A Neural Architecture for Detecting User Confusion in Eye-tracking Data

Shane D. Sims and Cristina Conati
The University of British Columbia
Vancouver, BC, Canada
{ssims, conati}@cs.ubc.ca

ABSTRACT

Encouraged by the success of deep learning in a variety of domains, we investigate the effectiveness of a novel application of such methods for detecting user confusion with eye-tracking data. We introduce an architecture that uses RNN and CNN sub-models in parallel, to take advantage of the temporal and visuospatial aspects of our data. Experiments with a dataset of user interactions with the ValueChart visualization tool show that our model outperforms an existing model based on a Random Forest classifier, resulting in a 22% improvement in combined confused & not confused class accuracies.

CCS CONCEPTS

• Human-centered Computing ~ User Models • Computing Methodologies ~ Neural Networks

KEYWORDS

Eye-tracking; user model; user affect; neural networks; classification

ACM Reference format:

Shane Sims and Cristina Conati. 2020. A Neural Architecture for Detecting User Confusion in Eye-tracking Data. In Proceedings of ACM International Conference on Multimodal Interaction (ICMI'20). ACM, Utrecht, The Netherlands, 8 pages. https://doi.org/10.1145/3382507.3418828

1 Introduction

There is increasing interest in creating AI agents that can predict their user's needs, states, and abilities, and then personalize the interaction with the user accordingly. This includes understanding and reacting to a user's affective state. One such state is confusion, which is particularly relevant to user experience while interacting with complex interfaces because when a user is confused, they can experience a decrease in satisfaction and performance (e.g., [27]). A system that can detect its user's confusion gains an awareness that can be leveraged to provide appropriate interventions to resolve such confusion. Detecting and resolving confusion is becoming especially relevant in supporting users interacting with Information Visualizations (InfoVis) because data visualizations are now widespread in our daily lives and confusion has been found to hinder their usage, especially when they increase in complexity (e.g., [25]).

Prior work [24] showed that confusion during visualization processing can be detected using a Random Forest (RF) classifier and features based on summative statistics of eye-tracking (ET) data (user gaze, pupil size, and head distance from the screen) computed as the interaction unfolds. This classifier achieved 57% and 91% accuracy in predicting confusion and lack thereof, respectively. In this paper, we investigate whether we can improve upon the results of [24] by employing a deep learning model to detect confusion from the same ET data set.

The use of deep learning is generally limited in research on modeling and adapting an interaction to user affect, partially due to the difficulty in collecting and labelling large amounts of relevant data. Corpora of data are available for sentiment analysis [39], i.e. detecting positive vs. negative affect (valence) from text, because it is relatively easy to label valence, at least as compared to generating labels for finer-grained emotional states. There has also been work in using deep learning to detect affect from acted emotions in video (e.g., [10]) where the affective labels are known a priori. By comparison, collecting datasets for specific unscripted user affective states in interactive tasks is very laborious, and thus such datasets are usually small compared to those in domains where deep learning has been most successful (e.g., [19]).

For this reason, approaches to predicting user affect mostly use classical machine learning methods similar to those used in [24]. There are two groups of notable exceptions. Works such as [4, 15, 18] seek to predict multiple emotions (including confusion) in students interacting with educational software. They leverage Recurrent Neural Networks (RNNs) to learn from sequences of student interface actions but do so with engineered features based on knowledge of what is important while interacting with each system, thus not fully leveraging the RNN's ability to learn representations from low-level data. The second exception relates to work that used deep learning on EEG signals to predict emotional valence and arousal in users watching short videos (1 minute), designed to elicit specific emotional reactions [36, 26]. Thus, such work is geared toward providing proof of concept on the suitability of deep learning to capture affective signals from

ICMI '20, October 25–29, 2020, Virtual event, The Netherlands

ACM ISBN 978-1-4503-7581-8/20/10$15.00
https://doi.org/10.1145/3382507.3418828

EEG data; it does not pertain to modeling and possibly responding to affect as users engage with an interactive system.

The scarcity of affective interaction data is exacerbated with ET data because collecting reliable data currently requires specialized equipment and collection in a lab setting. The dataset used in this paper is no exception, containing data from only 136 users. To address this issue, we propose a deep learning architecture purposefully designed to process eye-tracking data while being as lightweight as possible (section 4), which achieves a 49% improvement in detecting confusion compared to [24], with no loss in detecting an absence of confusion.

Therefore, the first contribution of this work is that, to the best of our knowledge, we are the first to show the suitability of a deep learning approach for classifying user affect from ET data. This result may have wider implications for the use of ET data in user modelling as a whole, where such data has been shown to have great potential for modelling not only affect (e.g. [3, 17, 23]) but also user cognitive abilities (e.g. [19]) and long-term traits (e.g., [32]). By demonstrating the effectiveness of using deep learning based methods with a relatively small eye-tracking dataset, we hope to provide an impetus for further research in this direction.

Our second contribution is the architecture we designed to achieve our results, which combines a Recurrent Neural Network (RNN) and a Convolutional Neural Network (CNN) to learn from sequential and visuospatial information in the ET data. Previous work that combined CNNs and RNNs dealt with temporal data (videos and EEG signals) suitable for having CNNs process input at each time step, as the RNN does [8, 31, 36, 26, 38]. This approach is not suitable for ET data (see section 2). However, ET data has the property that a temporal sequence of the data can be represented in a single frame in a meaningful way, namely with a scanpath image that records spatial information about the aggregate eye-movements in the sequence. To leverage this property of ET data, our proposed architecture uses an RNN sub-model that takes sequential raw eye-tracking samples as input while a CNN sub-model processes the corresponding scanpath image in parallel. The sub-models are jointly trained in an end-to-end fashion as one unit. A formal evaluation of the model shows that it achieves better performance than either of its components do alone and significantly improves over previous work using non-deep learning methods [24]. We see our results as promising evidence that our proposed approach is worthy of further investigation as a general architecture, as interest in detecting user states from eye-tracking data continues to increase and more datasets of this type become available.

The rest of this paper is structured as follows. Section 2 discusses related work. Section 3 presents the dataset of user confusion data leveraged in is paper. Section 4 presents the deep learning model we propose, including an RRN (section 4,1), a CNN (section 4.2), and the VTNet architecture that combines the two (section 4.3). Section 5 describes the evaluation of our proposed approach, and section 6 concludes the paper.

2 Related Work

The body of work in predicting user affect with deep learning methods is relatively small (compared to tasks like image classification) and occurs mostly in computer vision and natural language processing (NLP), where established methods can be adapted for classifying emotion from images, video, and speech (e.g., [1, 10, 9]). Exceptions pertain to classifying the emotions of students interacting with an intelligent tutoring system (ITS) [4, 15, 18], and affective states from EEG signals [36, 26]. The ITS related works use RNNs to classify emotion from sequences of high-level interaction events (e.g., viewing a video lecture or textbook material taking a quiz), which does not take full advantage of the RNN's ability to learn a representation from low-level data (e.g. mouse movements). Like our work, the works leveraging EEG data [36, 26] use raw signals but are concerned with predicting emotional valence and arousal in users viewing music videos explicitly designed to elicit specific emotional responses [22], as opposed to affective events spontaneously occurring during an interactive task.

Eye-tracking (ET) data has been shown to contain good predictors of both affective and cognitive states, such as mind-wandering [3], boredom and curiosity [17], affective valence [23] and learning [19] while interacting with educational software, user intention while playing a strategy game [16], reader difficulty with texts in foreign languages [29], schizophrenic symptom severity [33], and user confusion while interacting with a visualization-based decision support tool [24]. This latter work predicted confusion by combining features derived from eye-tracking and interaction data as input to a Random Forest (RF) classifier. The classifier learns from engineered features based on summative statistics (e.g., mean and standard deviation) of measures related to the user's gaze, pupil size, and head distance to the screen. These measures include, for instance, rate and duration of fixations (gaze maintained at a point), and length and angles of saccades (paths between fixations). We compare our deep learning based approach directly to this work.

The only work we identified that uses a deep learning approach to make predictions from ET data is one that aims to diagnose patient developmental disorders [30]. An RNN is used to learn patterns pertaining to the disorder from how patients look at a trained practitioner who is conducting a diagnostic interview. The model takes as input a temporal sequence that indicates if a patient was looking at certain regions (nose, jaw, etc.) of an interviewer's face or not, at each time step. Deep learning has also been used for gaze estimation (i.e. predicting the (x, y) coordinates of a person's gaze on a 2D plane) from images of the viewer's face (e.g., [41]). Note that gaze estimation is what eye-tracking hardware does and this is distinct from using the estimated gaze data for a predictive task, as done in this work.

There are a number of works that (like our own) combine the particular strengths of RNNs and CNNs. Most of these works (e.g., [8, 31, 37]) relate specifically to processing videos, using a model class known as Recurrent Convolutional Networks (RCNs). RCNs typically operate on an input of image sequences (i.e. the frames of a video), where at each step a CNN extracts visual features from the given frame and feeds them to the RNN, which models the temporal dynamics of the sequence. In addition, the aforementioned work [36, 26] on detecting affective valence and arousal from EEG signals use an RNN and CNN on a sequence of multichannel EEG signals, where at each time step the RNN is fed the vector of channel values, while a CNN is given a matrix

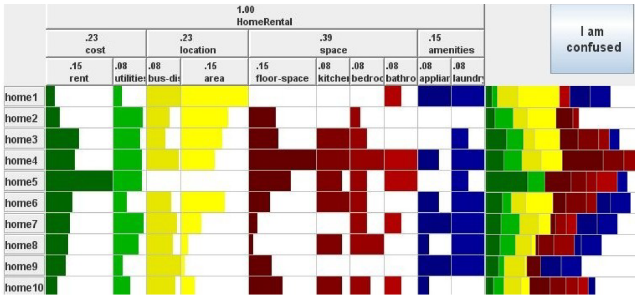


Figure 1: An example of ValueChart to choose house from available options (rows) based on attributes (columns)

representing the same values, but arranged in a way that reflects the spatial relationship among the sensors placed on the user's head. This approach leverages the strength of CNNs in detecting patterns from spatial information, but the information must be provided at each time step to reflect the changing signal values; also used by [38] to predict user intention from EEG signals.

All of these approaches combine CNN and RNN at every time step and therefore do not decouple the temporal from the spatial aspects of the data completely. Providing a temporally developed scanpath as input to a CNN at every time step (analogous to the above approach) would be less meaningful in our context because the purpose of providing the scanpath in the first place is to give a high-level picture of the user's activity over the course of an entire interaction. By providing such a single scanpath to the CNN, our approach allows for processing this high-level spatial representation of the user's overall activity prior to an episode of confusion, which complements more local temporal information about potential episodes of confusion generated by the RNN from raw sequences. Combining the CNN and RNN in this way is also beneficial computationally, as while in previous work the CNN had to operate on a datapoint at each time step, our method requires the CNN to operate only once per datapoint; an important consideration for deploying a model to a system that intends to detect and address a user's confusion in real-time.

3 Dataset

3.1 Data Collection

The dataset used in this paper is the same one used in [24], generated via a study designed to collect data for confusion episodes from users interacting with ValueChart[5], an interactive visualization-based tool for supporting decision making. Figure 1 shows an example of ValueChart configured for selecting rental properties from a set of alternatives (represented by the rows in the chart), based on a set of relevant attributes represented as columns (e.g., rent, location). The width of each column indicates the importance (weight) of the corresponding attribute. The amount of filled color in each cell specifies how the corresponding alternative fares with respect to the related attribute. The stacked bars to the right group all values for each alternative displaying its overall value (e.g., home4 in Figure 1 has the best overall value). Users can inspect the value of each attribute (e.g., the rent of home1), by left-clicking on the related alternative, they can sort the alternatives based on a specific attribute by double-clicking on its name, they can swap attribute position, and they can change an attribute's importance by resizing the width of its column. Although extensively evaluated for usability [35], the complexity of the decision tasks means that users can still experience confusion while interacting with ValueChart.

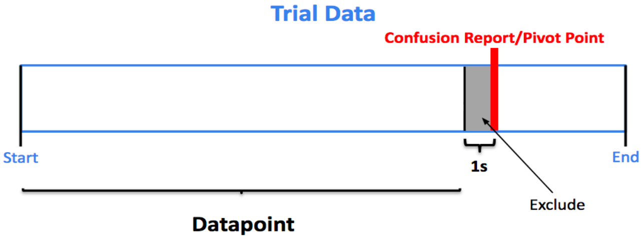


Figure 2: A datapoint of raw ET samples extracted from a study trial

In the study that generated the dataset, 136 participants performed tasks with ValueChart, relevant to exploring available options for a home rental decision problem. There were 5 task types (e.g., retrieve the cheapest home, select the best home based on size and location), each repeated 8 times, resulting in 5440 tasks (mean duration = 22.3s, st. dev. = 18.4). The user's eyes were tracked with a Tobii T120 eye-tracker embedded in the study computer's monitor. In addition to gaze, this eye-tracker also collects information on pupil size and head distance.

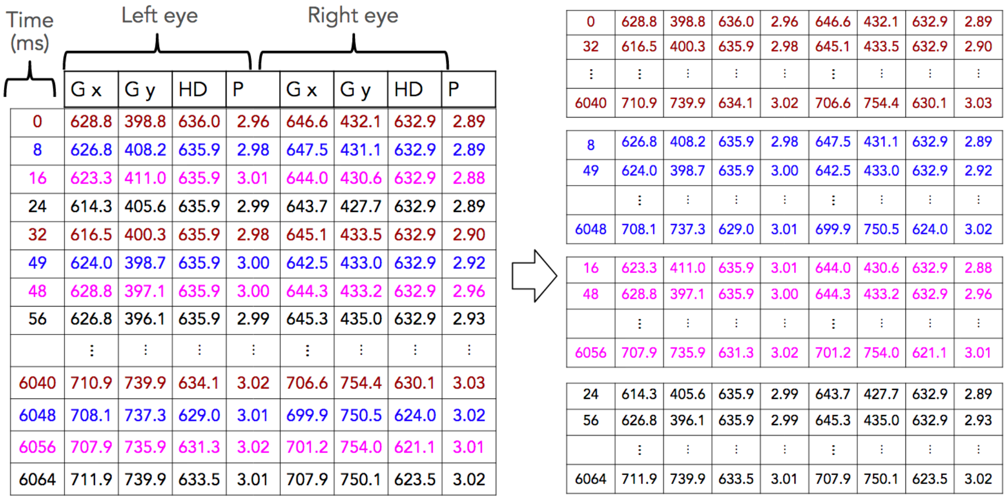
To collect ground truth labels for confusion, users self-reported their confusion during a task by clicking on a button labelled I am confused (top right in Figure 1). The confusion reports were verified at the end of the study session by asking users to confirm them after seeing replays of relevant interaction segments. This process resulted in 112 (2%) tasks with reported confusion (there was never more than one report per task) and 5328 without. This highly imbalanced dataset confirms that, overall, ValueChart has good usability but user confusion can still happen. In fact, 60% of users reported confusion at least once, indicating that it is worth capturing as a signal that indicates the user needs help.

Each datapoint in the dataset is a task segment that ends when a confusion self-report occurs or at a randomly selected pivot point for tasks where no confusion was reported (See Figure 2), with an average duration of 13.7 seconds (st. dev 11.3s). As the figure shows, the last second of data before a confusion report is removed to exclude signs of the intention to push the *I am confused* button.

3.2 Data Pre-processing

The Tobi T120 eye-tracker collects raw eye-tracking samples at a rate of 120 Hz. This raw data is usually processed with proprietary software into sequences of fixations, identified by clustering raw data to distinguish small eye-movements from real attention shifts. Leveraging fixations and saccades (the gaze paths between fixations) is the standard way to analyze ET data. In fact, the results on detecting confusion by Lallé et al. (2016) [24], which is the gold standard to which we compare our work, showed that summary statistics around fixations and saccades are strong features for classifying confusion.

In contrast, we leverage deep learning to learn from the raw ET samples, the lowest level of data available from the eye-tracker, to ascertain whether this provides any further discriminators useful for classifying confusion. Any patterns that could be lost in going



Time (ms)	Left eye				Right eye			
	G x	G y	HD	P	G x	G y	HD	P
0	628.8	398.8	636.0	2.96	646.6	432.1	632.9	2.89
8	626.8	408.2	635.9	2.98	647.5	431.1	632.9	2.89
16	623.3	411.0	635.9	3.01	644.0	430.6	632.9	2.88
24	614.3	405.6	635.9	2.99	643.7	427.7	632.9	2.89
32	616.5	400.3	635.9	2.98	645.1	433.5	632.9	2.90
49	624.0	398.7	635.9	3.00	642.5	433.0	632.9	2.92
48	628.8	397.1	635.9	3.00	644.3	433.2	632.9	2.96
56	626.8	396.1	635.9	2.99	645.3	435.0	632.9	2.93
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
6040	710.9	739.9	634.1	3.02	706.6	754.4	630.1	3.03
6048	708.1	737.3	629.0	3.01	699.9	750.5	624.0	3.02
6056	707.9	735.9	631.3	3.02	701.2	754.0	621.1	3.01
6064	711.9	739.9	633.5	3.01	707.9	750.1	623.5	3.02

0	628.8	398.8	636.0	2.96	646.6	432.1	632.9	2.89
32	616.5	400.3	635.9	2.98	645.1	433.5	632.9	2.90
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
6040	710.9	739.9	634.1	3.02	706.6	754.4	630.1	3.03

8	626.8	408.2	635.9	2.98	647.5	431.1	632.9	2.89
49	624.0	398.7	635.9	3.00	642.5	433.0	632.9	2.92
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
6048	708.1	737.3	629.0	3.01	699.9	750.5	624.0	3.02

16	623.3	411.0	635.9	3.01	644.0	430.6	632.9	2.88
48	628.8	397.1	635.9	3.00	644.3	433.2	632.9	2.96
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
6056	707.9	735.9	631.3	3.02	701.2	754.0	621.1	3.01

24	614.3	405.6	635.9	2.99	643.7	427.7	632.9	2.89
56	626.8	396.1	635.9	2.99	645.3	435.0	632.9	2.93
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
6064	711.9	739.9	633.5	3.01	707.9	750.1	623.5	3.02

Figure 3: The 2D array to the left is an example of a datapoint consisting of ET samples (rows). This datapoint is cyclically split to create four separate datapoints (right): rows that are four steps apart in the left table (coded with the same color) are assigned to the same split datapoint to the right

to a higher level of data abstraction are necessarily maintained at this level, where the model has the opportunity to discover these patterns, as well as any interactions among them [2].

Figure 3 (left) shows an example of a datapoint consisting of a sequence of raw ET samples, namely a 2D array with the number of rows corresponding to the number of samples captured in one of the confused/not confused datapoints described in section 3.1 (and shown in Figure 2). Each ET sample (a row in Figure 3, left) includes 4 measures for each eye: the x and y gaze coordinates (Gx, Gy, in Figure 3 (left)) on the study screen, the size of the pupil (P), and the distance of that eye from the screen (HD).

An advantage of learning from the raw ET samples is that they can support ad hoc data augmentation. Data augmentation is commonly used to deal with limited data availability, and in its simplest incarnation, it involves duplicating data points exactly (random over-sampling) [12]. Because of the nature of our data, we can do something better. We observe that in our datapoints (i.e., sequences of raw ET samples), values change only by a small amount from one sample to the next, because of the high sampling rate. This can be seen by looking at adjacent rows on the left of Figure 3. Given this observation, we split the sequence of ET samples in each datapoint into four separate datapoints with the same label of confusion or lack thereof. We do so by performing a cyclic split (e.g., as when dealing a deck of cards), which preserves the temporal structure of the time series data. Figure 3 demonstrates this splitting process: samples (rows) that are four steps apart in the 2D array to the left (coded with the same color in the figure) are assigned to the same split datapoint to the right. Thus, a datapoint with n samples is cyclically split into four datapoints, each containing n/4 samples.

This cyclical split provides our deep learning models with multiple opportunities to learn from the same datapoint in a more intelligent way than by simply duplicating it. The difference between resulting items provides intra-class variance, while the cyclic partition ensures the preservation of the data's sequential pattern. A different approach to data augmentation that has been used with signal data is to create multiple datapoints by slicing each datapoint using a sliding window, as in [36]. This approach is suitable when class discriminators are present in similar forms throughout the entire sequence (e.g. an EEG signal that captures a lingering emotion, as in [36]) because the sliding window breaks up the data sequence into segments that are essentially equivalent in terms of predictive power. We do not use this method because of the nature of confusion, which makes it unwarranted to assume that indicators of confusion are present to the same degree throughout the entire signal.

A difficulty in using raw ET data collected at a high sampling rate is the length of the resulting sequences (as discussed in the next section). The cyclical split also helps with this issue because it reduces the length of each datapoint by a factor of four.

4 Models and Approach

This section describes the intuition behind using an RNN and a CNN on ET data and combining them in a way that is appropriate for the data. Due to the relatively small size of our dataset, in each case, it was important to minimize model complexity. Thus, reducing the number of learnable parameters to avoid overfitting was the driving force behind the various design choices described in this section.

4.1 RNNs

RNNs are a neural network variant especially suited for sequential data, such as ET data. We chose to investigate RNNs because of the nature of confusion itself. As an affective state, confusion doesn't occur instantly. Rather, it develops over a period of time as the brain uncovers discrepancies between its existing knowledge and what it observes and continues with subsequent attempts to resolve these discrepancies, until the person either resolves their confusion or gives up [9].

Confusion may develop based on events further back in time, in a strictly local sequence, or as a combination of both. RNNs are able

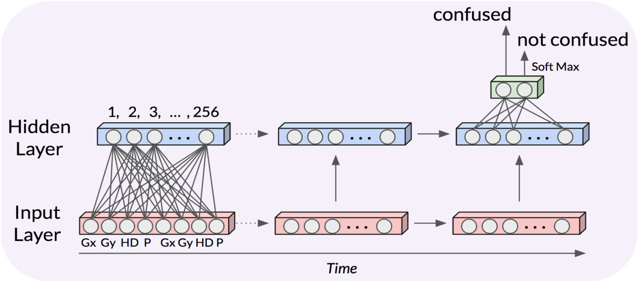


Figure 4: The RNN architecture used in this paper

to handle such varied temporal dependencies, which is why it was chosen for this investigation.

Two variations of RNN have become popular for modelling temporal data: Long-Short Term Memory (LSTM) networks and Gated Recurrent Units (GRU). LSTMs are gated RNNs that use self-loops to facilitate the learning of long-term dependencies while also ensuring long-term gradient flow [14]. A GRU is essentially a simplified LSTM that reduces the number of gates and thus the number of learnable parameters [7]. Because of this reduction in parameters, we chose to use the GRU as the RNN sub-model in our architecture.

Based on evidence that for RNNs, neural network depth in the traditional sense (i.e. the number of layers) is not as important as recurrent depth for classification tasks [40], we limit our model to a single layer, thus limiting complexity. Figure 4 visualizes the GRU architecture we use. We chose a hidden layer of 256 units during hyperparameter tuning using common heuristics [13]. The GRU's hidden layer is fully connected to each of the input elements, namely the values of an ET sample for a given time step. At each time step, the GRU produces an output value interpreted as a probability for the confused/not confused class using the softmax equation, and this output at the end of a datapoint is the prediction of confusion or not for the corresponding trial (see Figure 4, right).

While there is no fixed length on which RNNs must operate, in practice sequences should be shorter than 400 steps (and often much shorter) [28]. Even after the cyclical split described in section 3.2, 50% of our datapoints have a length longer than 600 ET samples. We address this issue by considering only 5 seconds of relevant ET samples before a confusion self-report (or placeholder for no confusion) in each data item since Lallé et al. (2016) found this interval to perform as well as considering the full length of data back to the start of the trial[1].

4.2 CNNs

Another way in which a sequence of raw ET samples can be represented is as a scanpath image. Given the coordinates in the raw eye-tracking samples, these images are created to contain the path made by the user's gaze over the sequence, where dots represent individual samples, and connecting lines represent the transitions between two samples (shown in Figure 5)[2]. The temporal information of the gaze sequence is lost, but visuospatial information comes to the forefront. We leverage a CNN architecture to predict confusion in our datapoints from the scanpath images of the corresponding sequences of raw ET samples.

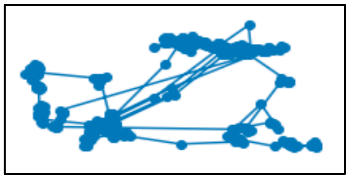


Figure 5: Example scanpath image

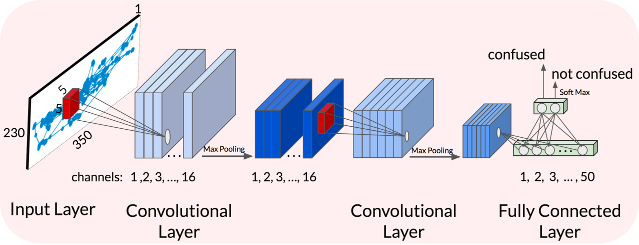


Figure 6: The CNN architecture used in this paper

Because the sequence length does not change the size of the corresponding scanpath image, we use full sequences as input to the CNN input, as opposed to the 5 second segments used for the RNN. This allows us to leverage the full information of the user's gaze activity over the trial, regardless of how long it lasts. Although this might seem unwarranted given that Lallé et al. [24] found no added benefit when considering full sequence vs 5 second ones, their comparison was based on a uniform data representation consisting of summary statistics of gaze, pupil size, and head distance. Here we combine temporal information on 5 seconds of data, with a different representation focusing on the user's complete attention patterns prior to the confusion report (or pivot point).

Scanpaths are rather different from the images that CNNs are typically used for. For instance, CNNs have been successfully used with natural images containing a hierarchy of parts (e.g. a car's wheels and their subcomponents) as well as properties such as colour and texture, that CNNs model in their various layers [11]. No such hierarchies nor properties appear in a scanpath image. Instead, scanpaths capture a strictly visual and spatial (visuospatial) representation of gaze data where dots visualize where given gaze samples are located in relation to the others, the density of dots indicates the amount of user attention to a specific area, and connecting lines indicate the relative length and frequency of the saccades to and from that area. The image as a whole provides information about the user's overall attention over the interface.

A CNN can capture these relevant scanpath characteristics. We chose some of the hyperparameters of the CNN architecture (shown in Figure 6) with knowledge of our data and the CNN model class in mind while balancing the competing goal of minimizing learnable model parameters to prevent overfitting our

[1] These 5 seconds exclude the one second just before the report, as discussed in section 3.1 and Figure 2.

[2] Such images are commonly available via the eye-tracker's software, based on fixations.

small dataset. The choices made to balance these competing goals are as follows.

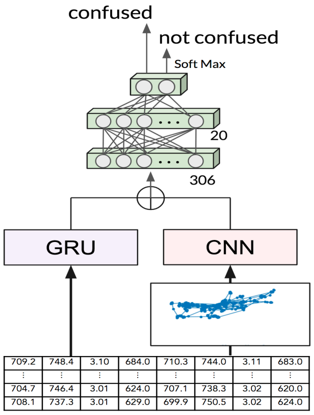
1. As scanpaths consist of dots and lines, the deep hierarchies associated with natural images are not required. As such, our CNN consists of two convolutional layers (see Figure 6) of 16 and 6 channels, respectively. We determined these hyperparameters by increasing each from one until validation set performance decreased. Having two layers makes sense, as this is enough to extract simple visual features while avoiding the additional parameters that come from unnecessary layers and overfitting to patterns unique to the training data.
2. Although having only two convolutional layers is advantageous for the reasons described above, it prevents the model from building a large receptive field (important for capturing local information) via depth. To balance this, we use a slightly larger kernel size than is common (5x5 vs. the more common 3x3) in order to increase the receptive field's width directly (kernel is the dark red square shown over the input image and in a subsequent layer in Figure 6). Though a larger kernel size requires more weights, the increase is much less than would come from additional convolutional layers, thus satisfying our goal of building a small model.
3. We make two changes related to the input. First, as colour has no meaning in a scanpath image, we use a single grayscale input channel, to further reduce the number of parameters. Second, as our images do not contain fine or nuanced textures (like the hair of an animal for instance), high resolution is not important. Thus, we downsize the images by a factor of 6, to reduce the dimensions and parameters of each convolutional layer. This single-channel low-resolution input image (and its dimensions) are denoted as the input layer in Figure 6.

Finally, the CNN contains a 50-unit hidden layer connecting the output of the convolutions with the class predictions in the output layer (right of Figure 6). The size of this hidden layer was chosen as a reasonable progression between the numerous neurons resulting from the convolutions and the two-unit output layer.

4.3 VTNet

Having developed the intuition behind using each of the RNN and CNN on eye-tracking data, here we describe an architecture to leverage the strength of both models together. In our approach, each of the CNN and RNN takes a different representation of the same data sequence and processes it independently. This model (visuospatial-temporal network, or VTNet from now on) is shown in Figure 7. The GRU's 256-unit hidden state that results from processing a datapoint is concatenated with the 50 element vector output of the CNN resulting from processing the corresponding scanpath, creating a single vector of size 306. This combined output vector is fully connected to a simple neural network with one hidden layer, (to create a differentiable classifier with minimal additional parameters), which classifies the input as either confused or not confused. The entire model is then learned end-to-end as a single unit.

Our hypothesis in creating the VTNet was that having a model that can process a multimodal representation of ET data will enhance its predictive abilities by having access to sequential information close to any confusion report as well as spatial information from earlier parts of the trial. This may be beneficial to predicting confusion if there are signals that occur earlier in the trial than the last 5 seconds available to the GRU.



709.2	748.4	3.10	684.0	710.3	744.0	3.11	683.0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
704.7	746.4	3.01	624.0	707.1	738.3	3.02	620.0
708.1	737.3	3.01	629.0	699.9	750.5	3.02	624.0

Figure 7: The VTNet architecture

Previous architectures that combine CNNs and RNNs (see section 2), do so by feeding input to both sub-models at each time step and are thus not suitable for our learning task. This is because processing a scanpath image as it develops over time through a CNN to extract features for RNN input gives no more information than that already available in the raw sequence. Instead, ET data has the property that a given temporal sequence of data can be represented in a single frame in a meaningful way. That is, a given image of a user's entire scanpath contains information about the aggregate spatial eye-movement.

4.4 Implementation

All of our neural network-based models are implemented using PyTorch (https://github.com/sdv4/VTNet_ICMI2020). We use negative log-likelihood as our loss function, with the Adam optimizer [21]. We limit training to 100 epochs, employing linear learning rate decay and early stopping to end training when validation performance stops improving. We train our models using a single Nvidia GTX 1080 GPU.

5 Evaluation

We first determine how the GRU model (the RNN component of VTNet) performs compared to the RF approach in [24] (section 5.2). We begin with this comparison because the RNN is the most intuitive neural model to use with raw ET data. Next, in Section 5.3 we evaluate the performance of the CNN architecture described in section 4.2 on scanpath images and determine whether combining it with the GRU in the VTNet architecture is more effective than its constituent parts are alone, as hypothesized in section 4.3.

5.1 Experimental Setup

Model performance is evaluated with confused class accuracy (Conf.) and not confused class accuracy (N. Conf.), which are the proportion of confused and not confused tasks correctly identified as such, respectively. Because of the dataset's class imbalance, both metrics together are more meaningful than accuracy alone. For instance, a 98% overall accuracy could be achieved by simply classifying everything as not confused, but not capturing any instance of confusion, thus preventing the real-time provision of support when confusion does arise. We also report the mean of confused and not confused class accuracies as combined accuracy; a unified measure of performance.

All models are evaluated using 10 runs of 10-fold cross-validation (giving 100 iterations of CV in total) to reduce fluctuations in the results due to the random selection of folds. All results reported in the next section are the average of the 10 runs of 10-fold CV. Further, cross-validation is done across users so that no user contributes data points to both the training and test sets of a given fold, thus measuring model performance on unseen users. Cross-validation is also stratified so that the distribution of confusion data points in each fold is kept similar to that of the dataset as a whole.

For the RF model, nested CV (i.e., further cross-validation on each training set) was used for feature selection, hyperparameter tuning, and to choose the decision threshold that maximizes confused and not confused class accuracies[3]. For the deep learning models, using nested CV would be computationally onerous. Instead, for each of the 100 iterations of CV, we randomly select 20% of the data as a validation set for hyperparameter tuning and decision threshold setting. Note that contrary to the nested CV, the validation set is holdout data that is not re-added to the training set for a final round of training prior to evaluation on the test set. This effectively results in the DL models being trained on 20% less data than the RF model.

To address the imbalance between confused and not confused datapoints in the dataset, Lallé et al., (2016) [24] used Synthetic Minority Oversampling Technique (SMOTE) [6] for their RF model but recall that their model was not learning from ET data sequences. SMOTE is not generally suitable to augment sequences, because it measures similarity between samples by Euclidean distance, which is a bad match for long and temporally misaligned pairs [12]. However, preliminary experiments showed that SMOTE increased GRU performance with our data, possibly because we limit sequence length to 5 seconds worth of samples, and because confusion self-reports may provide an anchor that maintains a degree of temporal alignment in our sequences. Thus, for evaluating the performance the GRU when used on its own (Section 5.2) and for the RF model, classes in the training sets are balanced by first using SMOTE to increase the size of the minority class (confused) by 200% and then randomly down-sampling the majority class (as was done in [24]), resulting in approximately 1350 confused and 1350 not confused datapoints.

We cannot use SMOTE when evaluating the CNN nor with the VTNet that includes it (Section 5.3), because we use the full ET sequences to produce the scanpaths and as mentioned, SMOTE does not work well when having substantially longer sequences [12]. Thus, for these models, we just down-sample the majority class to achieve class balance, which reduces the number of non-confused items to approximately 450 (matching the number of confused items). Validation and/or test sets are left unbalanced in all models, so as to evaluate the models on data reflecting the realistic class distribution of the original dataset.

Model	*Conf.*	*N. Conf.*	Combined
RF	0.53	**0.80**	0.67
GRU	**0.75**	**0.80**	**0.78**

Table 1: Test set performance of GRU and RF

5.2 Results of Comparing GRU and RF

Comparing the performance of the GRU and RF models (Table 1), shows that GRU outperforms the RF classifier in both confused and combined accuracies, with no change in not confused class accuracy. The GRU achieves a combined accuracy of 0.78, compared to the 0.67 achieved by the RF. We test this result with an independent samples t-test, which shows that the difference is statistically significant (t_{18} = 6.28, p < .001). The difference in confused accuracy is also significant (t_{18} = 6.22, p < .001), with a substantial 41.5% improvement over the not confused accuracy of the RF model. These results allow us to conclude that the GRU outperforms the RF in classifying confusion with this dataset, where the impact of the GRU is specifically in improving confused class accuracy, namely detecting confusion when it occurs, with no loss in the accuracy of predicting when a user is not confused.

In [24], the authors experimented with combining ET data and interaction data based on the interface actions available in the ValueChart (see section 3.1) to train their model. This combination gave them their best results, namely 0.61 confused accuracy and 0.926 not confused accuracy, for a combined accuracy of 0.768. With this additional data modality, the RF still doesn't perform better than the GRU trained only on ET data. This result is especially encouraging when we consider that the GRU is trained on 20% less data (the portion held out as the validation set). It should be noted that we also experimented with including interaction data in our approach, by adding information of mouse clicks to the vectors of sequential data fed to the GRU. However, adding this interaction data generated no significant improvement, likely because the number of these events is sparse in comparison to the number of samples in a given sequence. A more suitable way to include interaction data would be to include the mouse coordinates at each time a sample is collected. This would give a fine-grained stream of interaction data at a level of granularity similar to that found in the raw eye-tracking sample. However, tracking of mouse coordinates was not available for the dataset used in this investigation.

5.3 Results of Comparing GRU, CNN, and VTNet

After establishing the superiority of the GRU over the RF model in classifying confusion, we evaluate the CNN as an independent model and then the performance of the VTNet model that combines the two. The result of this comparison is summarized in Table 2. The VTNet has been trained with the same hyperparameter configuration as its corresponding sub-models. We see that for all three measures (confused accuracy, not

[3] This is done by choosing the threshold closest to the (0,1) point on the Receiver Operating Characteristic (ROC) curve.

Model	*Conf.*	*N. Conf.*	Combined
GRU	0.75	0.80	0.78
CNN	0.73	0.80	0.77
VTNet	**0.79**	**0.84**	**0.82**

Table 2: Test set performance of neural models

confused accuracy, and combined accuracy) the VTNet outperforms both the GRU and the CNN. One-way ANOVA with classifier type (VTNet, GRU, and CNN) as the factor shows a significant effect on all three measures (combined: $F_{3,36} = 47.59$, $p < .001$, $\eta_p^2=.27$; confused: $F_{3,36} = 39.74$, $p < .001$, $\eta_p^2= .76$; not confused: $F_{3,36} = 9.25$, $p < .001$, $\eta_p^2=.33$). Post hoc testing via Tukey HSD (which adjusts for multiple comparisons) shows that for all three measures, the difference is statistically significant between VTNet and both GRU and CNN, with no significant difference between the latter two. With this, we conclude that VTNet surpasses the performance of both constituent parts and is thus an effective model for classifying confusion from our ET data.

VTNet achieves a 79% confused class accuracy, which represents a 49% increase over the original RF model. It is also the only one of the three deep learning models to increase not confused class accuracy (reaching 84%), suggesting that combining temporal and visuospatial information from ET data manages to capture patterns pertaining to the absence of confusion that go otherwise undetected. That fact that the VTNet does not have SMOTE augmented data, yet still outperforms the GRU with augmented data, shows that there is a strong signal for confusion in the scanpath images, which complements well the temporal information captured by the GRU. This suggests that additional confusion signal is present further back in the trial than the 5 seconds processed by the GRU, contrary to that found in [24].

The performance of the VTNet model is also higher than other published approaches to predicting confusion using RNNs in a different context, namely leveraging the interaction data of users while they study with ITSs [4, 18]. Neither of these previous works reports positive or negative class accuracies (i.e. confused and not confused, in our case), but both report Area Under the Curve (AUC) for the model's ROC, namely an AUC of 0.57 for [4] and AUC of 0.72 for [18]. By comparison, we achieve an AUC of 0.84 with VTNet and eye-tracking data.

6 Conclusions and Future Work

In this paper, we presented a novel approach that leverages deep learning for detecting user confusion from raw sequences of eye-tracking (ET) data. Our work contributes to the research on automatic detection of user affective states, with the long-term goal of creating intelligent interactive systems that can respond to these states to improve user experience. We focus on user confusion because it is a state that is well-known to affect user satisfaction and performance with interactive systems (e.g., [27]), thus it would be highly valuable to empower such systems to detect confusion and provide appropriate interventions to resolve it.

The approach we presented in this paper to detect user confusion from ET data combines the strength of CNNs in spatial reasoning with the strength of RNNs in temporal reasoning. The resulting model (VTNet) outperforms its constituent models considered on their own when tested on a dataset capturing episodes of confusion for users interacting with a visualization-based interactive system for decision support (ValueChart). VTNet also largely outperforms a previous model based on Random Forests, on the same dataset [24], bringing a 22% increase in combined confused and not confused class accuracies, with the bulk of the increase (49%) being in detecting confusion when it occurs (79% accuracy) which is remarkable considering that our dataset contained only 2% datapoints for confusion.

Deep learning has proven very effective in domains with large datasets, showing 16-23% improvements when initially applied to speech recognition and a 41% reduction in error rate when applied to object recognition [2]. Our results provide encouraging evidence that deep learning can be useful even with the smaller datasets usually available for predictive tasks involving hard-to-collect interaction data (e.g., ET data) and complex user states (e.g., affective reactions). As such, our work extends existing preliminary work on using deep learning approaches for predicting user affect from user interface actions, by predicting the specific affective state of confusion from ET data.

Our approach also extends previous work on combining CNNs and RNNs, by integrating the two in a manner that suits the specific sequential and visuospatial nature of the ET data, where a temporal sequence of raw samples in a given timeframe can also be represented as a single visual scanpath for that timeframe. Our results provide evidence that there is a benefit to modelling sequential data local to a confusion episode, while having access to an image of the gaze activity over a longer span of interaction prior to confusion, indicating that there are important yet distinct signals in both representations, which when combined, give stronger results than either signal considered alone.

Moving forward, we will explore methods for increasing the VTNet performance, such as increasing receptive field size via dilated convolutions. We will integrate our predictors of confusion into ValueCharts, and investigate responses designed to mitigate confusion as it is detected during a user's interaction with the system. We will also investigate whether our results generalize to predicting confusion in other interactive tasks, and to predict other states relevant to ascertain user experience with an interactive task. Along these lines, we plan to test the VTNet approach on other ET datasets that have been used to predict user states such as learning [19], affective valence [23], as well as early stages Alzheimer's disease (to appear). Finally, we believe that our VTNet approach could be applied to other data modalities that have been used for affect detection. For instance, we are interested in looking at speech and EEG data, where the VTNet could be adapted to learn from the combination of the temporal signals with the related spectrogram (for speech) or with a heatmap representation of the signal over the brain for EEG.

ACKNOWLEDGMENTS

This research was funded by the Natural Sciences and Engineering Research Council of Canada, NSERC grants RGPIN-2016-04611 and RTI-2019-00711.

REFERENCES

[1] Mohamed R. Amer, Behjat Siddiquie, Colleen Richey, and Ajay Divakaran. 2014. Emotion Detection in Speech Using Deep Networks. In 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 3724–3728.

[2] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation Learning: A Review and New Perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence35, 8 (2013), 1798–1828.

[3] Robert Bixler and Sidney D'Mello. 2015. Automatic Gaze-based Detection of Mind Wandering with Metacognitive Awareness. In International Conference on User Modeling, Adaptation, and Personalization. Springer, 31–43.

[4] Anthony F. Botelho, Ryan S. Baker, and Neil T. Heffernan. 2017. Improving Sensor-free Affect Detection Using Deep Learning. In International Conference on Artificial Intelligence in Education. Springer, 40–51.

[5] Giuseppe Carenini and John Loyd. 2004. ValueCharts: Analyzing Linear Models Expressing Preferences and Evaluations. In Proceedings of the Working Conference on Advanced Visual Interfaces. 150–157.

[6] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research (2002), 321–357.

[7] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau,Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. arXiv preprint arXiv:1406.1078(2014).

[8] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. 2015. Long-term Recurrent Convolutional Networks for Visual Recognition and Description. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2625–2634.

[9] Sidney D'Mello, Blair Lehman, Reinhard Pekrun, and Art Graesser. 2014. Confusion can be Beneficial for Learning. Learning and Instruction (2014), 153–170.

[10] Samira Ebrahimi Kahou, Vincent Michalski, Kishore Konda, Roland Memisevic, and Christopher Pal. 2015. Recurrent Neural Networks for Emotion Recognition in Video. In Proceedings of the 2015 ACM on International Conference on Multimodal Interaction. 467–474.

[11] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 580–587.

[12] Zhichen Gong and Huanhuan Chen. 2016. Model-based Oversampling for Imbalanced Sequence Classification. In Proceedings of the 25th ACM International on Conference on Information and Knowledge Management. 1009–308.

[13] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. Deep Learning. MIT Press.

[14] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-term Memory. Neural Computation 9, 8 (1997), 1735–1780.

[15] Stephen Hutt, Joseph F. Grafsgaard, and Sidney K D'Mello. 2019. Time to Scale: Generalizable Affect Detection for Tens of Thousands of Students Across an Entire School Year. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems. 1–14.

[16] Aulikki Hyrskykari, Päivi Majaranta, Antti Aaltonen, and Kari-Jouko Räihä. 2000. Design Issues of iDICT: A Gaze-assisted Translation Aid. In *Proceedings of the 2000 Symposium on Eye tracking Research & Applications*. 9–14.

[17] Natasha Jaques, Cristina Conati, Jason M Harley, and Roger Azevedo. 2014. Predicting Affect from Gaze Data During Interaction with an Intelligent Tutoring System. In International Conference on Intelligent Tutoring Systems. Springer, 29–38.

[18] Yang Jiang, Nigel Bosch, Ryan S. Baker, Luc Paquette, Jaclyn Ocumpaugh, Juliana Ma Alexandra L. Andres, Allison L. Moore, and Gautam Biswas. 2018. Expert Feature-engineering vs. Deep Neural Networks: Which is Better for Sensor-free Affect Detection? In International Conference on Artificial Intelligence in Education. Springer, 198–211.

[19] Samad Kardan and Cristina Conati. 2015. Providing Adaptive Support in an Interactive Simulation for Learning: An Experimental Evaluation. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems.3671–3680.

[20] Andrej Karpathy and Li Fei-Fei. 2015. Deep Visual-semantic Alignments for Generating Image Descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 3128–3137.

[21] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for Stochastic Optimization. arXiv preprint arXiv:1412.6980.

[22] Sander Koelstra, *et al.*, 2012. DEAP: A Database for Emotion Analysis Using Physiological Signals. IEEE Transactions on Affective Computing 3.1: 18-31.

[23] Sébastien Lallé, Cristina Conati, and Roger Azevedo. 2018. Prediction of Student Achievement Goals and Emotion Valence During Interaction with Pedagogical Agents. In Proceedings of the 17th International Conference on Autonomous Agents and Multi Agent Systems, 1222–1231.

[24] Sébastien Lallé, Cristina Conati, and Giuseppe Carenini. 2016. Predicting Confusion in Information Visualization from Eye Tracking and Interaction Data. In IJCAI. 2529–2535.

[25] Sukwon Lee, Sung-Hee Kim, Ya-Hsin Hung, Heidi Lam, Younah Kang, and Ji Soo Yi. 2015. How do People Make Sense of Unfamiliar Visualizations? A Grounded Model of Novice's Information Visualization Sense Making. IEEE Transactions on Visualization and Computer Graphics, 1, 499–508.

[26] Xiang Li, Dawei Song, Peng Zhang, Guangliang Yu, Yuexian Hou, and Bin Hu. 2016. Emotion Recognition from Multi-channel EEG Data Through Convolutional Recurrent Neural Network. In 2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), pp. 352-359.

[27] Sucheta Nadkarni and Reetika Gupta. 2007. A Task-based Model of Perceived Website Complexity. Mis Quarterly (2007), 501–524.

[28] Daniel Neil, Michael Pfeiffer, and Shih-Chii Liu. 2016. Phased LSTM: Accelerating Recurrent Network Training for Long or Event-based Sequences. In Advances in neural information processing systems. 3882–3890.

[29] Joshua Newn, Ronal Singh, Fraser Allison, Prashan Madumal, Eduardo Velloso, and Frank Vetere. 2019. Designing Interactions with Intention-Aware Gaze-Enabled Artificial Agents. In Human-Computer Interaction – INTERACT 2019 (Lecture Notes in Computer Science). Springer International Publishing, Cham, 255–281.

[30] Guido Pusiol, Andre Esteva, Scott S. Hall, Michael Frank, Arnold Milstein, and Li Fei-Fei. 2016. Vision-based Classification of Developmental Disorders Using Eye-movements. In International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, 317–325.

[31] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. 2015. Unsupervised Learning of Video Representations Using LSTMs. In International Conference on Machine Learning. 843–852.

[32] Ben Steichen, Cristina Conati, and Giuseppe Carenini. 2014. Inferring Visualization Task Properties, User Performance, and User Cognitive Abilities from Eye-gaze Data. ACM Transactions on Interactive Intelligent Systems (TiiS'14), 1–29.

[33] Alexandria K. Vail, Tadas Baltrušaitis, Luciana Pennant, Elizabeth Liebson, Justin Baker, and Louis-Philippe Morency. "Visual attention in schizophrenia: Eye contact and gaze aversion during clinical interactions." In 2017 Seventh International Conference on Affective Computing and Intelligent Interaction (ACII), pp. 490-497. IEEE, 2017.

[34] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In International Conference on Machine Learning. 2048–2057.

[35] J.S. Yi. 2008. Visualized Decision Making: Development and Application of Information Visualization Techniques to Improve Decision Quality of Nursing Home Choice, Georgia Institute of Technology. PhD Thesis.

[36] Yilong Yang, Qingfeng Wu, Ming Qiu, Yingdong Wang, and Xiaowei Chen, 2018. Emotion Recognition from Multi-channel EEG Through Parallel Convolutional Recurrent Neural Network. International Joint Conference on Neural Networks (IJCNN).

[37] Haiyang Yu, Zhihai Wu, Shuqin Wang, Yunpeng Wang, and Xiaolei Ma. 2017. Spatiotemporal Recurrent Convolutional Networks for Traffic Prediction in Transportation Networks. Sensors'17.

[38] Dalin Zhang, Lina Yao, Xiang Zhang, Sen Wang, Weitong Chen, Robert Boots, and Boualem Benatallah. 2018. Cascade and Parallel Convolutional Recurrent Neural Networks on EEG-based Intention Recognition for Brain Computer Interface. In Thirty-Second AAAI Conference on Artificial Intelligence.

[39] Zhang, Lei, Shuai Wang, and Bing Liu. 2018. Deep Learning for Sentiment Analysis: A Survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 8, no. 4: e1253.

[40] Saizheng Zhang, Yuhuai Wu, Tong Che, Zhouhan Lin, Roland Memisevic, Russ R Salakhutdinov, and Yoshua Bengio. 2016. Architectural Complexity Measures of Recurrent Neural Networks. In Advances in Neural Information Processing Systems.1822–1830.

[41] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 2015. Appearance-based Gaze Estimation in the Wild. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 4511–4520.