Behavior Research Methods
https://doi.org/10.3758/s13428-022-01978-2

Online eye-movement classification with temporal convolutional networks

Carlos Elmadjian[1] · Candy Gonzales[1] · Rodrigo Lima da Costa[1] · Carlos H. Morimoto[2]

Accepted: 8 September 2022

Abstract
The simultaneous classification of the three most basic eye-movement patterns is known as the ternary eye-movement classification problem (3EMCP). Dynamic, interactive real-time applications that must instantly adjust or respond to certain eye behaviors would highly benefit from accurate, robust, fast, and low-latency classification methods. Recent developments based on 1D-CNN-BiLSTM and TCN architectures have demonstrated to be more accurate and robust than previous solutions, but solely considering offline applications. In this paper, we propose a TCN classifier for the 3EMCP, adapted to online applications, that does not require look-ahead buffers. We introduce a new lightweight preprocessing technique that allows the TCN to make real-time predictions at about 500 Hz with low latency using commodity hardware. We evaluate the TCN performance against other two deep neural models: a CNN-LSTM and a CNN-BiLSTM, also adapted to online classification. Furthermore, we compare the performance of the deep neural models against a lightweight real-time Bayesian classifier (I-BDT). Our results, considering two publicly available datasets, show that the proposed TCN model consistently outperforms other methods for all classes. The results also show that, though it is possible to achieve reasonable accuracy levels with zero-length look ahead, the performance of all methods improve with the use of look-ahead information. The codebase, pre-trained models, and datasets are available at `https://github.com/elmadjian/OEMC`.

Keywords Eye-movement patterns · Online classification · Convolutional neural networks

Introduction

The recognition of eye-movement patterns is an important step within several application domains, as they provide meaningful information about the subject's behavior and attentional state. On a higher level, it can provide insights about the user's cognitive state (Sanches et al., 2017; Burch et al., 2019) or degree of attention (Maruyama et al., 2016; Wang & Hung, 2019), and on more practical scenarios, support applications in various domains, such as biometrics (George & Routray, 2016; Bayat & Pomplun, 2017; Abdrabou et al., 2021), text input (MacKenzie & Zhang, 2008; Tula & Morimoto, 2016; Feng et al., 2021), accessibility (Koochaki & Najafizadeh, 2018), and adaptive systems (Edwards, 1998; de Greef et al., 2009), among many others.

Eye movements are typically captured using video-based eye-tracking devices (called eye trackers) (Morimoto & Mimica, 2005). The point-of-gaze on a computer monitor or other surface of interest might be computed using a calibration procedure (Morimoto et al., 2020). Eye trackers have been successfully used as an input device for computer interfaces, for example, to help people with motor disabilities to communicate (Majaranta & Bulling, 2014). More robust real-time methods for the classification of eye-movement patterns might improve the performance of current gaze interaction systems and help create

This research was funded by the São Paulo Research Foundation, grants 2015/26802-1 and 2016/10148-3.

✉ Carlos Elmadjian
elmad@ime.usp.br

Candy Gonzales
candytg@ime.usp.br

Rodrigo Lima da Costa
rodrigo.lima@ime.usp.br

Carlos H. Morimoto
hitoshi@ime.usp.br

[1] University of São Paulo, R. do Matão, 1010, 256-A, São Paulo, Brazil

[2] University of São Paulo, R. do Matão, 1010, 209-C, São Paulo, Brazil

Published online: 11 October 2022

novel applications, such as in the AR/VR domain (Pfeiffer, 2008). The most commonly used eye-movement patterns for applications are fixations, saccades, and smooth pursuits. An eye fixation is typically elicited when the user looks at a single scene point, creating a temporary cluster of gaze coordinates within a limited area. Saccades are said to be rapid ballistic eye movements in a single direction, required to change fixation points, whereas smooth pursuits are performed when following a moving target with the head relatively still (Leigh & Zee, 2015; Hessels et al., 2018).

Though pointing in a computer interface can be made directly with our eyes, target selection using gaze only is still a challenge. Besides blinks and winks, other interaction paradigms have been suggested based on these three basic eye-movement patterns. The most common paradigm uses long target fixations where the user must stare at the desired target for a predetermined dwell-time, until it gets selected (Jacob, 1990). Target selection using saccades is possible using eye transition between gazed regions (Huckauf & Urbina, 2008; Diaz et al., 2013; Kurauchi et al., 2020), and smooth pursuits have been suggested as a calibration-free interaction technique (Vidal et al., 2012; Velloso et al., 2018).

The simultaneous classification of these three eye-movement patterns is particularly challenging due to the noise and low precision of the eye tracker, and also because it is not always easy to isolate one eye behavior from the other, such as fixations from smooth pursuits or vestibulo-ocular reflex (Komogortsev & Karpov, 2013). This simultaneous classification is often referred to as the "ternary eye-movement classification problem" (3EMCP) (Berndt et al., 2019). Early solutions to the 3EMCP resorted to either threshold-based algorithms (Nyström & Holmqvist, 2010; Koh et al., 2010; Komogortsev & Karpov, 2013; Diaz-Tula & Morimoto, 2017) or statistical models (Komogortsev & Khan, 2007, 2009). They are known to be lightweight and able to make predictions at high frame rates, but often at the cost of a subpar accuracy and lack of generalization compared to more modern data-driven methods.

With the popularization of deep learning techniques, the 3EMCP has been successfully modeled as a sequence-to-sequence (seq2seq) problem, in which recurrent networks are trained with temporal eye data windows of a fixed size. This approach has led to the state-of-the-art offline eye-movement event recognition algorithms (Startsev et al., 2019a, c; Elmadjian et al., 2020), but these solutions have some issues when used in online real-time applications.

In the case of real-time classification, it is necessary to find a reasonable trade-off between model accuracy and latency to make a prediction, since complex models might be unable to process high sampling rates. This trade-off is particularly relevant for the online 3EMCP classifier, as it must constantly predict transition patterns without being able to take future information into account, as in the case of its offline counterpart. Figure 1 illustrates this process.

In a previous work (Elmadjian et al., 2020), we have shown that temporal convolutional networks (TCNs) can achieve higher F1 scores than bi-directional recurrent architectures in the offline 3EMCP. We performed a series of experiments comparing CNN-BiLSTM models (Startsev et al., 2019a, c) against a deep TCN model with a non-causal structure, which resulted in a gain of approximately 3% on smooth pursuit classification in favor of the TCN. However, the high complexity of this previous model makes it inappropriate for real-time applications, due to latency and the non-causality configuration. As leakage of future information should be avoided in such a setting (i.e., using features from a timestamp t_{i+n} to make a prediction at t_i), in this paper we argue that the original causal TCN (Bai et al., 2018) classifier is more appropriate for online applications.

To extend our previous study, we now evaluate a causal sequential-to-one (seq2one) TCN model against other adapted state-of-the-art deep neural architectures using an additional novel dataset with non-calibrated eye data. In order to do that, we propose a novel preprocessing technique for feature extraction that allows these models to be more lightweight and compatible with the online 3EMCP. As previous deep neural models have been evaluated only in offline settings, we have selected the I-BDT classifier (Santini et al., 2016) as our baseline to evaluate the online performance of all models.

The main contributions of this work can be summarized as follows:

- The assessment of end-to-end architectures for real-time 3EMCP applications. We conduct a series of experiments and simulations with a continuous stream of data from two different large datasets, analyzing the accuracy and prediction latency from each model.
- A novel data extraction technique tailored for online eye-movement pattern recognition that, in contrast with the one proposed in Startsev et al. (2019a), does not make use of future information and is less memory intensive, without sacrificing accuracy.
- A novel dataset with almost 1.5 million samples of eye-tracking data (approximately 2 h of recordings) labeled with stimuli-induced events and collected using a head-mounted eye tracker.

The next section presents works related to eye-movement classification, with a focus on recent data-driven models. Our proposed model and methodology are described in Section "Proposed and evaluated models".

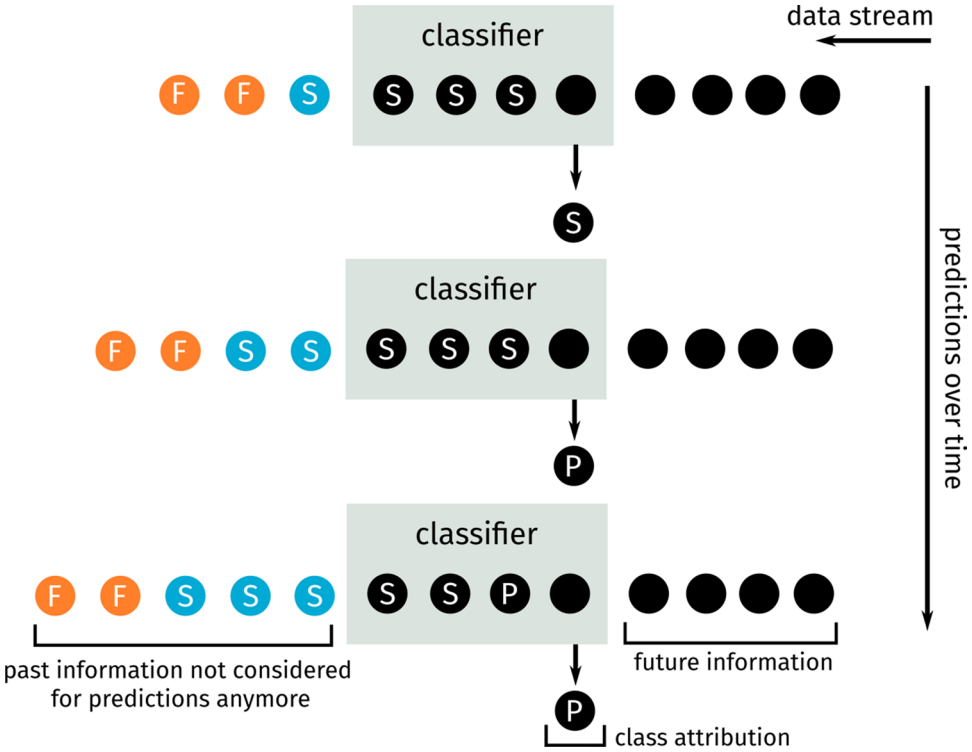

Fig. 1 In the online problem, a classifier does not have access beforehand to future samples and cannot take too long to make a prediction. Therefore, transitioning patterns and high-frequency data will usually be more challenging to handle. Here, we see an example of an online classifier that predicts one sample at a time from sequences of a fixed size. The symbols *F*, *S*, and *P* stand for fixations, saccades, and smooth pursuits, respectively. *Colored samples* indicate already classified samples that are not considered for predictions anymore

Related work

Early solutions to eye-movement classification mostly relied on threshold-based algorithms. Due to its simplicity and low computational requirements, these techniques are still applied in modern real-time applications.

One such example is the *velocity threshold* criterion (Salvucci and Goldberg, 2000) that separates saccades from other eye movements based on their characteristic high velocity that can be computed from the eye data stream. However, static velocity thresholds are not robust to noisy gaze data, which instigated the development of adaptive velocity thresholds (Nyström & Holmqvist, 2010).

Dispersion threshold models fixations as consecutive samples within maximum spatial separation under a fixed period (Berg et al., 2009). Berg et al. (2009) demonstrated that this technique tends to produce more reliable results than velocity-based algorithms, and it is also possible to use principal component analysis (PCA) to establish more generic dispersion thresholds for the discrimination of fixations, saccades, and smooth pursuits.

Although fast and simple to implement, threshold-based methods present low accuracy and require constant human intervention for parameter tuning. To overcome this, researchers started to devise more elaborated techniques that could also be fit for online classification. Komogortsev and Khan (2007) created, for example, the attention focus Kalman filter (AFKF), a framework that aimed to solve the 3EMCP in real-time. The AFKF framework was in fact an extension of a previous method (Sauter et al., 1991), in which a Kalman filter with a χ^2 test is used to detect saccades. In this updated version, Komogortsev and Khan proposed that smooth pursuits could be treated as a negative event (i.e., neither saccades or fixations), provided they did not surpass the velocity of 140 deg/s. A user study (Koh et al., 2009) confirmed afterwards that an AFKF-based technique (Komogortsev & Khan, 2007) resulted in superior accuracy than a velocity-based approach (Salvucci & Goldberg, 2000).

Santini et al. (2016) presented a more reliable 3EMCP solution called the Bayesian Decision Theory Identification (I-BDT) algorithm. This method consisted of defining priors and likelihoods for all events, and then calculating the posterior for each event, given eye velocity and movement ratio over windows according to the Bayes' rule.

Although more sophisticated models continued to be proposed in the literature, many of these works focused primarily on accuracy, at the expense of being able to run

in an online fashion (Larsson et al., 2015; Hoppe and Bulling, 2016; Fu et al., 2018; Berndt et al., 2019). Thus, in this work, we will consider the I-BDT (Santini et al., 2016) as a baseline for 3EMCP real-time applications.

Data-driven models

With the growth and popularity of novel machine learning techniques, data-driven methods also started to be exploited in the domain of eye-movement classification. These techniques are typically fully automated, leveraging approaches that can learn, from a significant amount of data, the parameters for a classifier.

Classical machine learning techniques, such as the one proposed by Vidal et al. (2012), considered a set of shape features that characterize smooth pursuits as a statistical phenomenon. Another example is the work of Zemblys et al. (2018), in which they introduced a random forests-based model to classify fixations, saccades, and post-saccadic oscillations.

With the widespread use of artificial neural models, Hoppe and Bulling (2016) proposed a CNN-based end-to-end architecture to classify fixations, saccades, and smooth pursuits in a continuous gaze stream, from which frequency features are extracted to train their model. Later, Startsev et al. (2019a) introduced a 1D-CNN-BiLSTM *seq2seq* network that also addresses the 3EMCP. Besides the three eye-movement classes, they also added a 'noise' class, so that the model was not forced to choose between one of the three. They showed that their 1D-CNN-BiLSTM classifier outperforms 12 previous baseline models in the GazeCom dataset (Dorr et al., 2010; Agtzidis et al., 2016; Larsson et al., 2015; Berg et al., 2009; Komogortsev and Khan, 2009; Komogortsev & Karpov, 2013) in offline classification.

Note that as the performance of neural networks is highly impacted by the amount of data available and its variability on training (Goodfellow et al., 2016), some authors suggested gaze data augmentation techniques for this problem. Zemblys et al. (2019) proposed gazeNet, a generative network that can create different kinds of eye-movement events. More recently, Fuhl (2020) introduced CNNs for raw eye-tracking data segmentation, generation, and reconstruction.

Bai et al. suggested that TCNs present desirable properties for 1D sequential patterns of arbitrary length, such as a highly parallel structure and a large reception field (Bai et al., 2018). In our previous work on the 3EMCP (Elmadjian et al., 2020), we have shown that TCNs were able to surpass other state-of-the-art models on F1 score, particularly with respect to smooth pursuits, achieving an improvement of 3–6% on sample level, depending on the context window size.

Data-driven online models

To obtain high-accuracy results in offline applications, authors frequently neglected the model suitability for real-time scenarios. A downside of most of these data-driven techniques is that they often require preprocessing steps that cannot be reproduced on online data streams.

On the other hand, classifiers such as I-BDT that were designed for online applications present lower accuracy. Furthermore, because their model is user-dependent, they required frequent parameter adjustments, whereas deep learning models can be trained with environmental variability and data from multiple users.

In this work, we aim to achieve the best of both worlds: the convenience and lightness of online models with the accuracy and generalization of offline data-driven methods. Next, we present a few adapted deep neural models that can be applied to the online 3EMCP, that are evaluated considering the trade-off between classification accuracy and prediction latency.

Proposed and evaluated models

In the domain of sequential problems, recurrent networks have been extensively and successfully applied to natural language processing (Fu et al., 2018; Ma and Hovy, 2016; Li et al., 2016) and speech recognition (Zhang et al., 2018; Wang et al., 2019) problems. More recently, alternative feed-forward architectures, such as attention-based (Vaswani et al., 2017) and temporal convolutional models (Bai et al., 2018), have been proposed due to their parallel structure and the ability to extract meaningful information from arbitrarily large contexts.

In the scope of online eye-movement classification, however, little is known about the feasibility and performance of deep neural architectures. Based on previous related work, we have selected for assessment three base architectures for this problem, altering them to work as *seq2one* models. We have also chosen the I-BDT algorithm (Santini et al., 2016) as our baseline.

We start with the **1D-CNN-BiLSTM** model based on Startsev et al. (2019c). This is a hybrid model that presents initially three 1D convolutional layers, with filters of size 32, 16, and 8, using the *ReLU* activation function. The output of the convolutional sequence is forwarded to a stacked BiLSTM, with 16 units at each layer and hyperbolic tangent activation function. The network output presents a fully connected layer with a *softmax* activation function for the multiclass problem (see Fig. 2a).

The second method to be investigated is the **1D-CNN-LSTM**. This architecture is almost identical to the 1D-CNN-BiLSTM model, except for its recurrent structure, that is

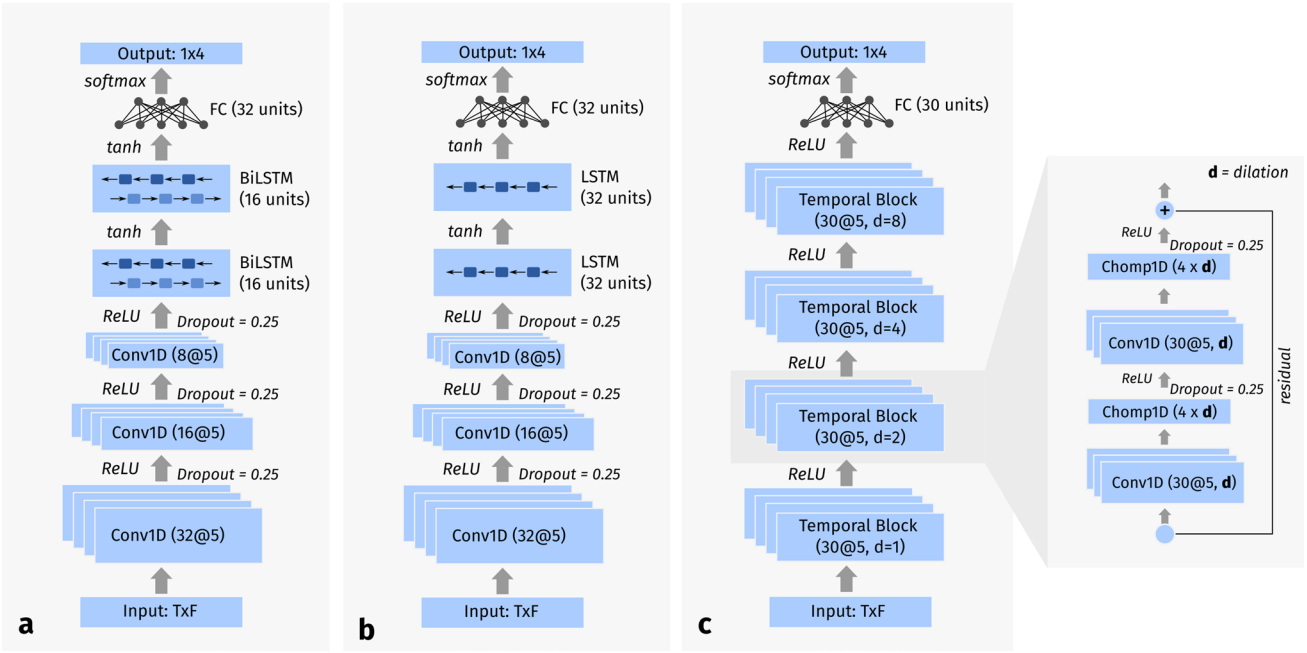

Fig. 2 Overall schematics of the evaluated deep neural architectures: the 1D-CNN-BiLSTM (**a**), the 1D-CNN-LSTM (**b**), and the causal TCN, comprised of temporal blocks (**c**). All architectures are of the form *seq2one*. The letters T and F refer to time step and feature dimensions, respectively. The symbol @ is used to indicate the kernel size applied to convolutions

strictly unidirectional, although, as a compensation for its single direction, we have doubled the number of units of the CNN-BiLSTM. We chose to build and evaluate this model because it can be regarded as Startsev et al.'s modified version for the online problem (see Fig. 2b).

The third deep neural method to be investigated is the **TCN** (Bai et al., 2018), based on a simplified version of our previous work (Elmadjian et al., 2020) that should perform better for online classification.

The architecture is composed of four hidden *temporal blocks*, with 30 units each. Each block comprises two one-dimensional convolutional layers and a residual connection. Activation between layers is done via *ReLU* function, and four different dilation sizes (1,2,4,8) are used. Unlike our previous model, all convolutions are causal, i.e., any output at time t is only convolved with elements up to time t, which ensures no leakage of future information. Zero-padding is employed to make the outputs of the temporal blocks have the same length of the inputs (Fig. 2c).

Our final investigated model is the **I-BDT** introduced by Santini et al. (2016). It is a Bayesian algorithm that makes prediction in real time for the 3EMCP. Though it is not a neural model, the I-BDT can also be regarded as data-driven approach, as its parameters can only be determined by training with a dataset. More specifically, it needs to learn the prior probabilities for each individual class, and then calculate the posterior probabilities based on these priors using the Bayes theorem (Santini et al., 2016). Some necessary modifications were made in the algorithm and are fully described in Section "Evaluating the I-BDT".

Applicability and scope of the models

In terms of applicability, deep neural models aim to answer a challenge that is beyond the scope of the I-BDT algorithm, which is generalization. The idea is that these architectures can learn from the examples and the combined variance from users observed during training to make predictions about completely unknown users, though at the cost of a hefty training time and large datasets. The I-BDT was designed to learn the parameters only for a single user, at the benefit of a short training time.

Furthermore, despite being data-driven, the I-BDT model is based on explicit heuristics, and it is easily explainable. For example, saccades and fixations are determined based on velocity likelihoods that are learned from calculated distributions of each pattern, while smooth pursuit likelihoods are determined by the average number of times that there was a movement between one sample and another (i.e., the "movement ratio", according to the authors). That is entirely different for most deep neural models: there are thousands (sometimes millions) of parameters to be learned, and the way each class is represented internally is not intuitively obvious or explainable. We say that these are completely implicit models, meaning that inference about model behavior in this case is much harder to achieve.

Online eye-movement classification presents two main issues over offline classification. The first issue is the intrinsic prediction latency from the sensor's and the models' response time, and the second issue is the inability to employ future information to disambiguate the classification without increasing response time. The lack of look-ahead information might lead to a relatively poor performance of online classifiers, particularly during state transitions. To assess the performance of the online models we will consider two public datasets, that are described in the following section. The experiment protocol is described in Section "Accuracy and latency assessment of online classification models".

Datasets

We considered two different datasets in this study: the GazeCom dataset (Startsev et al., 2019b) and another one we created specifically for this problem, which we call the Head-Mounted Raw eye-movement dataset, or simply HMR.

The GazeCom contains 18 clips from approximately 47 viewers (the number slightly varies according to the video), with labels for fixations, saccades, smooth pursuits, and noise. The labels are given based on the agreement and judgment of two experts, after observing the patterns that each user performed when watching short videos. Each clip is limited to 21 s for training.

The GazeCom data were collected using a 250-Hz remote+ eye tracker. It is constituted by 4.3 million samples, being 72.5% fixations, 10.5% saccades, 11% smooth pursuits, and 5.9% noise or blinks. The average confidence level is 99.5%. The average duration of each pattern is shown in Fig. 3. For more information about the GazeCom collection procedure, please refer to Dorr et al. (2010).

HMR dataset

The HMR dataset is composed of data from 13 participants (six female) using a head-mounted 200-Hz Pupil Core eye tracker (Kassner et al., 2014), with a reported accuracy of 0.6°, and resolution of 192 × 192 pixels. Their ages varied from 21 to 46 years old, and all of them had normal or corrected-to-normal vision. Two had a previous experience with eye trackers. For each participant, we collected the normalized pupil center data independently from left and right eyes in the eye camera space, totaling 26 video files. Each file has roughly 5 min, giving us almost 2 h of recording.

The reason for creating the HMR dataset was threefold: first, there is a lack of datasets with raw (i.e., non-calibrated) labeled eye-movement patterns captured from a head-mounted eye tracker. Second, the GazeCom dataset contains few smooth pursuit events (5%) and samples (11%). Third, we conjecture that a dataset could have more reliable labels through a stimuli-driven data collection process similar to Santini et al. (2016) (see Section "HMR data collection" for the gaze collection procedure).

In terms of statistics, the HMR dataset is constituted by 1.49 million samples, of which 56.3% are fixations, 6.4% saccades, 25.4% smooth pursuits, and 11.7% blinks. The average confidence level is 88.9%, automatically provided by the Pupil Capture software, and almost all confidence level drops are correlated to blink events. The average duration

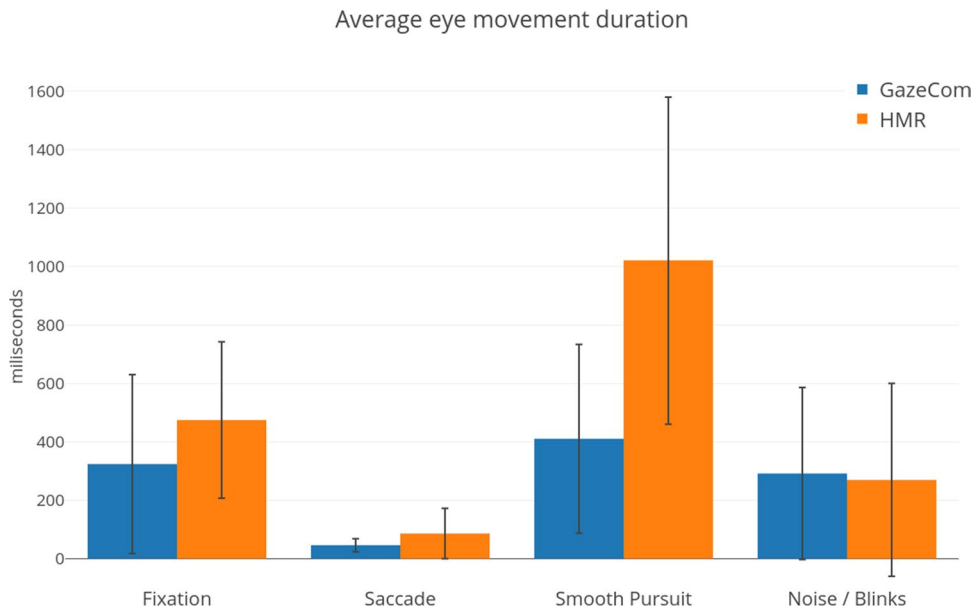

Fig. 3 Average eye-movement pattern durations from GazeCom and HMR datasets with error bars displaying 1 s.d.

of each pattern is shown in Fig. 3, along with the GazeCom data. Table 1 shows the proportion of events within and between the two datasets.

HMR data collection

To collect the data, participants sat at 50 cm from a 144-Hz 24" monitor and were asked to visually follow a target on screen. No calibration and no chin rest were employed in the process, but volunteers were advised to stay still if possible. A custom Python program was written to control the behavior of the visual targets.

A target could be placed at one of nine positions uniformly distributed on a 3 × 3 grid. The target could either stay static (in order to drive a fixation response), jump to another grid point (for a saccade response), or translate gradually to another grid point (for a pursuit response). The target could also change its color to drive a blink response. Although the color stimulus does not induce a natural blink response, we instructed participants to avoid blinking unless there was a temporary color change in the target as shown in Fig. 4d.

Target positions were randomly assigned, but followed a weighted uniform distribution, so that a less used position became more likely to be drawn at each transition. The smooth pursuit trajectories were randomly created using three distinct points of the display grid and interpolated according to a cubic polynomial regression.

The duration of each stimulus was drawn from a range of predefined values, except for blinks, which were set at 2000 ms. Fixation stimuli ranged between 400 and 700 ms, pursuits from 1500 to 2000 ms, and saccades were induced by fading out a target at one point and showing it at a different location within 270–300 ms.

The target drawing function was interpolated to match the Pupil eye-tracker sampling rate (despite the lower 144-Hz refresh rate from the monitor vs. the 200-Hz eye data rate), to obtain a one-to-one correspondence between stimuli and samples. Therefore, the eye data were automatically annotated according to the expected eye-movement response. However, because there is a natural human response delay, we had to manually shift the automatic labels by a particular constant for each individual. The whole dataset was later reviewed twice by two experts, who made fine adjustments

Table 1 Sample and event proportions found in each dataset

Eye Movement	GazeCom		HMR	
	Samples	Events	Samples	Events
Fixations	72.55%	44.93%	56.30%	45.25%
Saccades	10.53%	45.61%	6.46%	28.57%
Smooth Pursuits	11.02%	5.39%	25.47%	9.53%
Noise / Blinks	5.90%	4.07%	11.77%	16.65%

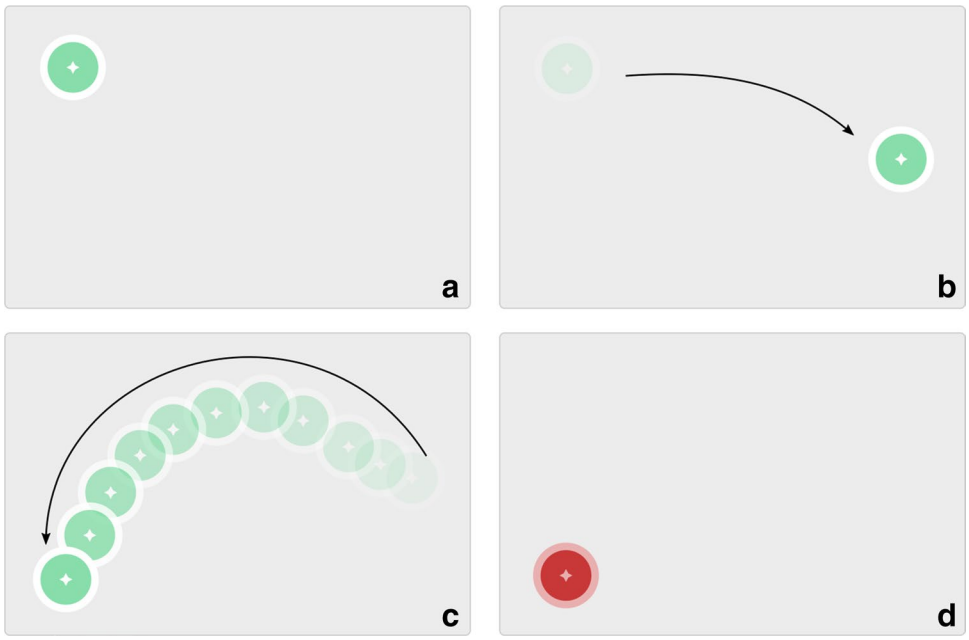

Fig. 4 Stimuli-induced experiment for the HMR data collection. Fixations were induced by stationary targets on screen (**a**); saccades, by making a previous fixation point disappearing and appearing elsewhere (**b**); smooth pursuits, by moving the target at a constant speed to another position (**c**); blinks were encouraged when the target changed its color to red (**d**)

to the length of each pattern, using a graphical tool developed by our research group.

The HMR dataset is available at our public repository, along with the visualization and editing tools used in the process. Though this is a potentially useful dataset for the eye-tracking community, it does present a more limited diversity and realism of expected eye patterns in contrast with the GazeCom dataset.

Accuracy and latency assessment of online classification models

In this section, we present a series of experiments and procedures to evaluate the trade-off between model complexity and latency of deep architectures using commodity hardware, as well as the impact of delaying prediction using short look-ahead buffers to improve the classification accuracy during transitions.

Training of the models

We have opted for a fivefold cross-validation training regimen for the architectures. Each one of them was trained using 70% of the data. We have reserved 10% of the data for validation and 20% for testing. No randomization or stratification was used for cross-validation, i.e., the models were trained with contiguous temporal chunks of data extracted from the $k-1$ folds of each dataset in lexicographic order, and then evaluated on the remaining fold not seen during training, moving the context window one sample at a time (see Fig. 5 for further details).

All deep neural architectures used a dropout rate of 25% and a kernel size of 5 for the convolutional layers. The

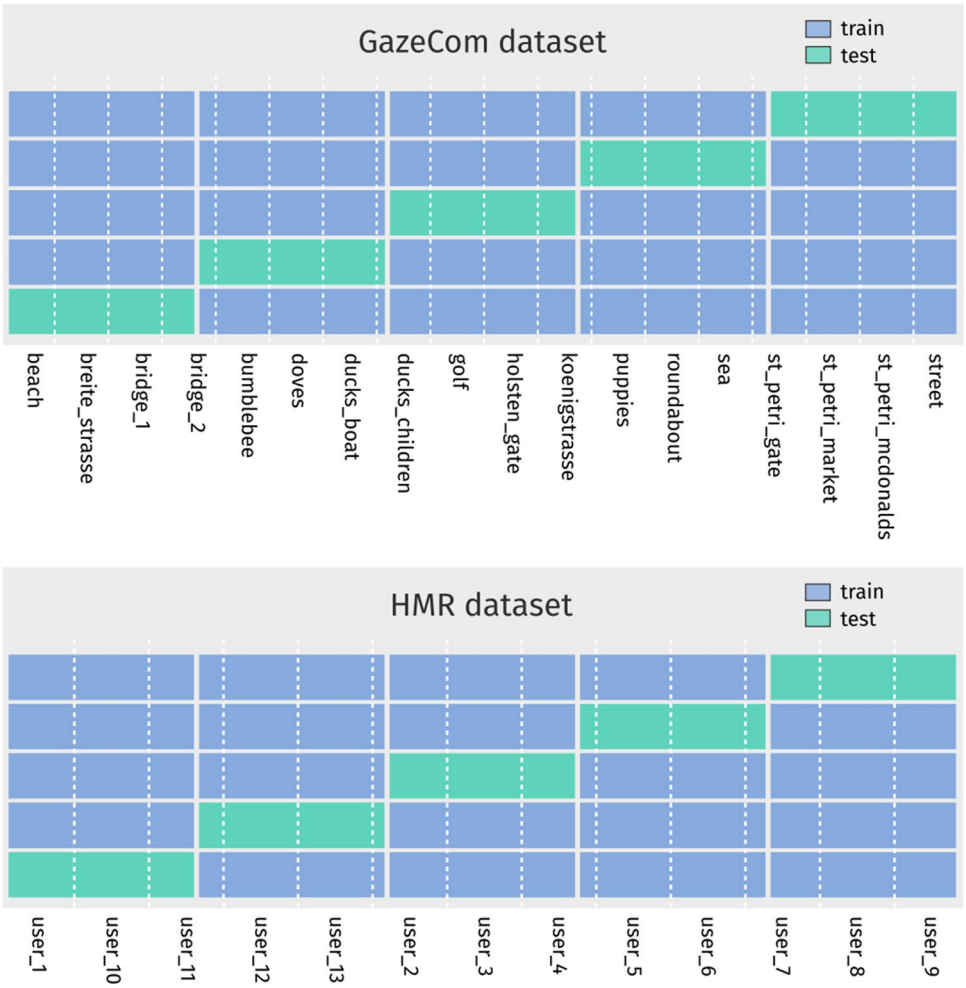

Fig. 5 Fivefold cross-validation procedure used for GazeCom and HMR datasets when training the deep neural models. The folds were split based on the lexicographic order of the recordings, so that at least two recordings from GazeCom were never seen during training, while with HMR at least one user was never seen

source code of all models is available at the following public repository: https://github.com/elmadjian/OEMC.

A multi-scale sliding window of about 1 s was used to feed the architectures, that corresponds to 250 samples on GazeCom and 200 samples on HMR. Overall, 2,957,080 context windows were considered on GazeCom and 1,069,027 on HMR. The direction and speed were calculated along different scales within a window. For the I-BDT, the hyperparameters were selected to maximize the performance of the algorithm in each eye-tracking recording. For the neural nets, the hyperparameters were determined empirically through a grid search with a subset of the GazeCom and HMR datasets. This search also confirmed previous findings on optimal configuration of number of units or layers for the CNN-BiLSTM (Startsev et al., 2019c) (i.e., when increasing model complexity does not result in improved performance). The number of learnable parameters for each neural network is shown in Table 2.

Therefore, though all models are evaluated by a common metric, the networks were trained with data from multiple recordings, while the I-BDT learned its parameters only from a single recording at a time, with the results being averaged at the end.

All models were trained and evaluated on a desktop computer with an Intel i7-7700 CPU with 16 GB of RAM, and with a NVidia GeForce GTX 1070 GPU (8Gb VRAM), running Ubuntu 18.04. The neural net models were implemented in Python 3, using the PyTorch library (1.8.1). As for the I-BDT, the public MATLAB implementation provided by the authors were used with the appropriate adjustments for each dataset (see Section "Evaluating the I-BDT" for further details).

Data preprocessing for online classification

It is often assumed that bidirectional networks are not fit for online classification because they need to traverse a whole context window in both directions in order to produce its output (Brueckner and Schuller, 2014; Startsev et al., 2019a). This also implies that such architectures tend to have higher accuracy in the central area of the context window rather than on the most recent (last) sample. Additionally, bidirectional and 1D convolutional nets usually present better results to predict a sample $s(i)$ (the one at time i) when future temporal features (from samples at time $j > i$) are used for the classification of $s(i)$. The use of a look-ahead buffer, containing samples from time $j > i$, can be used for online classification to improve accuracy, though it increases the response time delay as the online classifier must wait for all required j samples to classify $s(i)$.

Table 2 Number of learnable parameters for each neural model and dataset

Architecture	Parameters	
	GazeCom	HMR
TCN	36,634	35,734
CNN-LSTM	22,860	21,548
CNN-BiLSTM	18,764	17,452

To make training and evaluation suitable for online classification, we propose a multi-scale one-way feature extraction procedure that gives more importance to more recent samples. This method avoids leakage from future events to the past by calculating speed and direction in different time frames, taking the most recent sample of a context window as our anchor point and then performing a series of strides from there to extract the features. This method is depicted visually in Fig. 6.

A grid search determined the best number of strides used within a context window of 1 s for each dataset. For the HMR, we calculated eight sets of different scales for speed and direction, whereas for the GazeCom we used ten strides. The scale size (or stride) is always computed from the latest sample in the context window (i.e., our anchor point) with respect to the less recent samples, with the scale increasing exponentially according to the following equation:

$$\text{stride}(x) = \lceil 2^{x-s} \times n \rceil, \{x \in \mathbb{N} | x \leq s\} \quad (1)$$

where s is the maximum number of strides in a given window (i.e., eight on HMR and ten on GazeCom), and n is the number of individual samples contained in the same window (200 for HMR and 250 for GazeCom).

Investigation on the ideal number of time steps

One of the reasons why deep recurrent networks have been so successful in recent years is their ability to infer temporal dependency between sequential data. This is mostly credited to *memory cells* in the network topology, which store partial relevant information from a series (Hochreiter & Schmidhuber, 1997). TCNs, on the other hand, do not process a stream of data in an ordered fashion, but can achieve similar results through dilations and residual connections, which ultimately leads to a large receptive field that allows for long-term dependencies.

Acquiring "memory" in a 1-D sequence is usually done through tensors of the form $B \times T \times F$, in which we have a dimension representing B batches containing the samples of a context window, one for the T time steps we want to look into the past for each sample point, and another representing the F features extracted anchored on this point.

In theory, providing a comprehensive number of time steps to a model should increase its capacity and thus its ability to find long-term relationships. This has been

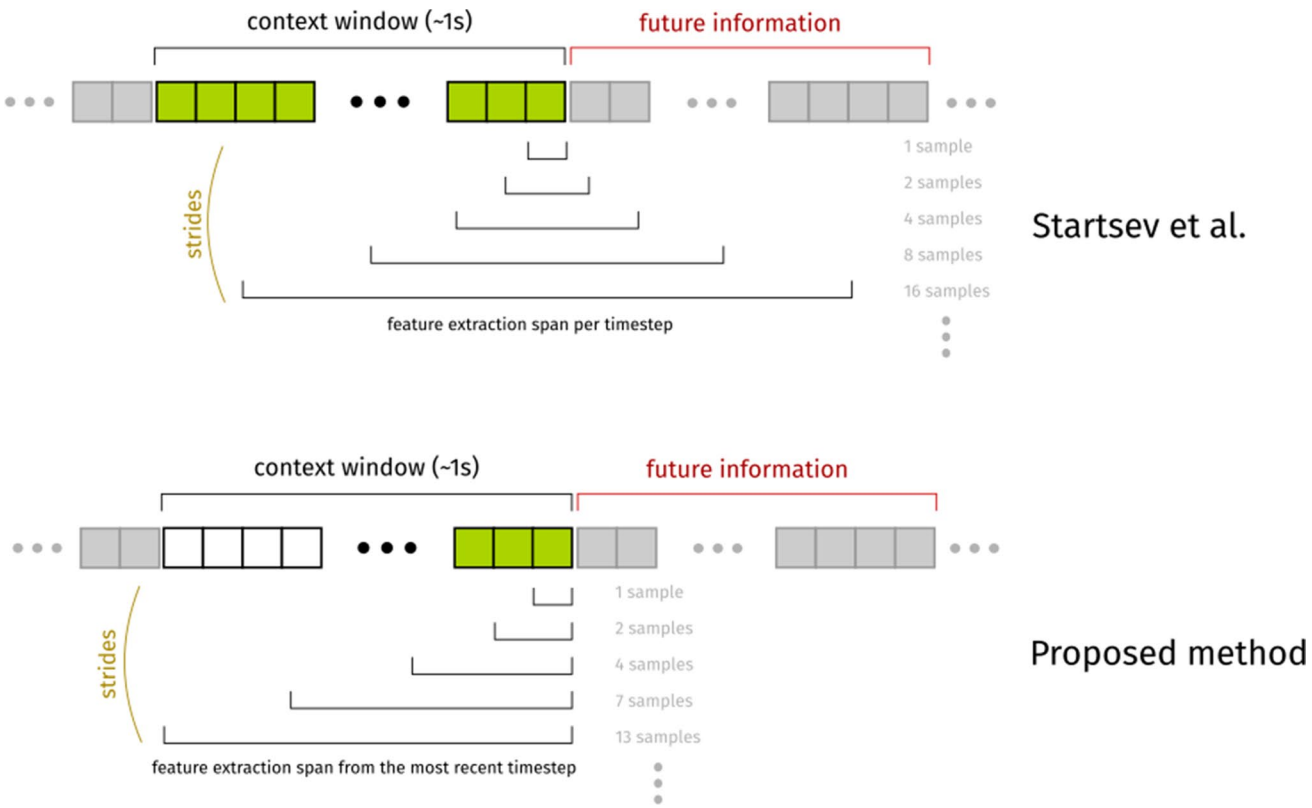

Fig. 6 With previous deep neural models (Startsev et al., 2019a), the multi-scale feature extraction step resorted to capturing information that is beyond the context window, with leakage of future information, and considering all possible time steps (*marked in green*). Our process, on the other hand, gives considerably more importance to the most recent samples, processing features only within the delimited context window and using only a fraction of time steps to minimize the response delay

empirically verified in the literature and was corroborated by our previous work, in which we expanded the number of time steps of an offline TCN to increase its accuracy (Elmadjian et al., 2020). However, an indefinite growth in time steps does not necessarily translate into improved performance, and it adds further latency and memory requirements to the model that might outweigh eventual marginal gains.

In the case of the offline 3EMCP, Startsev et al. fed their 1D-CNN-BiLSTM with mirrored time steps with the size of the whole context window (Startsev et al., 2019a) (see Fig. 6). Due to the computing cost associated with increasing time steps, and because in the online 3EMCP we are interested only in predicting the next event and not all the samples contained in the context window, we wondered whether a large number of time steps would be detrimental to the performance of the model. Therefore, we investigate the impact in model classification of different extents of time steps per context window, namely, 1%, 10%, 25%, 50%, and 100% of the context window, always considering the last sample in the window as the anchor point.

Effects of delaying model prediction

In online classification, we frequently have to deal with another major problem: pattern transition. It can be particularly challenging, even for human experts, to identify the beginning of saccades or smooth pursuits when only a few samples are available. Moreover, when we are classifying a continuous stream of data, this situation happens every time the eyes move to a different pattern.

Considering that online classifiers are most beneficial for real-time interactive applications, we investigate whether imposing a short delay on model prediction would result in a substantial accuracy improvement.

The question that arises is what would be a reasonable delay? For interactive applications, we will rely on human response time and what is perceived as instantaneous (Miller, 1968). We propose to study a set of different delays in prediction within 20, 40, 60, and 80 ms.

By delaying our prediction, in theory the classifier could have access to more samples from a given pattern to make a more qualified forecasting, which could lead to an improved general performance. This way, the model could still be perceived as instantaneous, albeit making predictions with increased latency.

Model latency on commodity hardware

Although deep learning architectures have achieved great success addressing a myriad of problems, many of them struggle in practice when running in real time due to the high computational cost. This makes an investigation on the latency of deep neural models not only legitimate, but necessary, if we are tackling an online classification problem.

Therefore, we measure the time of the three deep neural architectures using either CPU and GPU, on average, to make a prediction. This is done by aggregating all the computed prediction times in each dataset though an online simulation procedure. Our concern here is whether these models can be considered lightweight and with a fast enough throughput to be useful with off-the-shelf hardware and commercial eye trackers, which typically operate between 90 and 500 Hz.

Evaluating the I-BDT

In the dataset proposed by Santini et al., in which the I-BDT was originally evaluated (Santini et al., 2016), all recordings had a common beginning, with a set of fixations, saccades, and smooth pursuits induced by the researchers. This is not the case for the GazeCom and HMR datasets, so we could not simply select the first 15 s to train the algorithm, as it was specified in its original implementation. Therefore, for each video, we identified the 15-s interval containing the highest number of distinct classes and use it to train the I-BDT model. The remaining intervals were used for evaluation.

Additionally, we discarded all recordings in which there were no examples of either fixations, saccades, or smooth pursuits, something that was particularly recurrent in the GazeCom dataset. This gives I-BDT the opportunity to learn parameters from all patterns. A total of 601 out of 844 recordings from GazeCom were selected, whereas no recording had to be discarded from the HMR dataset.

We also applied a linear transformation to the input features. Exploratory tests showed that converting the eye-position coordinates to the same resolution of the I-BDT dataset (768 × 576 pixels) improved classification performance. Therefore, the eye features from the GazeCom and HMR datasets were re-scaled, respectively, from 1280 × 720 and 192 × 192 pixels to 768 × 576 pixels. All values were converted to integer coordinates.

In the I-BDT, all samples with no detected pupil are considered noise and thus are not fed to the classifier. Again, this information was not available in either dataset, so we marked noise and blink samples in GazeCom and HMR as no pupil detection. Furthermore, timestamps were divided by 1000 to make them compatible with the I-BDT implementation.

Evaluation metrics

We have selected the F1-score on sample and event level as our main evaluation metric. The F1-score is calculated from the precision (*Prec*) and recall (*Rec*) of the models, where the precision is the number of true-positive results divided by the number of all positive results, including those not identified correctly, and the recall is the number of true-positive results divided by the number of all samples that should have been identified as positive.

The F1-score is a measure of accuracy of the model computed from the harmonic mean of the precision and recall as

$$F1 = \frac{2 \times Prec \times Rec}{Prec + Rec} = \frac{TP}{TP + (FP + FN)/2} \tag{2}$$

where TP is the number of true positives, and FP and FN are the number of false positives and false negatives, respectively.

Precision and recall values on sample level were calculated based on the aggregated confusion matrix built after running each partial model against its corresponding test fold with contiguous eye-movement data. To compute the event scores, we considered contiguous labels in both datasets as single events, and we defined a predicted event as the highest frequency class from a model output within the ground truth span (see Fig. 7). This scoring criterion tends to be more forgiving than the intersection over union (IoU) (Hooge et al., 2018), as it does not excessively penalize failures in contiguity for predicted events, and it is similar to the criterion established by Hoppe and Bulling (2016).

An additional evaluation metric employed—in this case, only for the deep neural models—is the receiver operating characteristic (ROC) curve, as it provides a visual and more general understanding of how the classifiers are behaving at different true-positive and false-positive rates with distinct datasets. As this is a multiclass problem, we calculated the true- and false-positive rates by performing one-vs.-all scoring, and then macro-averaging the combined results for all classes. As both datasets are imbalanced, using macro averaging instead of micro averaging should be more informative, as it gives equal weights for all classes and penalizes more misclassifications of underrepresented patterns.

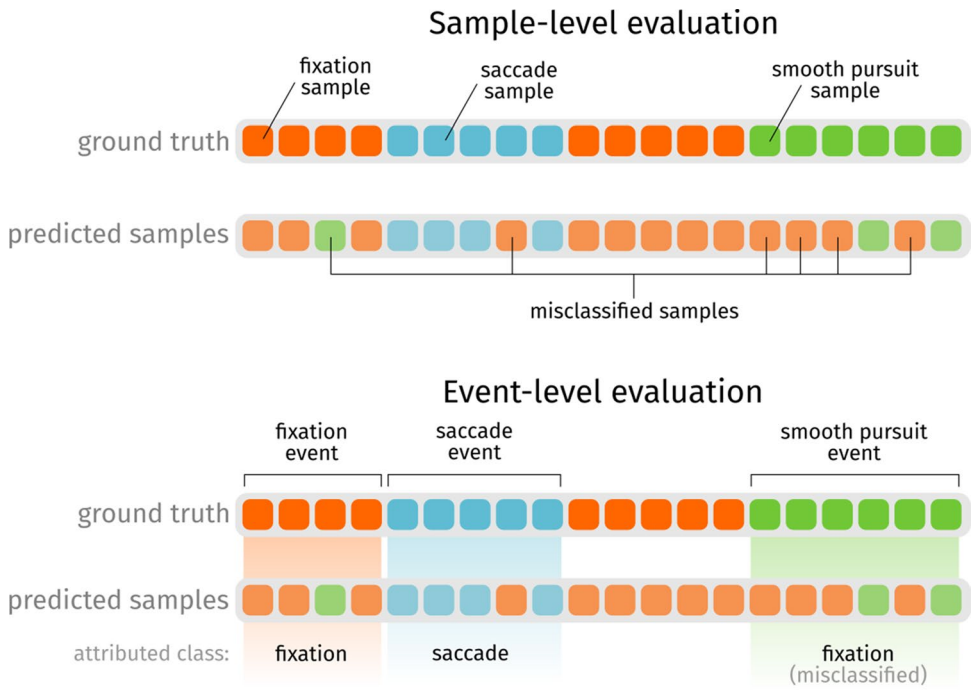

Fig. 7 Criteria for sample- and event-level evaluation. On sample level, we compute the confusion matrix by comparing each individual sample predicted by the model on a continuous data stream against the ground truth. We define an event as the set of contiguous labels on ground truth and we say that the assigned event is defined by the highest frequency class among the predicted samples within a ground truth event

Results

In this section, we present the general performance results for the methods investigated in this work, namely, the I-BDT model and the deep neural architectures TCN, CNN-LSTM, and CNN-BiLSTM. We also show some comparative and exploratory analysis of the deep neural networks, highlighting the advantages of our training regimen for this time series problem in contrast with more traditional approaches used in offline eye-movement classification.

The aggregated F1-scores for all models with respect to both datasets are shown in Table 3. The F1-scores associated to each individual pattern can be seen in Table 4. It is important to note that, although the results are displayed within the same metric, the deep neural models were trained according to a more challenging and rigorous procedure than the I-BDT algorithm. We discuss the validity and implications of this decision in the following section.

Table 3 Models tagged with * indicate results for the GazeCom dataset whereas models tagged with + are associated to the HMR dataset

Model	F1 Sample			F1 Event		
	Precision	Recall	F1	Precision	Recall	F1
I-BDT*	77.96	78.23	75.69	75.06	58.87	55.51
CNN-BiLSTM*	81.03	81.05	80.50	90.44	90.32	90.11
CNN-LSTM*	80.64	80.65	80.15	89.99	89.82	89.65
TCN*	**85.79**	**86.40**	**85.31**	**93.25**	**93.18**	**92.74**
I-BDT+	74.28	73.97	71.22	77.43	70.22	68.49
CNN-BiLSTM+	84.46	84.45	83.73	87.82	86.79	86.47
CNN-LSTM+	84.85	84.94	84.26	87.99	87.00	86.70
TCN+	**85.89**	**86.14**	**85.51**	**88.45**	**87.66**	**87.39**

Highest values for each dataset are highlighted in bold

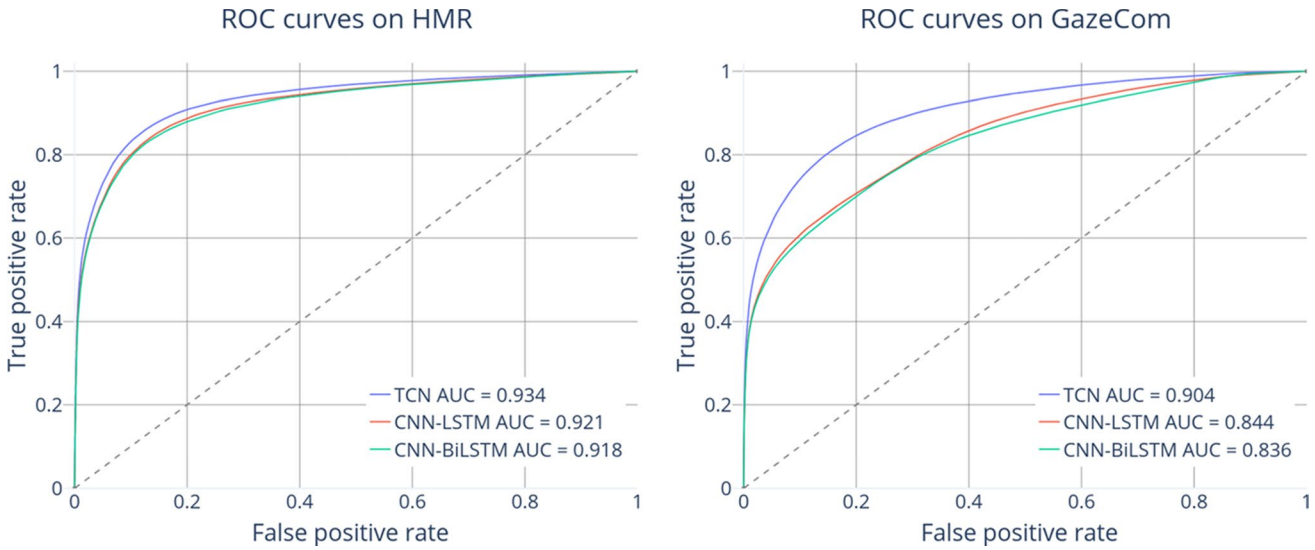

Fig. 8 ROC curves and AUC aggregated values using macro-averaging of all classes on sample level

Table 4 Models tagged * are associated to the GazeCom dataset and the ones tagged with + to the HMR dataset. The Blink columns also includes noise data

Model	F1 Sample				F1 Event			
	Fixation	Saccade	Pursuit	Blink	Fixation	Saccade	Pursuit	Blink
I-BDT*	86.80	51.02	42.16	–	68.25	47.24	37.71	–
CNN-BiLSTM*	88.38	82.09	40.65	57.17	90.82	95.77	48.48	75.98
CNN-LSTM*	88.09	82.16	38.94	57.86	90.30	95.57	45.92	64.14
TCN*	**91.97**	**83.90**	**57.30**	**59.64**	**93.60**	**97.16**	**61.15**	**77.38**
I-BDT+	82.55	56.26	49.96	–	76.51	64.25	45.70	–
CNN-BiLSTM+	88.86	69.53	85.64	62.85	89.12	91.50	81.11	74.31
CNN-LSTM+	89.19	70.41	86.49	63.48	89.36	91.28	81.85	74.96
TCN+	**90.07**	**72.07**	**88.89**	**63.73**	**90.00**	**91.87**	**84.17**	**75.02**

Highest values for each dataset are highlighted in bold

Based on the probability outputs for each class, we also built the ROC curves for all deep neural models for both datasets on sample level, with the respective area under the curve (AUC) for each model, shown in Fig. 8.

The TCN model presents the highest scores in both aggregated and individual eye-movement patterns, though the difference was more salient against other deep neural models on the Gazecom dataset, particularly when considering event-level scores. The gap between models was also more accentuated on underrepresented individual patterns in both datasets, that is, saccades on HMR and smooth pursuits on Gazecom.

To better understand the misclassification behavior of each model, we show the confusion matrix generated by the classifiers on sample level in Fig. 9. As seen in past studies using GazeCom, a major source of confusion in all models was the classification between fixations and smooth pursuits. On the HMR dataset, this kind of misclassification was less evident.

For latency measurements, we simulated trained models from all three neural architectures against a continuous stream of out-of-sample data, considering only the required time for a model to make a prediction, that is, discarding the elicited time for preprocessing the features. Figure 10 shows the average prediction latency and the standard deviation for each model on GPU and CPU.

For real-time applications with more relaxed latency constraints, we assessed whether increasing sizes of look-ahead windows could improve the classification accuracy. For this reason, we also evaluated the models considering look-ahead windows of 0 (no delay), 20, 40, 60, and 80 ms. Figure 11 shows the results for both HMR and Gazecom datasets.

Finally, we investigated how advantageous it would be to train models with few or larger number of time steps,

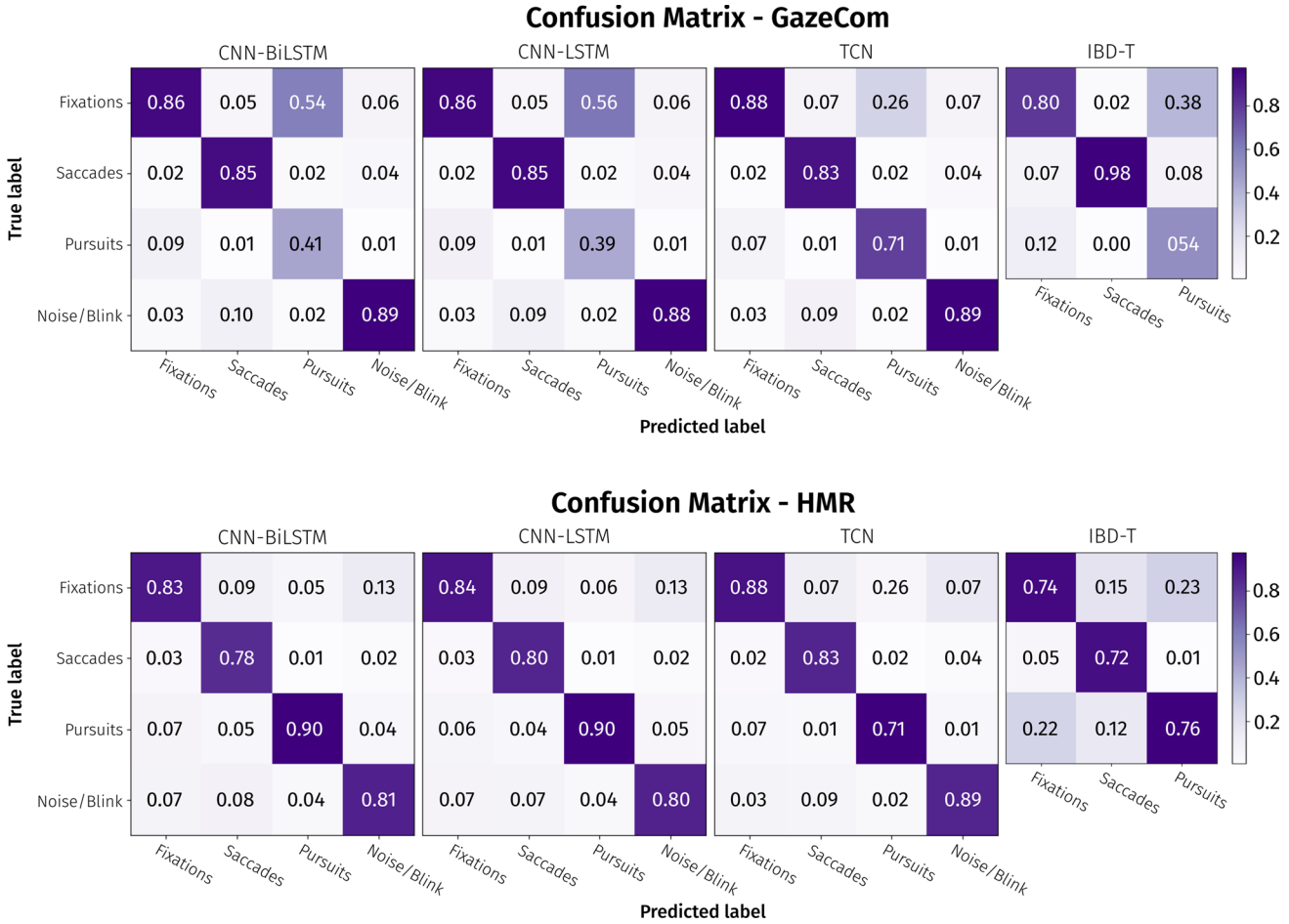

Fig. 9 Confusion matrices for all models on GazeCom and HMR datasets. The data was normalized according to the predicted value (i.e., column-wise) and measurements were made on sample level

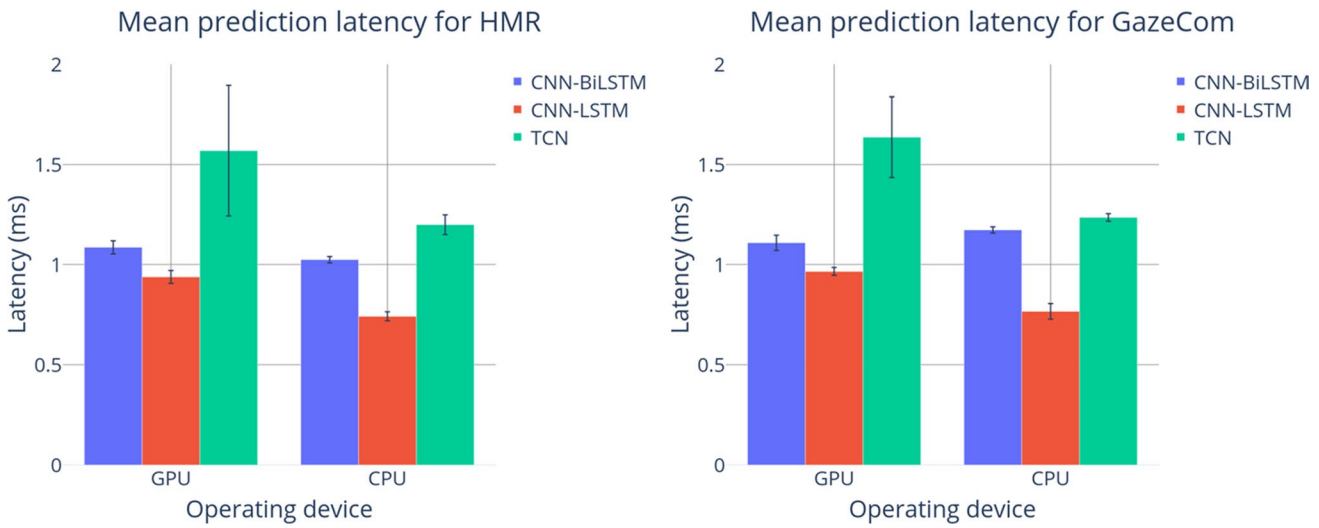

Fig. 10 Mean prediction latency using 100 ms of time steps for a single instance of the designed sliding window

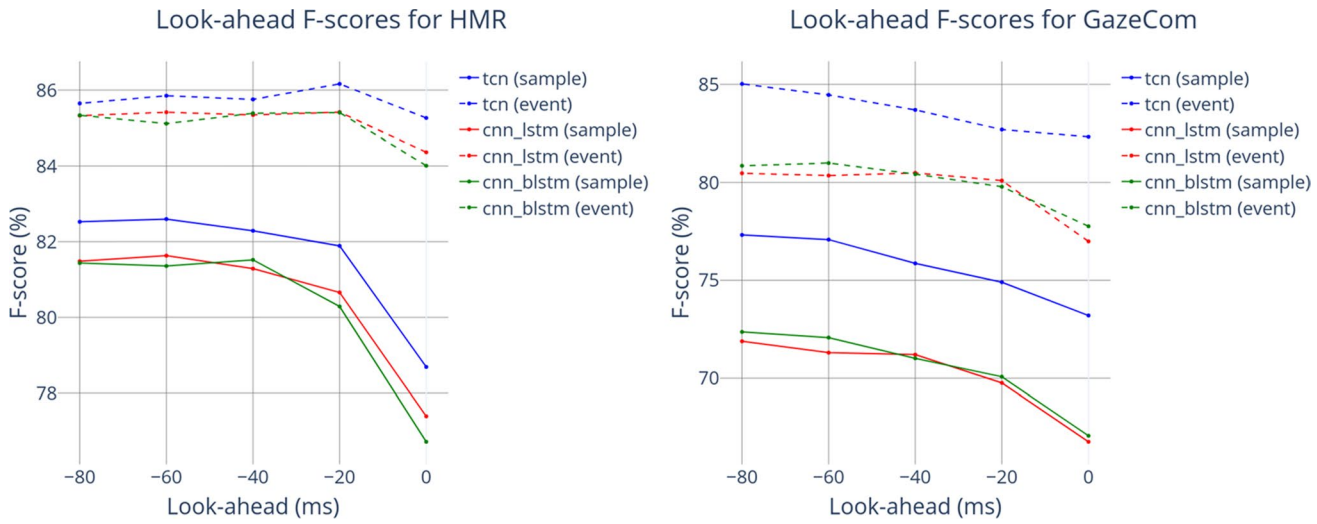

Fig. 11 Performance of deep neural models along different look-ahead steps, ranging from a lag of 0 to 80 ms

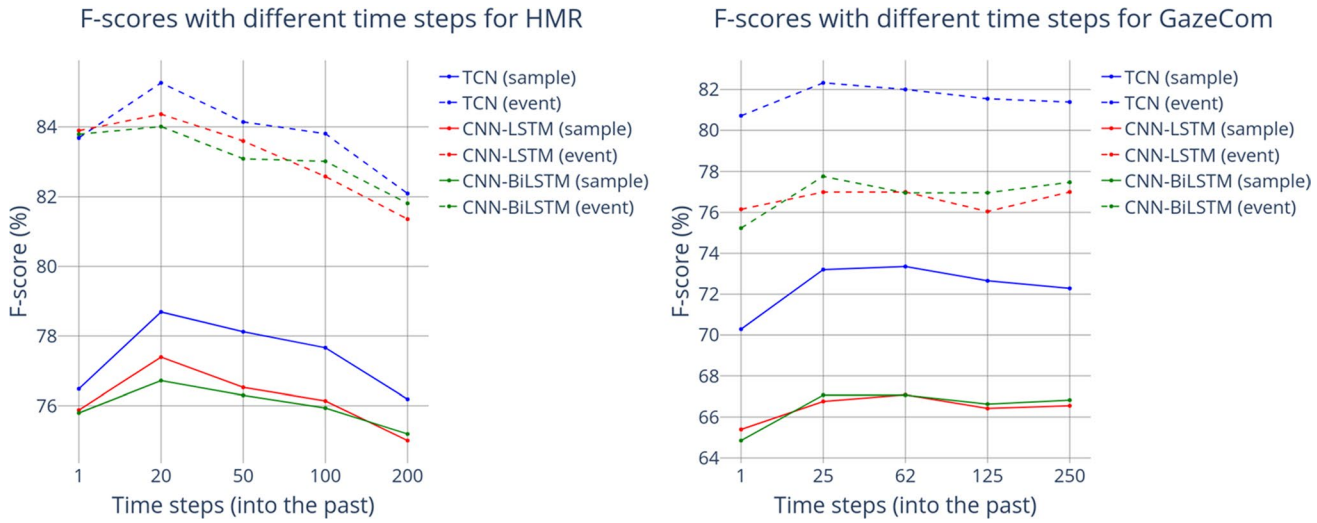

Fig. 12 Performance considering different time steps when looking into the past. Time steps are equivalent between HMR and GazeCom in terms of temporal span

as both TCNs and LSTM-based networks can learn long time-dependent relationships in the feature space. Our results indicate that all architectures peak their performance when training with approximately 100 ms of sequential past information (20 and 25 time steps, respectively, on HMR and GazeCom datasets). These results can be observed in Fig. 12.

Discussion

Our results indicate that the proposed TCN model consistently achieves the highest scores, not only in terms of general classification performance, but for individual patterns as well. This outcome is in accordance with our previous findings (Elmadjian et al., 2020), in which we showed that a non-causal TCN architecture was able to surpass other deep neural models for the offline eye-movement classification problem.

In general, all deep neural architectures scored higher than the I-BDT algorithm, our baseline model for 3EMCP online applications. This comparison with the baseline, though, has several caveats. To be fair to the limitations of the I-BDT, we trained and tested it only with individual users, and we excluded user videos where fixations, saccades, and smooth pursuits could not be found simultaneously, aside from blinks or noise, which were not fed to I-BDT.

Despite training the I-BDT using the window placement and size that maximized its performance, the deep networks were able to achieve not only a higher accuracy but also greater generalization. The cross-validated test folds were intentionally filled with data from users that were not seen during training and validation in the case of the HMR, and aside from feature preprocessing, no other treatment, such as noise reduction, data alignment, or exclusion, was employed to improve model performance.

This level of generalization and accuracy comes with the cost of a more complex training procedure in comparison to the I-BDT. This makes these models potentially unfit for scenarios where settings such as eye-tracker model or data stream frequency are constantly changing. On the other hand, if no significant changes are expected, this training happens only once, while the I-BDT always requires a per user calibration.

The results using the HMR dataset show that the TCN F1-score is about 14.3% higher on sample level and 18.9% higher on event level than the scores from I-BDT, while on GazeCom these differences were 9.6 and 37.2%, respectively, resulting on an overall improvement of ˜12% on sample level and 28% on event level. Considering the differences by class, the most noticeable one was with respect to smooth pursuits (sample 27%; event 31%).

When examining only the neural networks, the differences between the TCN the other two CNN models were more evident with the GazeCom dataset, perhaps because it has a larger data variance compared to the HMR dataset. The margins in favor of the TCN might be an indication of its larger capacity, which is corroborated by the larger gap observed between the TCN and the second best scores with respect to smooth pursuits, with a margin of 16.7% on sample level and 15.2% on event level on GazeCom. Overall, the average improvement of the TCN over the other neural models was about 3.0 and 1.7% on sample and event level, respectively.

The results are generally consistent among all deep neural nets when comparing F1-sample and F1-event scores, but the same cannot be said about the I-BDT. While there was an increase from F1-sample to F1-event scores for the deep architectures (TCN: 4.7%, CNN-LSTM: 6.0%, CNN-BiLSTM: 6.2%), the I-BDT showed a performance drop when compared to its own F1-sample scores of 2.7% on HMR and of 20.2% on GazeCom.

Figure 9 indicates that the most noticeable mistake on GazeCom occurred with the misclassification of pursuits into fixations, but not vice versa. The TCN presented the lowest error, though the remaining scores were fairly similar between the neural models. With respect to the HMR, the fixation/pursuit confusion was symmetric for the I-BDT in comparison with GazeCom, but it was not relevant for the other models. The neural models showed virtually identical results, with a small tendency to misclassify blinks as fixations.

One surprising finding was that the TCN was the slowest model in terms of prediction latency. Though all three neural architectures performed in the same order of magnitude, we expected the TCN to be more responsive due to its complete parallel structure when compared to the CNN-BiLSTM and CNN-LSTM, both of them with recurrent structures that have to be evaluated serially. There are a couple of explanations for that. First, our TCN implementation was not built completely on top of native optimized libraries from PyTorch, but the most likely reason is the fact that this TCN model is more complex, i.e., it has roughly 36,000 training parameters, while the others have about 20,000.

Overall, all deep architectures showed a prediction latency of at most 2 ms, within the expected throughput of typical commodity eye trackers, in particular to the ones used to create both datasets (200 and 250 Hz), indicating that the trained models are indeed light enough to be deployed in real-time interactive applications. Other evidence that the adapted neural models are very lightweight is that there was no reduction in latency when using GPUs instead of CPUs for prediction, suggesting that the data transfer time dominated the process in the case of GPUs.

As for considering "look-ahead" buffers, that is, delaying the model prediction to improve classification accuracy, we noticed a perceptible gain. The results indicate that all architectures behave consistently, though distinctively, within each dataset. On GazeCom, larger thresholds seem more beneficial. Nonetheless, there is a clear trade-off in which the contribution of increasing the look-ahead window starts to fade away. Based on our results, a reasonable look-ahead window, for all models, seems to be at 40 ms and at 60 ms for the HMR and GazeCom datasets, respectively.

Finally, in terms of time steps needed to build our feature tensor, our results indicate that roughly 100 ms is the ideal amount of sequential time steps from the past that need to be encoded for all models to achieve the highest scores in both HMR and GazeCom datasets. This goes contrary to the belief that more time steps leads to performance improvement, as it is the case with offline classification. One explanation could be the increasing entropy between older time steps and the most recent sample in a context window, whereas in the offline problem we can leak information from future and traverse data in both ways to increase accuracy.

Conclusions

In this work, we proposed a novel preprocessing technique and adapted state-of-the-art deep neural models for the online classification of eye-movement patterns. We showed, in particular, that the TCN architecture presents a larger

capacity, achieving higher F1-scores than the I-BDT and the 1D-CNN-LSTM networks in the online 3EMCP. Our evaluation using two different datasets shows that the TCN scores about 12% higher on sample level and 28% higher on event level over the I-BDT. Our results also show that the TCN outperforms the CNN-LSTM and CNN-BiLSTM by approximately 3% on sample and 2% on event level.

By modifying the deep neural models from *seq2seq* to *seq2one* architectures, training them with just a few time steps and increasing the importance of more recent samples, we managed to achieve a high throughput on sample prediction (approximately 500 Hz) using off-the-shelf hardware while maintaining a high accuracy.

Our investigation also shows that, though it is possible to achieve reasonable accuracy levels with zero-length look-ahead buffers, the performance of all methods improves as we increase the amount of look-ahead information, which is particularly relevant during eye-pattern transition. All methods have presented a 2–3% improvement in F1-score using a look-ahead window of 40 to 60 ms. For typical human-computer interaction applications that require response times under 100 ms, this increase in response delay might not have any impact on the task performance.

Declarations

Conflict of interest The authors declare that they have no conflicts of interest.

References

Abdrabou, Y., Shams, A., Mantawy, M. O., Ahmad Khan, A., Khamis, M., Alt, F., & Abdelrahman, Y. (2021). Gazemeter: Exploring the usage of gaze behaviour to enhance password assessments. In *ACM Symposium on eye tracking research and applications.*Association for Computing Machinery. https://doi.org/10.1145/3448017.3457384

Agtzidis, I., Startsev, M., & Dorr, M. (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)* (pp. 65–68).

Bai, S., Kolter, J.Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling.

Bayat, A., & Pomplun, M. (2017). Biometric identification through eye-movement patterns. In D.N. Cassenti (Ed.) *Advances in human factors in simulation and modeling - proceedings of the AHFE 2017 international conference on human factors in simulation and modeling, July 17-21, 2017, The Westin Bonaventure Hotel, Los Angeles, California, USA, Advances in Intelligent Systems and Computing*(Vol. 591 pp. 583–594). Springer. https://doi.org/10.1007/978-3-319-60591-3_53

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), 19–19. https://doi.org/10.1167/9.5.19

Berndt, S., Kirkpatrick, D., Taviano, T., & Komogortsev, O. (2019). Tertiary eye movement classification by a hybrid algorithm.

Brueckner, R., & Schuller, B. W. (2014). Social signal classification using deep blstm recurrent neural networks. In *IEEE International conference on acoustics, speech and signal processing, ICASSP 2014, Florence, Italy, May 4-9, 2014*(pp. 4823–4827). IEEE. https://doi.org/10.1109/ICASSP.2014.6854518

Burch, M., Kumar, A., & Timmermans, N. (2019). An interactive web-based visual analytics tool for detecting strategic eye movement patterns. In K. Krejtz, & B. Sharif (Eds.) *Proceedings of the 11th ACM symposium on eye tracking research & applications, ETRA 2019, Denver, CO, USA, June 25-28, 2019* (pp. 93:1–93:5). ACM. https://doi.org/10.1145/3317960.3321615

de Greef, T, Lafeber, H, van Oostendorp, H, & Lindenberg, J (2009). Eye movement as indicators of mental workload to trigger adaptive automation. In D Schmorrow, IV Estabrooke, & M Grootjen (Eds.) *Foundations of augmented cognition. Neuroergonomics and operational neuroscience, 5th international conference, FAC 2009 held as part of HCI international 2009 San Diego, CA, USA, July 19–24, 2009, proceedings, lecture notes in computer science* (Vol. 5638, pp. 219–228). Springer. https://doi.org/10.1007/978-3-642-02812-0_26

Diaz, G., Cooper, J., Kit, D., & Hayhoe, M. (2013). Real-time recording and classification of eye movements in an immersive virtual environment. *Journal of Vision, 13*(12), 5–5. https://doi.org/10.1167/13.12.5

Diaz-Tula, A., & Morimoto, C. H. (2017). Robust, real-time eye movement classification for gaze interaction using finite state machines. In *2017 COGAIN symposium*.

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. https://doi.org/10.1167/10.10.28

Edwards, G. W. (1998). A tool for creating eye-aware applications that adapt to changes in user behaviors. In M. Blattner, & A.I. Karshmer (Eds.) *Proceedings of the third international ACM conference on assistive technologies, ASSETS 1998, Marina del Rey, CA, USA, April 15-17, 1998*(pp. 67–74). ACM. https://doi.org/10.1145/274497.274511

Elmadjian, C.E.L., Gonzales, C., & Morimoto, C.H. (2020). Eye movement classification with temporal convolutional networks. In A.D. Bimbo, R. Cucchiara, S. Sclaroff, G.M. Farinella, T. Mei, M. Bertini, H.J. Escalante, & R. Vezzani (Eds.) *Pattern recognition. ICPR International workshops and challenges - virtual event, January 10–15, 2021, Proceedings, Part III, lecture notes in computer science* (Vol. 12663, pp. 390–404). Springer. https://doi.org/10.1007/978-3-030-68796-0_28

Feng, W., Zou, J., Kurauchi, A., Morimoto, C. H., & Betke, M. (2021). Hgaze typing: Head-gesture assisted gaze typing. In *ACM Symposium on eye tracking research and applications*. Association for Computing Machinery. https://doi.org/10.1145/3448017.3457379

Fu, L., Yin, Z., Wang, X., & Liu, Y. (2018). A hybrid algorithm for text classification based on CNN-BLSTM with attention. In M. Dong, M.A. Bijaksana, H. Sujaini, A. Romadhony, F.Z. Ruskanda, E. Nurfadhilah, & L.R. Aini (Eds.) *2018 International conference on Asian language processing, IALP 2018, Bandung, Indonesia, November 15–17, 2018*(pp. 31–34). IEEE. https://doi.org/10.1109/IALP.2018.8629219

Fuhl, W. (2020). Fully convolutional neural networks for raw eye tracking data segmentation, generation and reconstruction.

George, A., & Routray, A. (2016). A score level fusion method for eye movement biometrics. *Pattern Recognition Letters, 82*, 207–215. https://doi.org/10.1016/j.patrec.2015.11.020

Goodfellow, IJ, Bengio, Y, & Courville, AC (2016). Deep learning. Adaptive computation and machine learning. MIT Press. http://www.deeplearningbook.org/. Accessed 15 Mar 2022

Hessels, R. S., Niehorster, D. C., Nyström, M., Andersson, R., & Hooge, I. T. (2018). Is the eye-movement field confused about fixations and saccades? A survey among 124 researchers. *Royal Society Open Science, 5*(8), 180502.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hooge, I., Niehorster, D., Nyström, M., Andersson, R., & Hessels, R. (2018). Is human classification by experienced untrained observers a gold standard in fixation detection? *Behavior Research Methods, 50*(5), 1864–1881. Copyright 2018. Published by Elsevier Ltd. https://doi.org/10.3758/s13428-017-0955-x

Hoppe, S., & Bulling, A. (2016). End-to-end eye movement detection using convolutional neural networks.

Huckauf, A., & Urbina, M. H. (2008). Gazing with pEYEs: Towards a universal input for various applications. In *Proceedings of the 2008 symposium on eye tracking research & applications* (pp. 51–54).

Jacob, R. J. (1990). What you look at is what you get: Eye movement-based interaction techniques. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 11–18).

Kassner, M., Patera, W., & Bulling, A. (2014). Pupil: An open source platform for pervasive eye tracking and mobile gaze-based interaction. In A.J. Brush, A. Friday, J.A. Kientz, J. Scott, & J. Song (Eds.) *The 2014 ACM conference on ubiquitous computing, UbiComp '14 Adjunct, Seattle, WA, USA - September 13 - 17, 2014* (pp. 1151–1160). ACM. https://doi.org/10.1145/2638728.2641695

Koh, D. H., Munikrishne Gowda, S. A., & Komogortsev, O. V. (2009). Input evaluation of an eye-gaze-guided interface: Kalman filter vs. velocity threshold eye movement identification. In *Proceedings of the 1st ACM SIGCHI symposium on engineering interactive computing systems, EICS '09* (pp. 197–202). Association for Computing Machinery. https://doi.org/10.1145/1570433.1570470

Koh, D. H., Munikrishne Gowda, S., & Komogortsev, O. V. (2010). Real time eye movement identification protocol. In *CHI '10 Extended Abstracts on Human Factors in Computing Systems, CHI EA '10* (pp. 3499–3504). Association for Computing Machinery, New York, NY. https://doi.org/10.1145/1753846.1754008

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. https://doi.org/10.3758/s13428-012-0234-9

Komogortsev, O.V., & Khan, J.I. (2007). Kalman filtering in the design of eye-gaze-guided computer interfaces. In *Proceedings of the 12th international conference on human-computer interaction: Intelligent multimodal interaction environments, HCI'07* (pp. 679–689). Springer. http://dl.acm.org/citation.cfm?id=1769590.1769667

Komogortsev, O. V., & Khan, J. I. (2009). Eye movement prediction by oculomotor plant Kalman filter with brainstem control. *Journal of Control Theory and Applications, 7*(1), 14–22. https://doi.org/10.1007/s11768-009-7218-z.

Koochaki, F., & Najafizadeh, L. (2018). Predicting intention through eye gaze patterns. In *2018 IEEE biomedical circuits and systems conference, BioCAS 2018, Cleveland, OH, USA, October 17-19, 2018* (pp. 1–4). IEEE. https://doi.org/10.1109/BIOCAS.2018.8584665

Kurauchi, A., Feng, W., Joshi, A., Morimoto, C. H., & Betke, M. (2020). Swipe&switch: Text entry using gaze paths and context switching. In *Adjunct publication of the 33rd annual ACM symposium on user interface software and technology, UIST '20 adjunct* (pp. 84–86). New York: Association for Computing Machinery. https://doi.org/10.1145/3379350.3416193

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. https://doi.org/10.1016/j.bspc.2014.12.008. http://www.sciencedirect.com/science/article/pii/S1746809414002031

Leigh, R. J., & Zee, D. S. (2015). The neurology of eye movements. Contemporary Neurology.

Li, L., Wu, Z., Xu, M., Meng, H. M., & Cai, L. (2016). Combining CNN and BLSTM to extract textual and acoustic features for recognizing stances in mandarin ideological debate competition. In N. Morgan (Ed.) *Interspeech 2016, 17th annual conference of the international speech communication association, San Francisco, CA, USA, September 8-12, 2016* (pp. 1392–1396). ISCA. https://doi.org/10.21437/Interspeech.2016-324

Ma, X., & Hovy, E. H. (2016). End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th annual meeting of the association for computational linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, volume 1: long papers*: The Association for Computer Linguistics. https://doi.org/10.18653/v1/p16-1101

MacKenzie, I. S., & Zhang, X. (2008). Eye typing using word and letter prediction and a fixation algorithm. In K. Räihä, & A.T. Duchowski (Eds.) *Proceedings of the eye tracking research & application symposium, ETRA 2008, Savannah, Georgia, USA, March 26-28, 2008* (pp. 55–58). ACM. https://doi.org/10.1145/1344471.1344484

Majaranta, P., & Bulling, A. (2014). *Eye tracking and eye-based human–computer interaction* (pp. 39–65). Springer. https://doi.org/10.1007/978-1-4471-6392-3_3

Maruyama, H., Saito, Y., & Yamada, M. (2016). An analysis of changes in attention based on miniature eye movements. In *11th International conference on computer science & education, ICCSE 2016, Nagoya, Japan, August 23-25, 2016* (pp. 539–543). IEEE. https://doi.org/10.1109/ICCSE.2016.7581638

Miller, R. B. (1968). Response time in man-computer conversational transactions. In *American federation of information processing societies: proceedings of the AFIPS '68 fall joint computer conference, December 9-11, 1968, San Francisco, California, USA - Part I, AFIPS conference proceedings* (Vol. 33 pp. 267–277). AFIPS / ACM / Thomson Book Company. https://doi.org/10.1145/1476589.1476628

Morimoto, C. H., & Mimica, M. R. M. (2005). Eye gaze tracking techniques for interactive applications. *Computer Vision and Image Understanding, 98*(1), 4–24. https://doi.org/10.1016/j.cviu.2004.07.010.

Morimoto, C. H., Coutinho, F. L., & Hansen, D. W. (2020). Screen-light decomposition framework for point-of-gaze estimation using a single uncalibrated camera and multiple light sources. *Journal of Mathematical Imaging and Vision, 62*(4), 586–605. https://doi.org/10.1007/s10851-020-00947-8.

Nyström, M., & Holmqvist, K. (2010). An adaptive algorithm for fixation, saccade, and glissade detection in eyetracking data. *Behavior Research Methods, 42*(1), 188–204. https://doi.org/10.3758/BRM.42.1.188.

Pfeiffer, T. (2008). Towards gaze interaction in immersive virtual reality: Evaluation of a monocular eye tracking set-up. In *Virtuelle und Erweiterte Realität-Fünfter Workshop der GI-Fachgruppe VR/AR*.

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). ACM. https://doi.org/10.1145/355017.355028

Sanches, C. L., Augereau, O., & Kise, K. (2017). Using the eye gaze to predict document reading subjective understanding. In *1st International workshop on human-document interaction, 14th IAPR international conference on document analysis and recognition, HDI@ICDAR 2017, Kyoto, Japan, November 9-15, 2017* (pp. 28–31): IEEE. https://doi.org/10.1109/ICDAR.2017.377

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). ACM. https://doi.org/10.1145/2857491.2857512

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. https://doi.org/10.1007/BF02446297

Startsev, M., Agtzidis, I., & Dorr, M. (2019). 1d cnn with blstm for automated classification of fixations, saccades, and smooth pursuits. *Behavior Research Methods, 51*(2), 556–572. https://doi.org/10.3758/s13428-018-1144-2

Startsev, M., Agtzidis, I., & Dorr, M. (2019). Characterizing and automatically detecting smooth pursuit in a large-scale ground-truth data set of dynamic natural scenes. *Journal of Vision, 19*(14), 10–10. https://doi.org/10.1167/19.14.10.

Startsev, M., Agtzidis, I., & Dorr, M. (2019c). Sequence-to-sequence deep learning for eye movement classification. In *PERCEPTION (Vol. 48, pp. 200–200). SAGE PUBLICATIONS LTD 1 OLIVERS YARD, 55 CITY ROAD, LONDON EC1Y 1SP, ENGLAND*.

Tula, A. D., & Morimoto, C. H. (2016). Augkey: Increasing foveal throughput in eye typing with augmented keys. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, CHI '16* (pp. 3533–3544). ACM. https://doi.org/10.1145/2858036.2858517

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Polosukhin, I. (2017). Attention is all you need. In I. Guyon, U. von Luxburg, S. Bengio, H.M. Wallach, R. Fergus, S.V.N. Vishwanathan, & R. Garnett (Eds.) *Advances in neural information processing systems 30: annual conference on neural information processing systems 2017, December 4-9, 2017, Long Beach, CA, USA* (pp. 5998–6008). https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

Velloso, E., Coutinho, F. L., Kurauchi, A., & Morimoto, C. H. (2018). Circular orbits detection for gaze interaction using 2d correlation and profile matching algorithms. In *Proceedings of the 2018 ACM symposium on eye tracking research & applications* (p. 25). ACM.

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 and applications, ETRA '12* (pp. 177–180). ACM. https://doi.org/10.1145/2168556.2168586

Wang, C., & Hung, J. C. (2019). Comparative analysis of advertising attention to facebook social network: Evidence from eye-movement data. *Computers in Human Behavior, 100*, 192–208. https://doi.org/10.1016/j.chb.2018.08.007.

Wang, D., Wang, X., & Lv, S. (2019). End-to-end mandarin speech recognition combining CNN and BLSTM. *Symmetry, 11*(5), 644. https://doi.org/10.3390/sym11050644.

Zemblys, R., Niehorster, D. C., Komogortsev, O., & Holmqvist, K. (2018). Using machine learning to detect events in eye-tracking data. *Behavior Research Methods, 50*(1), 160–181.

Zemblys, R., Niehorster, D. C., & Holmqvist, K. (2019). gazenet: End-to-end eye-movement event detection with deep neural networks. *Behavior Research Methods, 51*(2), 840–864. https://doi.org/10.3758/s13428-018-1133-5.

Zhang, L., Wang, L., Dang, J., Guo, L., & Yu, Q. (2018). Gender-aware CNN-BLSTM for speech emotion recognition. In V. Kurková, Y. Manolopoulos, B. Hammer, L.S. Iliadis, & I. Maglogiannis (Eds.) *Artificial neural networks and machine learning - ICANN 2018 - 27th international conference on artificial neural networks, Rhodes, Greece, October 4–7, 2018, proceedings, Part I, lecture notes in computer science* (Vol. 11139 pp. 782–790). Springer. https://doi.org/10.1007/978-3-030-01418-6_76

Open Practices Statement The source code, pre-trained models, and datasets used in the experiments are available at https://github.com/elmadjian/OEMC. There were no preregistered user studies in this work.