 5.9% of all samples are labeled as parts of fixations, saccades, pursuits, or noise, respectively). Event-wise, the data set contains 38629 fixations, 39217 saccades, and 4631 smooth pursuits. For training (but not for testing) on this data set, we excluded gaze samples with timestamps over 21 s (confidently outside video durations) and re-sampled to 250-Hz recordings of one of the observers (SSK), whose files had a higher sampling frequency.

We used leave-one-video-out (LOVO) cross-validation for evaluation: The training and testing is run 18 times, each time training on all the data for 17 videos and testing on all the eye-tracking data collected for the remaining video clip.

This way, the model that generates eye-movement labels on a certain video had never seen any examples of data with this clip during training. We aggregate the labeled outputs for the test sets of all splits before the final evaluation.

There are two major ways to fully utilize an eye-tracking data set in the cross-validation scenario, in the absence of a dedicated test subset. The first, LOVO, is described above, and it ensures that no video clip-specific information can benefit the model. The second would ensure that no observer-specific information would be used. For this, we used a leave-n-observers-out (LnOO) procedure. In our case, to maintain symmetry with the 18 splits of LOVO, we introduced the same number of splits in our data, each containing three unique randomly selected observers (54 participants in total).

We hypothesize that LOVO should be less susceptible to overfitting than LnOO, since smooth pursuit is target-driven, and the observers' scanpaths tend to cluster in space and time (Agtzidis et al., 2016b), meaning that their characteristics for different observers would be similar for the same stimulus. We test this hypothesis with several experiments, where the only varied parameter is the cross-validation type.

Nyström-Andersson data set

We used an independent set of recordings in order to additionally validate our model. For this, we took the manually annotated eye-tracking recordings that were utilized in a recent study (Andersson et al., 2017). These contain labels provided by two manual raters: One rater ("CoderMN") originally labeled the data for Larsson et al. (2013), another ("CoderRA") was added by Andersson et al. (2017). The annotations of both raters include six labeled classes: fixations, saccades, postsaccadic oscillations (PSOs), smooth pursuit, blinks, and undefined events.

The whole data set comprises three subsets that use moving dots, images, and video clips as stimuli. We focus our evaluation on the "video" part, since our model was trained on this domain.

We will refer to this subset by the abbreviations of the manual labelers' names (in chronological order of publications, containing respective data sets): "MN-RA-data". In total, it contains ca. 58000 gaze samples (or about 2 min at 500 Hz). Notably, only half of this data consists of "unique" samples, the second half being duplicated, but with different ground truth labels (provided by the second rater). 37.7% of all the samples were labeled as fixation, 5.3% as saccade, 3% as PSO, 52.2% as pursuit, 1.7% as blink, and 0.04% as "unknown". Counting events yields 163 fixations, 244 saccades, 186 PSOs, 121 pursuits, and 8 blinks. The high ratio of pursuit is explained by the explicit instructions given to the participants ("follow [...] moving

objects for video clips" Larsson et al. 2013) vs. free viewing in GazeCom (Dorr et al., 2010).

As in Andersson et al. (2017), we evaluated all the considered automatic approaches and both manual raters against the "average coder" (i.e., effectively duplicating each recording, but providing the "true" labels by MN in one case and by RA in the other).

Model architecture

We implemented a joint architecture that consists of a small one-dimensional temporal convolutional network and a bidirectional LSTM layer, with one time-distributed dense layer both before and after the BLSTM (for higher-level feature extraction and to match the number of classes in the output without limiting the number of neurons in the BLSTM, respectively). In this work, we implement multi-class classification with the following labels: fixation, saccade, smooth pursuit, noise (e.g., blinks or tracking loss), also "unknown" (for potentially using partially labeled data), in order to comply with the ground truth labeling scheme in Startsev et al. (2016). The latter label was absent in both the training data and the produced outputs. The architecture is also illustrated in Fig. 1 on an example of using a five-dimensional feature space and simultaneously predicting labels in a window equivalent to about 1 s of 250-Hz samples.

The network used here is reminiscent of other deep sequence-processing approaches, which combine either 2D (Donahue, Anne Hendricks, Guadarrama, Rohrbach, Venugopalan, Saenko, & Darrell, 2015) or, more recently, 3D (Molchanov, Yang, Gupta, Kim, Tyree, & Kautz, 2016) convolutions with recurrent units for frame sequence analysis. Since our task is more modest, our parameter count is relatively low (ca. 10000, depending on the input feature space dimensionality, compared to millions of parameters even in compact static CNNs (Hasanpour, Rouhani, Fayyaz, & Sabokrou, 2016), or ca. 6 million parameters only for the convolutional part (Tran, Bourdev, Fergus, Torresani, & Paluri, 2015) of Molchanov et al., 2016).

The convolutional part of our architecture contains three layers with a gradually decreasing number of filters (32, 16, and 8) with the kernel size of 3, and a batch normalization operation before activation. The subsequent fully connected layer contains 32 units. We did not use pooling, as is customary for CNNs, since we wanted to preserve the one-to-one correspondence between inputs and outputs. This part of the network is therefore not intended for high-level feature extraction, but prepares the inputs to be used by the BLSTM that follows.

All layers before the BLSTM, except for the very first one, are preceded by dropout (rate 0.3), and use ReLU as activation function. The BLSTM (with 16 units) uses *tanh*,

and the last dense layer (5 units, according to the number of classes) – softmax activation.

The input to our network is a window of a pre-set length, which we varied to determine the influence of context size on the detection quality for each class. To minimize the border effects, our network uses valid padding, requiring its input to be padded. For both training and testing, we only mirror-pad the whole sequence (of undetermined length; typically ca. 5000 in our data), and not each window. We pad by three samples on each side (since each of the three convolutional layers uses valid padding and a kernel of size 3). For our context-size experiments, this means that for a prediction window of 129 samples (i.e., classification context size 129), for example, windows of length 135 must be provided as input. So when we generate the labels for the whole sequence, the neighboring input windows overlap by six samples, but the output windows do not.

We balanced neither training nor test data in any way.

To allow for fair comparison of results with different context sizes, we attempted to keep the training procedure consistent across experiments. To this end, we fixed the number of windows (of any size) that are used for training (90%) and for validation (10%) to 50000. For experiments with context no larger than 65 samples, we used windows without overlap and randomly sampled (without replacement) the required number of windows for training. For larger-context experiments, the windows were extracted with a step of 65 samples (at 250 Hz, equivalent to 260 ms).

We initialized convolutional and fully connected layers with random weights from a normal distribution (mean 0, standard deviation 0.05), and trained the network for 1000 iterations with batch size 5000 with the RMSprop optimizer (Tieleman & Hinton, 2012) with default parameters from the Keras framework (Chollet et al., 2015) (version 2.1.1; learning rate 0.001 without decay) and categorical cross-entropy as the loss function.

Evaluation

Metrics

Similar to Agtzidis et al. (2016b), Startsev et al. (2016), and Hoppe and Bulling (2016), we evaluated *sample-level* detection results. Even though all our models (and most of the baseline models) treat eye-movement classification as a multi-class problem, for evaluation purposes we consider each eye movement in turn, treating its detection as a binary classification problem (e.g., with labels “fixation” and “not fixation”). This evaluation approach is commonly used in the literature (Larsson et al., 2013; Agtzidis et al., 2016b; Andersson et al., 2017; Hoppe & Bulling, 2016). We can then compute typical performance metrics such as precision, recall, and F1 score.

As for the *event-level* evaluation, there is no consensus in the literature as to which methodology should be employed. Hoppe and Bulling (2016), for example, use ground truth event boundaries as pre-segmentation of the sequences: For each event in the ground truth, all corresponding individual predicted sample labels are considered. The event is classified by the majority vote of these labels. As Hoppe and Bulling (2016) themselves point out, this pre-segmentation noticeably simplifies the problem of eye-movement classification. Additionally, the authors only reported confusion matrices and respective per-class hit rates, which conceal the problem of false detections. Andersson et al. (2017) only assess the detected events in terms of the similarity of event duration distribution parameters to those of the ground truth.

In Hooge, Niehorster, Nyström, Andersson, and Hessels (2017), *event-level* fixation detection was assessed by an arguably fairer approach with a set of metrics that includes F1 scores for fixation episodes. We computed these for all three main event types in our data (fixations, saccades, and smooth pursuits): For each event in the ground truth, we look for the earliest algorithmically detected event of the same class that intersects with it. Only one-to-one matching is allowed. Thus, each ground truth event can be labeled as either a “hit” (a matching detected event found) or a “miss” (no match found). The detected events that were not matched with any ground truth events are labeled as “false alarms”. These labels correspond to true positives, false negatives, and false positives, which are needed to compute the F1 score.

One drawback of such matching scheme is that the area of event intersection is not taken into account. This way, for a ground truth event E_{GT} , the earlier detected event E_A will be preferred to a later detected event E_B , even if the intersection with the former is just one sample, i.e., $|E_{GT} \cap E_A| = 1$, while the intersection with E_B is far greater. Hooge et al. (2017) additionally compute measures such as relative timing offset and deviation of matched events in order to tie together agreement measures and eye-movement parameters, which would also penalize potential poor matches. These, however, have to be computed for both onset and offset of each event type, and are more suited for in-detail analysis of particular labeling patterns rather than for a concise quantitative evaluation. We propose using a typical object detection measure (Everingham, Van Gool, Williams, Winn, & Zisserman, 2010; Everingham, Eslami, Van Gool, Williams, Winn, & Zisserman, 2015), the ratio of the two matched events’ intersection to their union (often referred to as “intersection over union”, or IoU). If a ground truth event is labeled as a “miss”, its corresponding “match IoU” is set to 0. This way, the average “match IoU” is influenced both by the number of detected and missed events, and by the quality of correctly identified events.

We report this statistic for all event types in the ground truth data, in addition to episode-level F1 scores of Hooge et al. (2017) as well as sample-level F1 scores (for brevity, F1 scores are used instead of individual statistics such as sensitivity or specificity; this metric represents a balanced combination of precision and recall—their harmonic mean), for both GazeCom and MN-RA-data.

Another idea, which we adapt from object detection research, is only registering a “hit” when a certain IoU threshold is reached (Ren, He, Girshick, and Sun, 2015), thus avoiding the low-intersection matches potentially skewing the evaluation. The threshold that is often employed is 0.5 (Everingham et al., 2010). In the case of one-dimensional events, this threshold also gains interpretability: This is the lowest threshold that ensures that no two detected events can be candidate matches for a single ground truth event. Additionally, if two events have the same duration, their relative shift can be no more than one-third of their duration. For GazeCom, we further evaluate the algorithms at different levels of the IoU threshold used for event matching.

We also compute basic statistics over the detected eye-movement episodes, which we compare to those of the ground truth. Among those are the average duration (in milliseconds) and amplitude (in degrees of visual angle) of all detected fixation, saccade, and smooth pursuit episodes. Even though fixations are not traditionally characterized by their amplitude, it reflects certain properties of fixation detection: For instance, where does a fixation end and a saccade begin? While this choice has relatively little bearing on saccade amplitudes, it might significantly affect the amplitudes of fixations.

Data set-specific settings

For MN-RA-data, we focused only on our best (according to GazeCom evaluation) model.

Since MN-RA-data were recorded at 500 Hz (compared to 250 Hz for GazeCom), we simply doubled the sample-level intervals for feature extraction, which preserves the temporal scales of the respective features (as described in the “Features for classification” above). We also used a model that classifies 257-sample windows at once (our largest-context model). This way, the temporal context at 500 Hz is approximately equivalent to that of 129 samples at 250 Hz, which was used for the majority of GazeCom experiments. Notably, the model used for MN-RA-data processing was trained on 250-Hz recordings and tested on the ones with double the sampling frequency. Our estimate of the model’s generalization ability is, therefore, conservative.

Due to cross-validation training on GazeCom, and in order to maximize the amount of pursuit examples in the

training data, we predict labels in MN-RA-data using a model trained on all GazeCom clips except one without smooth pursuit in its ground truth annotation (“bridge_1”).

Andersson et al. (2017) ignore smooth pursuit detection in most of their quantitative evaluation (while separating it from fixations is a challenging problem on its own), and focus on postsaccadic oscillations instead (which our algorithm does not label), so we cannot compare with the reported results directly. However, on the MN-RA-data, we additionally followed the evaluation strategies of Andersson et al. (2017).

In order to compare our approach to the state-of-the-art performances on MN-RA-data that were reported in Andersson et al. (2017), we computed the Cohen’s kappa statistic (for each major eye-movement class separately).

Cohen’s kappa κ for two binary sets of labels (e.g., A and B) can be computed via the observed proportion of samples, where A and B agree on either accepting or rejecting the considered eye-movement label, p_{obs} , and the chance probability of agreement. The latter can be expressed through the proportions of samples, where each of A and B has accepted and rejected the label. We denote those as p_+^A , p_-^A , p_+^B , and p_-^B , respectively. In this case,

$$p_{chance} = p_+^A * p_+^B + p_-^A * p_-^B, \quad (1)$$

$$\kappa(A, B) = \frac{p_{obs} - p_{chance}}{1 - p_{chance}}. \quad (2)$$

Cohen’s kappa can assume values from $[-1; 1]$, higher score is better. We also considered the overall sample-level error rate (i.e., proportion of samples where the models disagree with the human rater, when all six ground truth label classes are taken into account). For this, we consider all “noise” labels assigned by our algorithm as blink labels, as blinks were the primary cause of “noise” samples in the GazeCom ground truth. It has to be noted that all sample-level metrics are, to some extent, volatile with respect to small perturbations in the data—changes in proportions of class labels, almost imperceptible relative shifts, etc. We would, therefore, recommend using event-level scores instead.

Baselines

For both data sets, we ran several literature models as baselines, to give this work’s evaluation more context.

These were: Agtzidis et al. (2016b) (implementation by Startsev et al. (2016)), Larsson et al. (2015) (reimplemented by Startsev et al. (2016)), Dorr et al. (2010) (the authors’ implementation), Berg et al. (2009) (toolbox implementation Walther & Koch, 2006), I-VMP, I-VDT, I-VVT, I-KF, I-HMM, I-VT, I-MST, and I-DT (all as implemented by Komogortsev (2014), with fixation

interval merging enabled). For their brief descriptions, see “Automatic detection”.

Since several of the baselines (Dorr et al., 2010; Agtzidis et al., 2016b, the used implementation of Larsson et al., 2015) were either developed in connection with or optimized for GazeCom, we performed grid search optimization (with respect to the average of all sample- and event-wise F1 scores, as reported in Table 2) of the parameters of those algorithms in Komogortsev (2014) that detect smooth pursuit: I-VDT, I-VVT, and I-VMP. The ranges and the parameters of the grid search can be seen in Fig. 2. Overall, the best parameter set for I-VDT was $80^\circ/\text{s}$ for the speed threshold and 0.7° for the dispersion threshold. For I-VVT, the low speed threshold of $80^\circ/\text{s}$ and the high threshold of $90^\circ/\text{s}$ were chosen. For I-VMP, the high speed threshold parameter was fixed to the same value as in the best parameter combination of I-VVT ($90^\circ/\text{s}$), and the window duration and the “magnitude of motion” threshold were set to 400 ms and 0.6, respectively.

It is an interesting outcome that, when trying to optimize the scores, half of which depend on events, I-VVT abandons pursuit detection (by setting very high speed thresholds) in favor of better-quality saccade and fixation identification. If optimization with respect to sample-level scores only is performed, this behavior is not observed. This indicates that simple speed thresholding is not sufficient for reasonable pursuit episode segmentation. We have, therefore, tried different speed thresholds for I-VMP, but $90^\circ/\text{s}$ was still the best option.

We have to note that taking the best set of parameters selected via an exhaustive search on the entire test set is prone to overfitting, so the reported performance figures for these baseline methods should be treated as optimistic estimates.

Since the fixation detection step of Dorr et al. (2010) targeted avoiding smooth pursuit, we treat missing labels (as long as the respective samples were confidently tracked) as pursuit for this algorithm. We also adapted the parameters

of I-MST to the sampling frequency of the data set (for both data sets).

Just as Andersson et al. (2017), we did not re-train any of the models before testing them on the MN-RA-data.

For the model of Agtzidis et al. (2016a), however, we had to set the density threshold (*minPts*), which is a parameter of its clustering algorithm. This value should be set proportionally to the number of observers, and the sampling frequency (Startsev et al., 2016). If was, therefore, set to $160 * N_{observers} / 46.9 * 500 / 250$, where $N_{observers}$ is the number of recordings for each clip (i.e., four for “dolphin”, six for “BergoDalbana”, and eight for “triple-jump”). GazeCom has an average of 46.9 observers per clip, and is recorded at 250 Hz. MN-RA-data, as mentioned already, consists of recordings at 500 Hz. The resulting *minPts* values were 28, 40, and 54, respectively.

For both data sets, we additionally implemented a random baseline model, which assigns one of the three major eye-movement labels according to their frequency in the ground truth data.

Results and discussion

Cross-validation procedure selection

We first address the cross-validation type selection. We considered two modes, leave-one-video-out (LOVO) and leave-n-observers-out (LnOO, $n = 3$). If the two cross-validation procedures were identical in terms of the danger of overfitting, we would expect very similar quantitative results. If one is more prone to overfitting behavior than the other, its results would be consistently higher. In this part of the evaluation, therefore, we are, somewhat counterintuitively, looking for a validation procedure with *lower* scores.

We conducted several experiments to determine the influence of the cross-validation procedure on the performance

Fig. 2 Grid search average F1 scores on GazeCom for I-VDT (**2a**), I-VVT (**2b**), and I-VMP (**2c**). Default parameters were (ordered as (x, y) pairs): I-VDT – ($70^\circ/\text{s}$, 1.35°), I-VVT – ($20^\circ/\text{s}$, $70^\circ/\text{s}$), I-VMP – (0.5 s, 0.1). These are not optimal for our data set

Table 1 Experiment on the choice of a suitable cross-validation technique for our 1D CNN-BLSTM model with speed and direction features and a context size of 129 samples (equivalent to ca. 0.5 s at 250 Hz)

Metric	LOVO	LnOO
Fixations sample F1	0.937	0.939
Saccade sample F1	0.892	0.892
Pursuit sample F1	0.680	0.706
Fixations episode F1	0.888	0.887
Saccade episode F1	0.944	0.946
Pursuit episode F1	0.585	0.583
Fixations episode F1 ($IoU \geq 0.5$)	0.854	0.855
Saccade episode F1 ($IoU \geq 0.5$)	0.922	0.922
Pursuit episode F1 ($IoU \geq 0.5$)	0.456	0.466
Fixations episode IoU	0.902	0.905
Saccade episode IoU	0.858	0.857
Pursuit episode IoU	0.521	0.549

LOVO refers to the leave-one-video-out approach, LnOO – to leave-n-observers-out. Both methods split the data in 18 non-overlapping groups of recordings in our case (18 videos in the data set, 18 groups of three observers each). Differences no less than 0.01 are **boldified**. This suggests that LOVO provides a more conservative estimate, compared to LnOO

estimates (while keeping the rest of the training and testing parameters fixed), all of which revealed the same pattern: While being comparable in terms of fixation and saccade detection, these strategies differ consistently and noticeably for smooth pursuit detection (see the results of one of these experiments in Table 1).

LnOO tends to overestimate the performance of the models, yielding higher scores on most of the metrics. We conclude that LOVO is less prone to overfitting and conduct the rest of the experiments using this technique. We note that overfitting seems to affect the detection of the stimulus characteristics-dependent eye-movement type—smooth pursuit—the most. For stimuli that only induce fixations and saccades, the concern of choosing an appropriate cross-validation technique is alleviated.

We conclude that excluding the tested stimulus (video clip, in this case) must be preferred to excluding the tested observer(s), if some form of cross-validation has to be employed, especially if the evaluation involves highly stimulus-driven eye-movement classes (e.g., smooth pursuit).

GazeCom results overview

An overview of all the evaluation results on the full GazeCom data set is presented in Table 2. It reports the models' performance on the three eye movement classes (fixations, saccades, and pursuits) for both sample- and

event-level detection. Table 3 additionally provides the IoU values for all the tested algorithms. Bold numbers mark best performance in each category.

Most of our BLSTM models were trained with the context window of 129 samples (ca. 0.5 s) at the output layer, as it presented a reasonable trade-off between training time (ca. 3 s per epoch on NVIDIA 1080Ti GPU) and the saturation of the effect that context size had on performance.

Individual feature groups

Looking at individual feature sets for our model (raw xy -coordinates, speed, direction, and acceleration), we find that speed is the best individual predictor of eye-movement classes.

Acceleration alone, not surprisingly, fails to differentiate between fixations and smooth pursuit (the largest parts of almost 90% of the smooth pursuit episodes are covered by fixation labels), since both perfect fixation and perfect pursuit lack the acceleration component, excepting onset and offset stages of pursuits. Saccade detection performance is, however, impressive.

Interestingly, direction of movement provides a decent feature for eye-movement classification. One would expect that within fixations, gaze trace directions are distributed almost uniformly because of (isotropic) oculomotor and tracker noise. Within pursuits, its distribution should be pronouncedly peaked, corresponding to the direction of the pursuit, and even more so within saccades due to their much higher speeds. Figure 3 plots these distributions of directional features for each major eye-movement type. The directions were computed at a fixed temporal scale of 16 ms and normalized per-episode so that the overall direction is at 0. Unlike perfect fixations, which would be completely stationary, fixations in our data set contain small drifts (mean displacement amplitude during fixation of 0.56° of visual angle, median 0.45°), so the distribution in Fig. 3 is not uniform. Gaze direction features during saccades and pursuits predictably yield much narrower distribution shapes.

Using just the xy coordinates of gaze has an advantage of its simplicity and the absence of any pre-processing. However, according to our evaluation, the models that use either speed or direction features instead generally perform better, especially for smooth pursuit detection. Nevertheless, our model without any feature extraction already outperforms the vast majority of the literature approaches.

Feature combinations

Experimenting with several feature sets at once, we found that acceleration as an additional feature did not improve average detection performance, probably due to its inability to distinguish pursuit from fixation. The combination

Table 2 GazeCom evaluation results as F1 scores for *sample-level* and *episode-level* detection (sorted by the average of all columns)

Model	average F1	Sample-level F1			Event-level F1		
		Fixation	Saccade	SP	Fixation	Saccade	SP
<i>1D CNN-BLSTM: speed + direction</i> ⁺	0.830	0.939	0.893	0.703	0.898	0.947	0.596
<i>1D CNN-BLSTM: speed + direction</i>	0.821	0.937	0.892	0.680	0.888	0.944	0.585
<i>1D CNN-BLSTM: speed</i>	0.808	0.932	0.891	0.675	0.877	0.942	0.529
(Agtzidis et al., 2016b)	0.769	0.886	0.864	0.646	0.810	0.884	0.527
<i>1D CNN-BLSTM: direction</i>	0.769	0.919	0.802	0.621	0.862	0.911	0.499
<i>1D CNN-BLSTM: xy</i>	0.752	0.913	0.855	0.517	0.861	0.932	0.435
(Larsson et al., 2015)	0.730	0.912	0.861	0.459	0.873	0.884	0.392
I-VMP**	0.718	0.909	0.680	0.581	0.792	0.815	0.531
(Berg et al., 2009)	0.695	0.883	0.697	0.422	0.886	0.856	0.424
(Dorr et al., 2010)	0.680	0.919	0.829*	0.381	0.902	0.854	0.193*
<i>1D CNN-BLSTM: acceleration</i>	0.668	0.904	0.876	0.160	0.877	0.943	0.245
I-VDT**	0.606	0.882	0.676	0.321	0.823	0.781	0.152
I-KF	0.563	0.892	0.736	–	0.877	0.876	–
I-HMM	0.546	0.891	0.712	–	0.817	0.857	–
I-VVT**	0.531	0.890	0.686	0.000	0.778	0.816	0.013
I-VT	0.528	0.891	0.705	–	0.761	0.810	–
I-MST	0.497	0.875	0.570	–	0.767	0.773	–
I-DT	0.480	0.877	0.478	–	0.759	0.765	–
Random Baseline	0.201	0.750	0.105	0.114	0.098	0.121	0.020

CNN-BLSTM results are for the context window size of just over 0.5 s (129 samples), except where marked with ⁺ (1 s, 257 samples). The * signs mark the numbers where the label was assumed from context and not actually assigned by the algorithm – i.e. missing labels were imputed. Performance estimates for models marked with ** are optimistic (thresholds were optimized on the entire test set). In each column, the highest value is **boldified**

of direction and speed, however, showed a noticeable improvement over using them separately, and the results for these features we present in the tables.

We retrained the model that uses direction and speed features for a larger context size (257 samples, ca. 1 s). This model demonstrates the highest F1 scores (or within half a percent of the best score achieved by any model) for all eye-movement types in both evaluation settings. It

outperforms the nearest literature approach by 2, 2.9, and 5.7% of the F1 score for fixations, saccades, and smooth pursuits, respectively. The gap is even wider (6.3 and 6.5% for saccades and smooth pursuit, respectively) for episode-level evaluation. Only for fixation episode detection, the Dorr et al. (2010) model performs slightly better (by 0.004). In terms of IoU values, our model improves the state-of-the-art scores by 0.04, 0.05, and 0.09 for fixations, saccades,

Fig. 3 Histogram of sample-level direction features, when normalized relative to the overall direction of each respective episode

Table 3 GazeCom evaluation results for *event-level* detection as intersection-over-union values (sorted by the average of all columns)

Model	Average IoU	Fixation IoU	Saccade IoU	SP IoU
<i>ID CNN-BLSTM: speed + direction</i> ⁺	0.768	0.906	0.858	0.541
<i>ID CNN-BLSTM: speed</i>	0.763	0.885	0.856	0.547
<i>ID CNN-BLSTM: speed + direction</i>	0.760	0.902	0.858	0.521
<i>ID CNN-BLSTM: xy</i>	0.665	0.880	0.801	0.313
(Dorr et al., 2010)	0.663	0.815	0.808	0.367*
(Agtzidis et al., 2016b)	0.663	0.742	0.799	0.448
<i>ID CNN-BLSTM: direction</i>	0.631	0.834	0.718	0.341
(Larsson et al., 2015)	0.625	0.789	0.809	0.277
I-VMP**	0.613	0.828	0.556	0.454
<i>ID CNN-BLSTM: acceleration</i>	0.606	0.906	0.834	0.077
I-VDT**	0.558	0.760	0.555	0.359
(Berg et al., 2009)	0.541	0.774	0.499	0.351
I-KF	0.504	0.842	0.671	–
I-HMM	0.501	0.870	0.633	–
I-VT	0.492	0.868	0.607	–
I-VVT**	0.477	0.863	0.567	0.000
I-MST	0.364	0.694	0.399	–
I-DT	0.313	0.592	0.347	–
Random baseline	0.055	0.077	0.071	0.017

CNN-BLSTM results are for the context window size of just over 0.5 s (129 samples), except where marked with ⁺ (1 s, 257 samples). The * signs mark the numbers where the label was assumed from context and not actually assigned by the algorithm – i.e. missing labels were imputed. Performance estimates for models marked with ** are optimistic (thresholds were optimized on the entire test set). In each column, the highest value is **boldified**

and pursuits, respectively, indicating the higher quality of the detected episodes across the board.

We also varied the IoU threshold that determines whether two episodes constitute a potential match, computing episode-level F1 scores every time (see Fig. 4). From this evaluation it can be seen that not only does our deep learning model outperform all literature models, but it maintains this advantage even when a stricter criterion for an event “hit” is considered (even though it was trained to optimize pure sample-level metrics). For fixations, while similar to the performance of Dorr et al. (2010) at lower IoU thresholds, our model is clearly better when it comes to higher thresholds. For saccades, it has to be noted that the labels of Dorr et al. (2010) were used as initialization for the manual annotators in order to obtain ground truth event labels for GazeCom. This results in a higher number of perfectly matching saccade episodes for Dorr et al. (2010) (as well as for Agtzidis et al. (2016b) and our implementation of Larsson et al. (2015), both of which use a very similar saccade detection procedure), when the manual raters decided not to change the borders of certain saccades.

As mentioned already, a threshold of 0.5 has its theoretical benefits (no two detected episodes can both be matches for a single ground truth event, some interpretability). Here,

we can see practical advantages as well, thanks to the *Random Baseline* model. Due to the prevalence of fixation samples in the GazeCom data set, assigning random labels with the same distribution of classes results in many fixation events, which occasionally intersect with fixations in the ground truth. In the absence of any IoU thresholding (the threshold of 0 in Fig. 4), the F1 scores for fixations and saccades are around 10%. Only by the threshold level of 0.5 does the fixation event-wise F1 score for the *Random Baseline* reach zero.

Common eye-tracking measures

In order to directly compare the average properties of the detected events to those in the ground truth, we compute the mean durations and amplitudes for the episodes of the three major eye-movement types in our data. The results are presented in Table 4. For this part of the evaluation, we consider only our best model (referred to as *ID CNN-BLSTM (best)* in the tables), which uses speed and direction features at a context size of roughly 1 s. We compare it to all the baseline algorithms that consider smooth pursuit.

For both measures, our algorithm is ranked second, while providing average fixation and saccade amplitudes that are

Table 4 Average durations (in milliseconds) and amplitudes (in degrees of visual angle) of different event types, as labelled by manual annotators (first row) or algorithmic procedures

Algorithm	Average event duration, ms				Average event amplitude, degrees			
	Fixation	Saccade	SP	(rank) avg. Δ	Fixation	Saccade	SP	(rank) avg. Δ
Ground truth	315.2	41.5	405.2		0.56	6.84	2.38	
<i>ID CNN-BLSTM (best)</i>	281.1	38.4	217.0	(2) 75.2	0.53	6.66	1.44	(2) 0.38
(Agtzidis et al., 2016b)	229.5	40.1	244.6	(3) 82.6	0.40	7.19	1.91	(1) 0.33
(Larsson et al., 2015)	335.3	41.0	320.1	(1) 35.2	0.70	7.45	3.15	(3) 0.51
I-VMP**	256.3	20.8	217.8	(4) 89.0	0.52	6.03	1.64	(4) 0.53
(Berg et al., 2009)	282.6	66.0	164.3	(5) 99.3	0.51	8.26	1.20	(6) 0.88
(Dorr et al., 2010)	340.3	41.7	68.2*	(6) 120.7	0.53	7.41	1.09*	(5) 0.63
I-VDT**	284.7	19.0	45.7	(7) 137.5	0.47	5.41	0.49	(8) 1.13
I-VVT**	261.1	21.4	0.5	(8) 159.7	0.64	6.11	0.04	(7) 1.05

In each column, the value with the lowest absolute difference to the respective ground truth value is **boldified**. The averages of these absolute differences for event durations and amplitudes occupy the fifth and the last columns, respectively, along with the rank of each considered model (lower is better). The rows are sorted as the respective rows of our main evaluation table – Table 2. The * signs mark the numbers where the label was assumed from context and not actually assigned by the algorithm – i.e. missing labels were imputed. Performance estimates for models marked with ** are optimistic (thresholds were optimized on the entire test set)

the closest to the ground truth. We note that the approaches with the average duration and amplitude of events closest to the ground truth differ for the two measures (Larsson et al., 2015; Agtzidis et al., 2016b, respectively).

From this evaluation, we can conclude that our algorithm detects many small smooth pursuit episodes, resulting in comparatively low average smooth pursuit duration and amplitude. This is confirmed by the relatively higher event-level false positive count of our algorithm (3475, compared to 2207 for Larsson et al., 2015). Our model's drastically lower false negatives count (1192 vs. 2966), however, allows it to achieve much higher F1 score for pursuit event detection.

We also have to stress that simple averages do not provide a comprehensive insight into the performance of an eye-movement detection algorithm, but rather offer an intuitively interpretable, though not entirely reliable, measure of detected event quality. There is no matching performed here, the entire set of episodes of a certain type is averaged at once. This is why we recommend using either the temporal offsets of matched episode pairs as introduced by Hooge et al. (2017), or IoU averaging or thresholding, as we suggest in “Metrics”. The latter allows for evaluating episode-level eye-movement detection performance at varying levels of match quality, which is assessed via a relatively easily interpretable IoU metric.

Context size matters

We also investigated the influence of the size of the context, where the network simultaneously assigns labels, on detection scores (see Fig. 5). We did this by running the

cross-validation process at a range of context sizes, with five speed features defining the input space. We tested contexts of 1, 2 + 1, 4 + 1, 8 + 1, ..., 256 + 1 samples. For the GazeCom sampling frequency, this corresponds to 4, 12, 20, 36 ms, ..., 1028 ms. Training for larger context sizes was computationally impractical.

Context size had the biggest influence on smooth pursuit detection. For speed features, when the context window size was reduced from just over 1 s of gaze data to merely one sample being classified at a time, the F1 scores for fixation and saccade samples decreased (in terms of absolute values) only by 2.8 and 5.1%, respectively, whereas smooth pursuit sample detection performance plummeted (decreased by over 40%).

For all eye movements, however, there is a general positive impact of expanding the context of the analysis. This effect seems to reach saturation point by the 1 s mark, with absolute improvements in all detection F1 scores being not much higher than 1% (except for smooth pursuit episodes, which could potentially benefit from even larger context sizes).

At the largest context size, this model is better at detecting smooth pursuit (for both sample- and event-level problem settings) than any baseline smooth pursuit detector, including the multi-observer approach in Agtzidis et al. (2016b), which uses information from up to 50 observers at the same time, allowing for higher-level heuristics.

Generalizability

To test our model on additional independent data, we present the evaluation results of our best model (speed and

direction used as features) with the context size ca. 0.5 s and all the literature models we tested on MN-RA-data set as sample- and event-level F1 scores (Table 5), as well as average IoU values (Table 6). This is the model with the largest context we trained, 257 samples. The duration in seconds is reduced due the doubling of the sampling frequency, compared to GazeCom.

Table 7 combines our evaluation results with the performances reported in Andersson et al. (2017) in the form of Cohen's kappa values and overall error rates for the MN-RA-data (for video stimuli). Evaluation results from Andersson et al. (2017) were included in the table if they represent the best performance with respect to at least one of the statistics that we include in this table. BIT refers to the algorithm in van der Lans, Wedel, and Pieters (2011), LNS—in Larsson et al. (2013).

For our model, performance on this data set is worse compared to GazeCom, but even human raters show

substantial differences in labeling the “ground truth” (Hooge et al., 2017; Andersson et al., 2017).

Nevertheless, the average out-of-the-box performance of our algorithm compares favorably to the state of the art in terms of sample-level classification (see Table 5). In terms of episode-level evaluation, our model shows somewhat competitive F1 scores (Table 5), but makes up for it in the average intersection over union statistic, which accounts for both the number of correctly identified episodes and the quality of the match (see Table 6). While its error rate is similar to that of I-VMP, the F1 and IoU scores are, on average, higher for our model, and its Cohen's κ scores are consistently superior to I-VMP across the board.

Our algorithm's 34% error rate may still be unacceptable for many applications, but so could be the manual rater disagreement of 19% as well. Our algorithm further demonstrates the highest Cohen's Kappa score for smooth pursuit detection, and second highest for fixation detection.

Fig. 4 Episode-level F1 scores at different IoU thresholds: At 0.0, the regular episode F1 score is computed; At 1.0, the episodes have to match exactly; The thresholds in-between represent increasing

levels of episode match strictness. The vertical dashed line marks the threshold, which is typically used when considering IoU scores

Fig. 5 Detection quality plotted against the context size (in samples at 250 Hz; log-scale) that the network classifies at once. *Dashed lines* represent individually chosen reference algorithms that perform best with respect to each eye movement. For both sample- and event-level F1 evaluation (**5a** and **5b**, respectively), fixation detection results of Dorr et al. (2010) are taken, for saccades – Startsev et al. (2016), for

pursuits – I-VMP. For event-level IoU evaluation (**5c**), “best other” fixation detection IoU is taken from I-HMM, for saccades – Larsson et al. (2015), for pursuits – I-VMP. We separately display the smooth pursuit detection results of the multi-observer algorithm’s toolbox (Startsev et al., 2016) (the dotted line), as it belongs to a different class of algorithms

The best saccade detection quality is achieved by LNS, which explicitly labels postsaccadic oscillations and thus increases saccade detection specificity.

For sample-level F1-score evaluation (Table 5), our model achieves second highest scores for fixation (with a very narrow margin) and pursuit detection, outperforming all competition in saccade detection.

Conclusions

We have proposed a deep learning system for eye-movement classification. Its overall performance surpasses all considered reference models on an independent small-scale data set. For the naturalistic larger-scale GazeCom, our approach outperforms the state of the art with respect to

the three major eye-movement classes: fixations, saccades, and smooth pursuits. To the best of our knowledge, this is the first inherently temporal machine learning model for eye-movement event classification that includes smooth pursuit. Unlike (Agtzidis et al., 2016b), which implicitly uses full temporal context, and explicitly combines information across a multitude of observers, our model can be adapted for online detection (by re-training without using look-ahead features and preserving the LSTM states⁴). The classification time is kept short due to the low—for deep-learning models, at least—parameter count of the trained models. Furthermore, we introduced and analyzed a new

⁴For online detection, one would need to either use a unidirectional LSTM and process the samples as they are recorded, or assemble windows of samples that end with the latest available ones and process the full windows with a BLSTM model.

Table 5 MN-RA-data evaluation results as F1 scores for *sample-level* and *episode-level* detection (sorted by the average of all columns). CNN-BLSTM results here are for the window size of just over 0.5s (257 samples at 500 Hz). The * signs mark the numbers where the label was assumed from context and not actually assigned by the algorithm – i.e. missing labels were imputed. In each column, the highest value is **boldified**

Model	average F1	Sample-level F1			Event-level F1		
		Fixation	Saccade	SP	Fixation	Saccade	SP
(Agtzidis et al., 2016b)	0.653	0.670	0.699	0.638	0.455	0.860	0.592
<i>ID CNN-BLSTM: speed + direction</i>	0.650	0.667	0.720	0.663	0.550	0.826	0.475
(Larsson et al., 2015)	0.633	0.609	0.698	0.424	0.741	0.871	0.456
(Dorr et al., 2010)	0.630	0.614	0.691	0.446*	0.710	0.841	0.476*
I-VMP	0.572	0.593	0.699	0.739	0.455	0.564	0.385
(Berg et al., 2009)	0.533	0.609	0.625	0.176	0.683	0.730	0.374
I-VDT	0.474	0.595	0.694	0.222	0.443	0.561	0.329
I-KF	0.444	0.578	0.643	–	0.639	0.806	–
I-HMM	0.421	0.577	0.711	–	0.535	0.702	–
I-DT	0.381	0.573	0.439	–	0.599	0.678	–
I-VT	0.375	0.575	0.701	–	0.412	0.560	–
I-VVT	0.365	0.573	0.701	0.242	0.067	0.560	0.046
I-MST	0.363	0.560	0.444	–	0.603	0.568	–
Random Baseline	0.180	0.387	0.051	0.535	0.023	0.066	0.017

Table 6 MN-RA-data evaluation results for *event-level* detection as intersection-over-union values (sorted by the average of all columns)

Model	Fixation ep. IoU	Saccade ep. IoU	SP ep. IoU
<i>ID CNN-BLSTM: speed + direction</i>	0.705	0.543	0.398
I-VMP	0.368	0.623	0.619
I-VDT	0.744	0.647	0.127
(Agtzidis et al., 2016b)	0.626	0.469	0.420
(Larsson et al., 2015)	0.754	0.514	0.215
I-VT	0.798	0.665	–
I-HMM	0.81	0.627	–
(Dorr et al., 2010)	0.646	0.461	0.284*
I-KF	0.785	0.543	–
(Berg et al., 2009)	0.706	0.353	0.075
I-DT	0.628	0.328	–
I-VVT	0.181	0.665	0.023
I-MST	0.438	0.208	–
Random baseline	0.024	0.038	0.016

BLSTM results here are for the window size of just over 0.5s (257 samples at 500 Hz). The * signs mark the numbers where the label was assumed from context and not actually assigned by the algorithm – i.e., missing labels were imputed. In each column, the highest value is **boldified**

Table 7 Cohen's kappa (higher is better) and overall error rates (lower is better) for the MN-RA-data set

Group	Error rate	Fixation κ	Saccade κ	SP κ
CoderMN	19%	0.83	0.94	0.83
CoderRA	19%	0.82	0.94	0.83
<i>ID CNN-BLSTM: speed + direction</i>	34%	0.41	0.70	0.41
I-VMP	34%	0.38	0.68	0.40
(Agtzidis et al., 2016b)	38%	0.43	0.68	0.40
(Dorr et al., 2010)	46%	0.25	0.67	0.20*
(Larsson et al., 2015)	47%	0.23	0.68	0.19
I-VDT	53%	0.16	0.67	0.09
Random Baseline	56%	0.00	0.00	0.00
(Berg et al., 2009)	57%	0.21	0.60	0.07
I-VVT	55%	0.14	0.68	0.02
I-HMM**	59%	0.13	0.71	—
I-VT**	59%	0.13	0.76	—
I-DT	60%	0.09	0.40	—
I-MST	61%	0.04	0.43	—
I-KF**	62%	0.14	0.59	—
BIT**	67%	0.14	0.00	—
LNS**	92%	0.00	0.81	—

BLSTM here uses speed and direction features and the context of ca. 0.5 s (257 samples at 500 Hz). The * signs mark the numbers where the label was assumed from context and not actually assigned by the algorithm—i.e., missing labels were imputed. The scores for the algorithms marked with the ** were taken directly from Andersson et al. (2017). The rows are sorted by their error rate. In each column, the best value achieved by any algorithm is **boldified** (the first two rows correspond to manual annotators)

event-level evaluation protocol that considers the quality of the matched episodes through enforcing restrictions on the pair of events that constitute a match. Our experiments additionally highlight the importance of temporal context, especially for detecting smooth pursuit.

The code for our model and results for all evaluated algorithms are provided at <http://www.michaeldorr.de/smoothpursuit>.

Acknowledgements This research was supported by the Elite Network Bavaria, funded by the Bavarian State Ministry for Research and Education.

References

- Agtzidis, I., Startsev, M., & Dorr, M. (2016a). 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)* (pp. 65–68).
- Agtzidis, I., Startsev, M., & Dorr, M. (2016b). Smooth pursuit detection based on multiple observers. In *Proceedings of the ninth biennial ACM symposium on eye tracking research & applications, ETRA '16* (pp. 303–306). New York: ACM.
- Anantrasirichai, N., Gilchrist, I. D., & Bull, D. R. (2016). Fixation identification for low-sample-rate mobile eye trackers. In *2016 IEEE international conference on image processing (ICIP)* (pp. 3126–3130).
- Andersson, R., Larsson, L., Holmqvist, K., Stridh, M., & Nyström, M. (2017). One algorithm to rule them all? An evaluation and discussion of ten eye movement event-detection algorithms. *Behavior Research Methods*, 49(2), 616–637.
- Behrens, F., MacKeben, M., & Schröder-Preikschat, W. (2010). An improved algorithm for automatic detection of saccades in eye movement data and for calculating saccade parameters. *Behavior Research Methods*, 42(3), 701–708.
- Berg, D. J., Boehnke, S. E., Marino, R. A., Munoz, D. P., & Itti, L. (2009). Free viewing of dynamic stimuli by humans and monkeys. *Journal of Vision*, 9(5), 1–15.
- Chollet, F., et al. (2015). Keras. <https://github.com/keras-team/keras>
- Collewyn, H., & Tamminga, E. P. (1984). Human eye movements during voluntary pursuit of different target motions on different backgrounds. *The Journal of Physiology*, 351(1), 217–250.
- Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., & Darrell, T. (2015). Long-term recurrent convolutional networks for visual recognition and description. In *The IEEE conference on computer vision and pattern recognition (CVPR)*.
- Dorr, M., Martinetz, T., Gegenfurtner, K. R., & Barth, E. (2010). Variability of eye movements when viewing dynamic natural scenes. *Journal of Vision*, 10(10), 28–28.
- Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2), 303–338.
- Everingham, M., Eslami, S. M. A., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2015). The Pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1), 98–136.

- Goldberg, J. H., & Schryver, J. C. (1995). Eye-gaze-contingent control of the computer interface: Methodology and example for zoom detection. *Behavior Research Methods Instruments, & Computers*, 27(3), 338–350.
- Hasanpour, S. H., Rouhani, M., Fayyaz, M., & Sabokrou, M. (2016). Lets keep it simple, using simple architectures to outperform deeper and more complex architectures. CoRR, arXiv:1608.06037
- Hooge, I. T. C., Niehorster, D. C., Nyström, M., Andersson, R., & Hessels, R. S. (2017). Is human classification by experienced untrained observers a gold standard in fixation detection? *Behavior Research Methods*.
- Hoppe, S., & Bulling, A. (2016). End-to-end eye movement detection using convolutional neural networks. ArXiv e-prints.
- Komogortsev, O. V. (2014). Eye movement classification software. http://cs.txstate.edu/ok11/emd_offline.html
- Komogortsev, O. V., Gobert, D. V., Jayarathna, S., Koh, D. H., & Gowda, S. M. (2010). Standardization of automated analyses of oculomotor fixation and saccadic behaviors. *IEEE Transactions on Biomedical Engineering*, 57(11), 2635–2645.
- Komogortsev, O. V., & Karpov, A. (2013). Automated classification and scoring of smooth pursuit eye movements in the presence of fixations and saccades. *Behavior Research Methods*, 45(1), 203–215.
- Kyoung Ko, H., Snodderly, D. M., & Poletti, M. (2016). Eye movements between saccades: Measuring ocular drift and tremor. *Vision Research*, 122, 93–104.
- Land, M. F. (2006). Eye movements and the control of actions in everyday life. *Progress in Retinal and Eye Research*, 25(3), 296–324.
- Larsson, L., Nyström, M., Andersson, R., & Stridh, M. (2015). Detection of fixations and smooth pursuit movements in high-speed eye-tracking data. *Biomedical Signal Processing and Control*, 18, 145–152.
- Larsson, L., Nyström, M., & Stridh, M. (2013). Detection of saccades and postsaccadic oscillations in the presence of smooth pursuit. *IEEE Transactions on Biomedical Engineering*, 60(9), 2484–2493.
- Molchanov, P., Yang, X., Gupta, S., Kim, K., Tyree, S., & Kautz, J. (2016). Online detection and classification of dynamic hand gestures with recurrent 3D convolutional neural network. In *The IEEE conference on computer vision and pattern recognition (CVPR)*.
- Nyström, M. (2015). Marcus Nyström — Humanities Lab, Lund University. <http://www.humlab.lu.se/en/person/MarcusNystrom>
- Nyström, M., & Holmqvist, K. (2010). An adaptive algorithm for fixation, saccade, and glissade detection in eyetracking data. *Behavior Research Methods*, 42(1), 188–204.
- Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., & Garnett, R. (Eds.) *Advances in neural information processing systems* 28 (pp. 91–99). Curran Associates, Inc.
- Rottach, K. G., Zivotofsky, A. Z., Das, V. E., Averbuch-Heller, L., Discenna, A. O., Poonyathalang, A., & Leigh, R. J. (1996). Comparison of horizontal, vertical and diagonal smooth pursuit eye movements in normal human subjects. *Vision Research*, 36(14), 2189–2195.
- Salvucci, D. D., & Anderson, J. R. (1998). Tracing eye movement protocols with cognitive process models. In *Proceedings of the 20th annual conference of the cognitive science society* (pp. 923–928). Lawrence Erlbaum Associates Inc.
- Salvucci, D. D., & Goldberg, J. H. (2000). Identifying fixations and saccades in eye-tracking protocols. In *Proceedings of the 2000 symposium on eye tracking research & applications, ETRA '00* (pp. 71–78). New York: ACM.
- San Agustin, J. (2010). *Off-the-shelf gaze interaction*. PhD thesis, IT-Universitetet i København.
- Santini, T. (2016). Automatic identification of eye movements. <http://ti.uni-tuebingen.de/Eye-Movements-Identification.1845.0.html>
- Santini, T., Fuhl, W., Kübler, T., & Kasneci, E. (2016). Bayesian identification of fixations, saccades, and smooth pursuits. In *Proceedings of the ninth biennial ACM symposium on eye tracking research & applications, ETRA '16* (pp. 163–170). New York: ACM.
- Sauter, D., Martin, B. J., Di Renzo, N., & Vomscheid, C. (1991). Analysis of eye tracking movements using innovations generated by a Kalman filter. *Medical and Biological Engineering and Computing*, 29(1), 63–69.
- Startsev, M., Agtzidis, I., & Dorr, M. (2016). Smooth pursuit. <http://michaeldorr.de/smoothpursuit/>
- Tieleman, T., & Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURS-ERA: Neural Networks for Machine Learning*, 4(2), 26–31.
- Tran, D., Bourdev, L., Fergus, R., Torresani, L., & Paluri, M. (2015). Learning spatiotemporal features with 3D convolutional networks. In *2015 IEEE international conference on computer vision (ICCV)* (pp. 4489–4497).
- van der Lans, R., Wedel, M., & Pieters, R. (2011). Defining eye-fixation sequences across individuals and tasks: The binocular-individual threshold (BIT) algorithm. *Behavior Research Methods*, 43(1), 239–257.
- Vidal, M., Bulling, A., & Gellersen, H. (2012). Detection of smooth pursuits using eye movement shape features. In *Proceedings of the symposium on eye tracking research & applications, ETRA '12* (pp. 177–180). New York: ACM.
- Walther, D., & Koch, C. (2006). Modeling attention to salient proto-objects. *Neural Networks*, 19(9), 1395–1407.
- Yarbus, A. L. (1967). *Eye movements during perception of moving objects* (pp. 159–170). Boston: Springer.
- Zembylas, R., Niehorster, D. C., Komogortsev, O., & Holmqvist, K. (2017). Using machine learning to detect events in eye-tracking data. *Behavior Research Methods*, 50(1), 160–181. <https://doi.org/10.3758/s13428-017-0860-3>