Generative Modeling of Synthetic Eye-tracking Data: NLP-based Approach with Recurrent Neural Networks

Mahmoud Elbattah[1], Jean-Luc Guérin[1], Romuald Carette[1,2], Federica Cilia[3] and Gilles Dequen[1]
[1]*Laboratoire MIS, Université de Picardie Jules Verne, Amiens, France*
[2]*Evolucare Technologies, Villers Bretonneux, France*
[3]*Laboratoire CRP-CPO, Université de Picardie Jules Verne, Amiens, France*

Keywords: Eye-tracking, Machine Learning, Recurrent Neural Networks, NLP.

Abstract: This study explores a Machine Learning-based approach for generating synthetic eye-tracking data. In this respect, a novel application of Recurrent Neural Networks is experimented. Our approach is based on learning the sequence patterns of eye-tracking data. The key idea is to represent eye-tracking records as textual strings, which describe the sequences of fixations and saccades. The study therefore could borrow methods from the Natural Language Processing (NLP) domain for transforming the raw eye-tracking data. The NLP-based transformation is utilised to convert the high-dimensional eye-tracking data into an amenable representation for learning. Furthermore, the generative modeling could be implemented as a task of text generation. Our empirical experiments support further exploration and development of such NLP-driven approaches for the purpose of producing synthetic eye-tracking datasets for a variety of potential applications.

1 INTRODUCTION

The human eyes represent a rich source of information not only reflecting on the emotional or mental conditions, but also for understanding the functioning of our cognitive system. The eye gaze serves as an appropriate proxy for learning the user's attention or focus on context (Zhai, 2003). In this regard, the eye-tracking technology has come into prominence to support the study and analysis of gaze behaviour in many respects.

Eye-tracking refers to the process of capturing, tracking and measuring the absolute point of gaze (POG) and eye movement (Majaranta and Bulling, 2014). The eye-tracking field notably has a long history that dates back to the 19th century. The early development is credited to the French ophthalmologist Louis Javal from the Sorbonne University. In his seminal research that commenced in 1878, Javal made the original observations of fixations and saccades based on the gaze behaviour during the reading process (Javal, 1878, 1879). Subsequently, Edmund Huey built a primitive eye-tracking tool for analyzing eye movements while reading (Huey, 1908). More advanced implementations of eye-tracking were developed by (Buswell, 1922, 1935). Photographic films were utilized to record the eye movements during looking at a variety of paintings. The eye-tracking records included both direction and duration of movements.

The technological advances continued to evolve towards the nearly universal adoption of video-based eye-tracking. Video-based techniques could be classified into: 1) Video-based tracking using remote or head-mounted cameras, and 2) Video-based tracking using infrared pupil-corneal reflection (P-CR) (Majaranta and Bulling, 2014). Further, recent developments discussed the use of Virtual Reality-based methods for eye-tracking (e.g. Meißner et al., 2019). Eye-tracking has been utilized in a multitude of commercial and research applications. Examples include marketing (e.g. Khushaba et al., 2013), psychology (e.g. Mele and Federici, 2012), product design (Khalighy et al. 2015), and many others.

However, the scarce availability or difficulty of acquiring eye-tracking datasets presents a key challenge. While having access to image or time series data, for example, has been largely facilitated thanks to repositories such as ImageNet (Deng et al., 2009) and UCR (Dau et al. 2019). The eye-tracking literature still lacks such large-scale repositories.

Elbattah, M., Guérin, J., Carette, R., Cilia, F. and Dequen, G.
Generative Modeling of Synthetic Eye-tracking Data: NLP-based Approach with Recurrent Neural Networks.
DOI: 10.5220/0010177204790484
In *Proceedings of the 12th International Joint Conference on Computational Intelligence (IJCCI 2020)*, pages 479-484
ISBN: 978-989-758-475-6

In this respect, this study explores the use of Machine Learning (ML) for generating synthetic eye-tracking data. Our approach attempts to borrow methods from the Natural Language Processing (NLP) domain to transform and model eye-tracking sequences. As such, the eye-tracking records could be represented as textual strings describing series of fixations and saccades. A long short-term memory (LSTM) model is employed for the generative modeling task. In summary, this paper attempts to present the following contributions:

- Compared to literature, the study explores a different NLP-driven approach, which models eye-tracking records as textual sequences. Such approach for generating synthetic eye-tracking data has not been proposed yet, to the best of our knowledge.
- In a broader context, it is practically demonstrated how NLP methods can be utilized to transform high-dimensional eye-tracking data into an amenable representation for the development of ML models.

2 RELATED WORK

The literature is replete with contributions that introduced methods to synthesize or simulate the human eye movements. Those methods could be broadly classified into two approaches. On one hand, the larger part of efforts sought to craft algorithmic models based on characteristics driven from the eye-tracking research. On the other hand, ML-based methods were developed to this end. Though the present work falls under the latter category, we provide representative studies from the both sides. The review is unavoidably selective rather than exhaustive due to the limitations of space.

2.1 Statistical Modeling for Generating Synthetic Eye-tracking Data

In an interesting application, it was proposed to synthesize the eye gaze from an input of head-motion sequence (Ma and Deng, 2009). Their method was based on statistically modeling the natural conjugation of gaze and head movements. Likewise, (Duchowski et al. 2016) developed a stochastic model of gaze. The synthetic data could be parameterised by a set of variables including sampling rate, micro-saccadic jitter, and simulated measurement error.

In similar vein, there have been numerous contributions for developing gaze models to generate realistic eye movement in animations or virtual environments. To name a few, statistical models of eye-tracking data were implemented based on the analysis of eye-tracking videos (Lee, Badler, and Badler, 2002). The models were aimed to reflect the dynamic characteristics of natural eye movements (e.g. saccade amplitude, velocity).

Another framework was proposed to automate the generation of realistic eye and head movements (Le, Ma, and Deng, 2012). It was basically aimed to separately learn inter-related statistical models for each component of movement based on a pre-recorded facial motion dataset. The framework also considered the subtle eyelid movement and blinks. Further contributions could be found in this regard (e.g. Oyekoya, Steptoe, and Steed, 2009; Steptoe, Oyekoya, and Steed, 2010; Trutoiu et al. 2011).

2.2 Machine Learning for Generating Synthetic Eye-tracking Data

In contrast to the above-mentioned studies, recent efforts experimented purely ML-based approaches. Eye-trackers typically produce abundant amounts of eye-gaze information. A few minutes of operating time would output thousands of records regarding the gaze position and eye movements. With such large amounts of data, ML could be an ideal path to extrapolate algorithms from data exclusively.

For instance, a fully Convolutional Neural Network (CNN) was used for the semantic segmentation of eye-tracking data (Fuhl, 2020). In conjunction with a variational auto-encoder, the CNN model was utilized further for the reconstruction and generation of eye movement data. Using a convolutional-recurrent architecture, (Assens et al. 2018) developed a framework named as ‘PathGAN’. With an approach of adversarial training, the PathGAN presented an end-to-end model for predicting the visual scanpath.

In another application with Recurrent Neural Networks (RNN), a real-time system for gaze animation was developed (Klein et al., 2019). Both motion and video data were used to train the RNN model, which could predict the motion of the body and the eyes. The data was captured by a head-mounted camera. Likewise, a sequence-to-sequence LSTM-based architecture was applied to generate synthetic eye-tracking data (Zemblys, Niehorster, and Holmqvist, 2019).

3 DATA DESCRIPTION

The dataset under consideration was collected as part of an autism-related study (Elbattah et al., 2019). Eye-tracking methods have been widely utilised in the Autism Spectrum Disorder (ASD) context, since abnormalities of eye gaze are largely recognised as the hallmark of autism (e.g. Guillon et al., 2014).

The eye-tracking experiments were based on a group of 59 children. The age of participants ranged from 3 to 12 years old. The participants were organized into two groups as: i) Typically Developing (TD), and ii) ASD. The participants were invited to watch a set of photographs and videos, which included scenarios to stimulate the viewer's gaze. The average duration of eye-tracking experiments was about 5 minutes.

They used a SMI Red-M eye tracker with 60 Hz sampling rate. The eye-tracking records contained three categories of eye movements including fixation, saccade, and blink. A fixation is the brief moment of pausing the gaze on an object while the brain is performing the perception process. The average duration of fixation was estimated to be about 330 milliseconds (Henderson, 2003). While saccades include a constant scanning with rapid and short eye movements. Saccades include quick ballistic jumps of 2^{o} or longer, which continue for about 30–120 milliseconds each (Jacob, 1995). The output sequence of fixations and saccades is called a scanpath.

The original dataset was constructed over 25 eye-tracking experiments, which produced more than 2M records stored in CSV-structured files. Table 1 provides a simplified snapshot of the raw eye-tracking records. The records describe the category of movements and the POG coordinates over the experiment runtime. Specifically, each row represents a point in the experiment timeline. As it appears, the eye-tracker's timing was 20 ms roughly. Due to limited space, many variables are not included in the table (e.g. pupil size, pupil position).

Table1: A snapshot of eye-tracking records.

Timestamp [ms]	Eye Movement	POG-(X,Y) [px]	Pupil Diameter (Right, Left) [mm]
8005654.06	Fixation	1033.9, 834.09	4.3785, 4.5431
8005673.95	Fixation	1030.3, 826.08	4.4050, 4.5283
8005693.85	Saccade	1027.3, 826.31	4.4273, 4.6036
8005713.70	Saccade	1015.0, 849.21	4.3514, 4.5827
8005733.58	Saccade	613.76, 418.17	4.3538, 4.5399

4 DATA TRANSFORMATION

As alluded earlier, the fixation-saccade sequences of eye-tracking records were basically considered as strings of text. As such, a set of NLP methods were applied in order to process and transform the raw eye-tracking dataset. The following sections explain the procedures of data transformation.

4.1 Extraction of Sequences

The raw dataset represented long-tailed sequences of fixations and saccades. Each sequence represented the output of an eye-tracking experiment for one of the participants. The dataset originally included 712 sequences. Raw sequences were very high dimensional including thousands or even hundreds of thousands of fixation-saccade elements. For example, considering an experiment of 5 minutes, with an eye-tracker of 20 ms resolution, would produce about 15K records (i.e. (5*60*1000)/20).

The initial procedure was to transform the raw sequences into a representation that can reduce that high dimensionality. To this end, the sequences were intially divided into smaller sub-sequences of fixed length (L). It was aimed to produce sequences of an adequate length, such that the LSTM model training could be more tractable. Eventually, the sequence length (L) was set as 20. All sub-sequences were labelled as the original full-length sequence.

4.2 Segmentation of Sequences

The extracted sequences can be merely viewed as text strings based on a binary set of words (i.e. fixation or saccade, and excluding blinks). To simplify the representation and processing of sequences, they were partitioned into fixed-size fragments. This can be more efficient for the tokenisation and encoding of sequences. The segmentation of sequences was partly inspired by the *K-mer* representation widely used in the Genomics domain (e.g. Chor et al., 2009). In comparable analogy, very long DNA sequences are broken down into smaller (k) sub-sequences to simplify the modeling and analysis tasks.

As such, a (K) number of fixations and saccades was grouped together to form fragments. For example, considering a sequence where L=20 and K=4, the sequence would be divided into 5 fragments (i.e. 20/4). Each fragment represents a combination of fixations and saccades occurring in a particular sequence. The fragments could be conceived as words as in a text sentence.

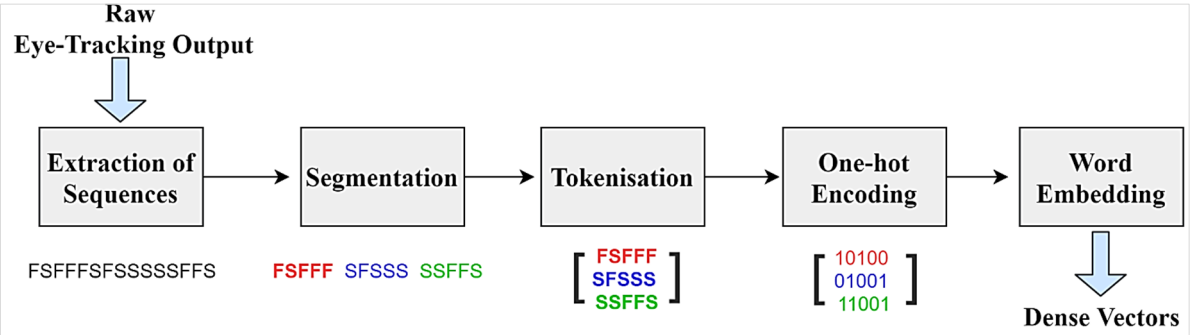

Figure 1: The pre-processing pipeline of raw eye-tracking data.

4.3 Tokenisation

The segmented sequences were in a suitable form for tokenisation, as commonly applied in NLP. The Keras library (Chollet, 2015) provides a convenient method for tokenisation, which was used for pre-processing the sequences.

Using the *Tokenizer* utility class, textual sequences could be vectorised into a list of integer values. Each integer was mapped to a value in a dictionary that encoded the entire corpus. The dictionary keys represented the vocabulary terms.

4.4 One-hot Encoding

Subsequently, tokens were represented as vectors by applying the one-hot encoding. It is a simple process that produces a vector of the length of the vocabulary with an entry for each word in the corpus. In this way, each word would be given a spot in the vocabulary, where the corresponding index is set to one. Keras also provides an easy-to-use APIs for applying the one-hot encoding. As such, the sequences eventually consisted of fragments of binary digits.

4.5 Word Embedding

The final step was to apply the *word embedding technique*, which is a vital procedure for making the sequences tractable for ML. The one-hot encoded vectors usually suffer from high dimensionality and sparsity, which makes the learning vulnerable to the curse of dimensionality. In this regard, embeddings were used to provide dense vectors of much lower dimensions compared to the encoded representation.

Keras provides a special Neural Network layer for the implementation of embeddings. The embedding layer can be basically regarded as a dictionary that maps integer indices into dense vectors. Based on integer inputs, it looks up these values in an internal dictionary, and returns the associated vectors. The embedding layer was used as the top layer of the generative model. Figure 1 illustrates the pipeline of pre-processing procedures.

5 EXPERIMENTS

Using L=20 and K=4, the dataset consisted of about 44K sequences. However, the experimental dataset included the ASD set only, which accounted for about ≈35% of the dataset. Ideally, the generative model would be utilised to generate synthetic samples of the minority class, which is the ASD in our case.

The LSTM approach was applied. LSTM models provide a potent mechanism for predictive and generative modeling as well. They learn data sequences (e.g. time series, text), and new plausible sequences can be synthetically generated. In our case, the goal was to learn text-like sequences of fixations and saccades, as explained before. Basically, the model was trained to predict the next word based on a sequence of preceding words.

The core component of the model was an LSTM layer of 50 cells. The LSTM layer was followed by a dense layer. The model architecture was decided largely empirically. The model was implemented using the *CuDNNLSTM* layer of Keras, which includes optimised routines for GPU computation. Figure 2 sketches the model architecture.

Figure 3 plots the model loss in training and validation over 20 epochs with 20% of the dataset used for validation. The Adam optimizer (Kingma and Ba, 2015) was used with its default parameters. As it appears, the model performance levelled off nearly after 10 epochs. Using the test set, the model could achieve about 75% accuracy of prediction. The experiments were run on the Google Cloud platform using a VM including a Nvidia Tesla P4 GPU, and 25GB RAM. The model was implemented using Keras and TensorFlow backend (Abadi et al., 2016).

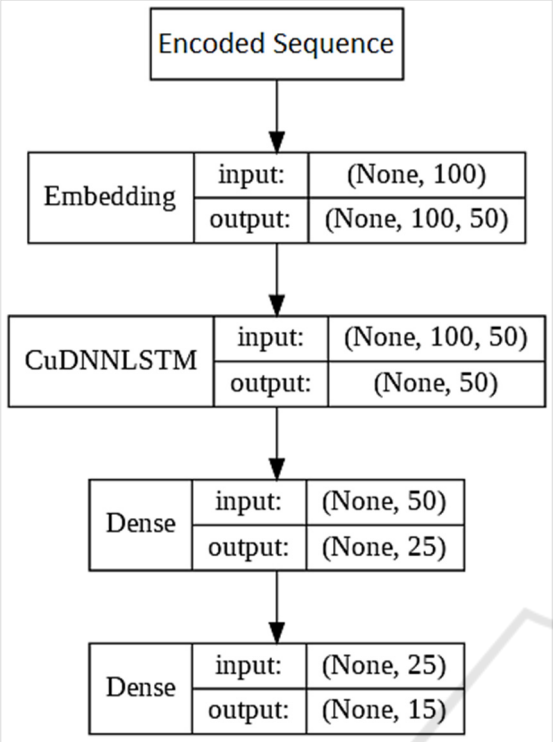

Figure 2: Model architecture.

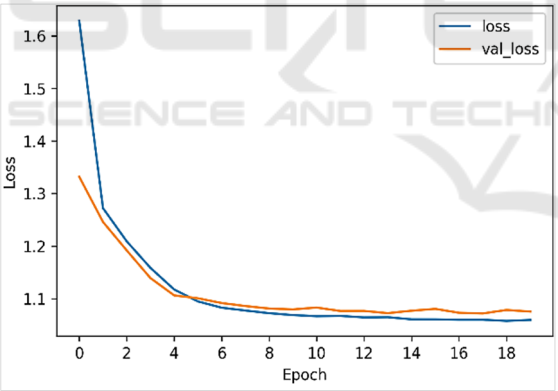

Figure 3: Model loss in training and validation sets.

The generation of synthetic sequences was experimented as follows. Initially, the trained LSTM model was saved. Keras allows to save the model into a binary HDF5 format. The saved model included the architecture along with the set of weight values.

After loading the model, a set of words were used as a seed input. The seed words were randomly sampled from the test set. Based on the seed input, the next word can be predicted. As such, the LSTM model could be used as a generative model to produce synthetic sequences. The process can be iteratively applied according to the desired sequence length. The experiments along with the model implementation are shared on our GitHub repository (Elbattah, 2020).

6 CONCLUSIONS

This paper presented an NLP-based approach for the generation of synthetic eye-tracking records. Using a sequence-based representation of the saccadic eye movement, eye-tracking records could be modelled as textual strings with an LSTM model.

The approach applicability was empirically demonstrated in our experiments, though using a relatively small dataset. It is conceived that the lack of open-access eye-tracking datasets could make our approach attractive for further studies. For instance, the generative model can serve as an alternative method for data augmentation in a wide range of eye-tracking applications.

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... & Kudlur, M. (2016). Tensorflow: A system for large-scale machine learning. *In Proceedings of the 12th {USENIX} Symposium on Operating Systems Design and Implementation* (pp. 265-283).

Assens, M., Giro-i-Nieto, X., McGuinness, K., & O'Connor, N. E. (2018). PathGAN: visual scanpath prediction with generative adversarial networks. *In Proceedings of the European Conference on Computer Vision (ECCV)*.

Buswell, G.T. (1922). *Fundamental reading habits: a study of their development*. American Psychological Association: Worcester, MA, USA.

Buswell, G.T. (1935). *How people look at pictures: a study of the psychology and perception in art*. University of Chicago Press, Chicago, IL, USA.

Chollet, F. (2015). Keras. GitHub Repository: https://github.com/fchollet/keras.

Chor, B., Horn, D., Goldman, N., Levy, Y., & Massingham, T. (2009). Genomic DNA k-mer spectra: models and modalities. *Genome Biology*, 10(10), R108.

Dau, H. A., Bagnall, A., Kamgar, K., Yeh, C. C. M., Zhu, Y., Gharghabi, S., ... & Keogh, E. (2019). The UCR time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6), 1293-1305.

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 248-255). IEEE.

Duchowski, A. T., Jörg, S., Allen, T. N., Giannopoulos, I., & Krejtz, K. (2016). Eye movement synthesis. *In Proceedings of the 9th Biennial ACM Symposium on Eye Tracking Research & Applications* (pp. 147-154).

Elbattah, M. (2020). GitHub Repository: https://github.com/Mahmoud-Elbattah/NCTA2020

Elbattah, M., Carette, R., Dequen, G., Guérin, J.L., & Cilia, F. (2019). Learning Clusters in Autism Spectrum Disorder: Image-Based Clustering of Eye-Tracking Scanpaths with Deep Autoencoder. *In Proceedings of the 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, (pp. 1417-1420). IEEE.

Fuhl, W. (2020). Fully Convolutional Neural Networks for Raw Eye Tracking Data Segmentation, Generation, and Reconstruction. *arXiv preprin*t arXiv:2002.10905.

Guillon, Q., Hadjikhani, N., Baduel, S., & Rogé, B. (2014). Visual social attention in autism spectrum disorder: Insights from eye tracking studies. *Neuroscience & Biobehavioral Reviews*, 42, 279-297.

Henderson, J. M. (2003). Human gaze control during real-world scene perception. *Trends in Cognitive Sciences*, 7(11), 498-504.

Huey, E. B. (1908). *The psychology and pedagogy of reading*. The Macmillan Company: New York, NY, USA.

Jacob, R.J. (1995). Eye tracking in advanced interface design. In W. Barfield W, T.A. Furness (eds). *Virtual Environments and Advanced Interface Design.* pp. 258–288. New York: Oxford University Press.

Javal, L. (1878). Essai sur la physiologie de la lecture. *Annales d'Oculistique*. 80:240–274.

Javal, L. (1879). Essai sur la physiologie de la lecture. *Annales d'Oculistique*. 82:242–253.

Khalighy, S., Green, G., Scheepers, C., & Whittet, C. (2015). Quantifying the qualities of aesthetics in product design using eye-tracking technology. *International Journal of Industrial Ergonomics*, 49, 31-43.

Klein, A., Yumak, Z., Beij, A., & van der Stappen, A. F. (2019). Data-driven gaze animation using recurrent neural networks. In Proceedings of ACM SIGGRAPH Conference on Motion, Interaction and Games (MIG) (pp. 1-11). ACM.

Kingma, D.P., & Ba, J. (2015). Adam: a method for stochastic optimization. *In Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Khushaba, R. N., Wise, C., Kodagoda, S., Louviere, J., Kahn, B. E., & Townsend, C. (2013). Consumer neuroscience: Assessing the brain response to marketing stimuli using electroencephalogram (EEG) and eye tracking. *Expert Systems with Applications*, 40(9), 3803-3812.

Le, B.H., Ma, X., & Deng, Z. (2012). Live speech driven head-and-eye motion generators. *IEEE Transactions on Visualization and Computer Graphics*, 18(11), pp. 1902-1914.

Lee, S. P., Badlr, J. B., & Badler, N. I. (2002). Eyes alive. *In Proceedings of the 29th annual Conference on Computer Graphics and Interactive Techniques* (pp. 637-644).

Ma, X., & Deng, Z. (2009). Natural eye motion synthesis by modeling gaze-head coupling. *In Proceedings of the IEEE Virtual Reality Conference* (pp. 143-150). IEEE.

Majaranta P., Bulling A. (2014). Eye tracking and eye-based human–computer interaction. In: Fairclough S., Gilleade K. (eds). *Advances in Physiological Computing. Human–Computer Interaction Series*. Springer, London.

Mele, M. L., & Federici, S. (2012). Gaze and eye-tracking solutions for psychological research. *Cognitive Processing*, 13(1), 261-265.

Meißner, M., Pfeiffer, J., Pfeiffer, T., & Oppewal, H. (2019). Combining virtual reality and mobile eye tracking to provide a naturalistic experimental environment for shopper research. *Journal of Business Research*, 100, 445-458.

Oyekoya, O., Steptoe, W., & Steed, A. (2009). A saliency-based method of simulating visual attention in virtual scenes. *In Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology* (pp. 199-206).

Steptoe, W., Oyekoya, O., & Steed, A. (2010). Eyelid kinematics for virtual characters. Computer Animation and Virtual Worlds, 21(3-4), pp. 161-171.

Trutoiu, L. C., Carter, E. J., Matthews, I., & Hodgins, J. K. (2011). Modeling and animating eye blinks. *ACM Transactions on Applied Perception (TAP)*, 8(3), 1-17.

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.

Zhai, S. (2003). What's in the eyes for attentive input. *Communications of the ACM*, 46(3), 34-39.