
This is an electronic reprint of the original article.
This reprint may differ from the original in pagination and typographic detail.

Arapakis, Ioannis; Leiva, Luis A.

Learning Efficient Representations of Mouse Movements to Predict User Attention

Published in:

SIGIR 2020 - Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval

DOI:

[10.1145/3397271.3401031](https://doi.org/10.1145/3397271.3401031)

Published: 25/07/2020

Document Version

Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

Please cite the original version:

Arapakis, I., & Leiva, L. A. (2020). Learning Efficient Representations of Mouse Movements to Predict User Attention. In *SIGIR 2020 - Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 1309-1318). ACM. <https://doi.org/10.1145/3397271.3401031>

This material is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.

Learning Efficient Representations of Mouse Movements to Predict User Attention

Ioannis Arapakis
Telefonica Research

Spain
ioannis.arapakis@telefonica.com

Luis A. Leiva
Aalto University

Finland
firstname.lastname@aalto.fi

ABSTRACT

Tracking mouse cursor movements can be used to predict user attention on heterogeneous page layouts like SERPs. So far, previous work has relied heavily on handcrafted features, which is a time-consuming approach that often requires domain expertise. We investigate different representations of mouse cursor movements, including time series, heatmaps, and trajectory-based images, to build and contrast both recurrent and convolutional neural networks that can predict user attention to direct displays, such as SERP advertisements. Our models are trained over *raw* mouse cursor data and achieve competitive performance. We conclude that neural network models should be adopted for downstream tasks involving mouse cursor movements, since they can provide an invaluable implicit feedback signal for re-ranking and evaluation.

CCS CONCEPTS

• **Information systems** → *Search interfaces*; • **Computing methodologies** → *Modeling and simulation*.

KEYWORDS

Sponsored search; Online advertising; Mouse cursor; Direct displays; User attention; Neural networks; Transfer Learning

ACM Reference Format:

Ioannis Arapakis and Luis A. Leiva. 2020. Learning Efficient Representations of Mouse Movements to Predict User Attention. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20)*, July 25–30, 2020, Virtual Event, China. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3397271.3401031>

1 INTRODUCTION

Search engine results pages (SERPs) have become sophisticated user interfaces (UIs) that include heterogeneous modules, or *direct displays*, such as image carousels, videos, cards, and a diverse kind of advertisements. Since users are no longer faced with a text-based linear listing of search results, research demands more sophisticated ways of understanding how users interact and examine SERPs. With multiple page elements competing for the user’s attention, understanding which elements do actually attract attention is key

to search engines, and has applications for ranking, search page optimization, and UI evaluation.

Researchers have shown that mouse cursor movements can be used to infer user attention [5] and information flow patterns [55] on SERPs. While mouse tracking cannot substitute eye tracking technology, it is nevertheless much more scalable and requires no special equipment. Further, most queries do not result in a click if the user can satisfy their information needs directly on the SERP [24], therefore search engines must rely on other behavioral signals to understand the underlying search intent. So, mouse tracking data can be gathered “for free” and can provide search engines with an implicit feedback signal for re-ranking and evaluation. For example, a search engine can predict user attention to individual SERP components such as the knowledge module [6] and re-design it accordingly. Similarly, predicting attention to advertisements [7] can improve current auction schemes and make them more transparent to bidders. Those are important and particularly key use cases, considering that previous research have assumed a uniform engagement with a web page and do not distinguish well enough between attended and ignored layout components [15, 49].

Previous work has relied on handcrafted features to model user interaction data on SERPs. For example, Guo and Agichtein were able to classify different query types [26] and infer search intent in search results [27] by examining, e.g. within-distances between cursor movements, hovers, and scrolling. Similarly, Arapakis and Leiva [5] derived 638 features from mouse cursor data to predict user attention to direct displays, and Lagun et al. [45] discovered frequent subsequences, or motifs, in mouse movements that were used to improve search results relevance. While these are very valuable works and have contributed to our current understanding of search behavior analysis, finding the right feature set for the task at hand is time-consuming and requires domain expertise. To solve this, we rely on artificial neural networks (ANNs) that are trained on different representations of mouse cursor movements.

We build and contrast both recurrent and convolutional ANNs to predict user attention to SERP advertisements. We thus tackle the problem of mouse movements classification using both sequential and pixel-based representations, the latter using different visual encoding of the temporal information embedded in mouse cursor movements, to be described later. Importantly, our models are trained on *raw* mouse cursor movements, which do not depend on a particular page structure, and achieve competitive performance while predicting user attention to different ad formats. Taken together, our results suggest that ANN-based models should be adopted for downstream tasks involving mouse cursor movements, as these models remove the need for handcrafted features and can capture better non-linearities within the data.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

SIGIR '20, July 25–30, 2020, Virtual Event, China

© 2020 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-8016-4/20/07...\$15.00

<https://doi.org/10.1145/3397271.3401031>

1.1 Preliminaries

Arguably, ANNs are universal function approximators [19, 33], since a feedforward network (i.e. with no loops) having a single hidden layer with a sufficiently large number of neurons can approximate *any* continuous function of n -dimensional input variables [51]. For most Deep Learning practitioners, sequence modeling is synonymous with Recurrent Neural Networks (RNNs). RNNs are an extension of regular ANNs that have connections feeding the hidden layers of the network back into themselves, also known as recurrent connections or feedback loops. Therefore it might seem clear that we should use RNNs to process mouse tracking data, since this kind of data can be straightforwardly modeled as multivariate time series of spatial (or spatiotemporal) cursor coordinates, where each coordinate can be assumed to depend on the previous one. Yet recent research has suggested that convolutional neural networks (CNNs) may outperform RNNs on tasks dealing with sequential data, such as audio synthesis and machine translation [11]. CNNs are also an extension of regular ANNs, but using feedforward connections instead of recurrent connections and assembling complex hierarchical patterns via smaller and simpler patterns through convolutional operations.

A known issue of training Deep Learning models is that gradients may either vanish or explode while they backpropagate through the network. This problem is particularly exacerbated in RNNs, due to their long-term dependencies within the data. However, mouse cursor trajectories are sometimes very short, e.g. a few seconds worth of interaction, therefore in this paper we explore both simple RNNs and more sophisticated versions thereof: Long Short-Term Memory (LSTM) and Gate Recurrent Unit (GRU) networks. Both networks have similar performance [40] and were designed to learn long-term information. We also investigate the bidirectional LSTM network, which allows RNNs to learn from past *and* future timesteps, to better understand the sequence context.

Another important limitation while training RNNs is the fine-tuning of many model parameters (network weights), because every timestep depends on the previous one, which usually requires high computational resources. Indeed, the temporal dependencies between previous sequence elements prevents parallelizing training of RNNs [53]. Therefore, in this paper we explore alternative pixel-based representations of mouse cursor data that can be handled with CNNs. Because CNNs usually require a large amount of training data, a common technique is *transfer learning*: use a pre-trained network on a larger dataset and calibrate the model architecture to the nature and characteristics of the smaller dataset. Concretely, we used transfer learning of popular CNN architectures including AlexNet [44], SqueezeNet [37], ResNet [31], and VGGNet [61], all of them state-of-the-art CNNs and widely used in downstream tasks such as image classification or video analysis.

2 RELATED WORK

The construct of attention has become the common currency on the Web. Objective measurements of attentional processes [64] are increasingly sought after by both the media industry and scholar communities to explain or predict user behavior. Along those lines, the connection between mouse cursor movements and the underlying psychological states has been a topic of research since

the early 90s [1, 2, 14, 52]. Some studies have investigated the utility of mouse cursor data for predicting the user’s emotional state [10, 41, 42, 65, 68], but also the extent that they can help identify demographic attributes like gender [43, 57, 66] and age [43, 57]. The above works demonstrate that certain cognitive and motor control mechanisms are embodied and reflected, to some extent, in our mouse cursor movements and online interactions.

Recently, a large body of research [15, 26, 27, 29, 34, 45, 49, 54, 55, 59] established further the cognitive grounding for hand-eye relationship and has demonstrated the utility of mouse cursor analysis as a low-cost and scalable proxy of visual attention, especially on SERPs. In line with this evidence, several works have investigated closely the user interactions that stem from mouse cursor data for various use cases, such as web search [5, 15, 26, 27, 29, 45, 49] or web page usability evaluation [8, 9, 46]. In what follows, we review previous research efforts that have focused on mouse cursor analysis to predict user interest and attention.

2.1 User Interest in Web Search Tasks

User models of scanning behaviour in SERPs have been assumed to be linear, as users tend to explore the list of search results from top to bottom. However, today’s SERPs include several direct displays such as image and video search results, featured snippets, or advertisement. To account for this SERP heterogeneity, Diaz et al. [20] incorporated ancillary page modules to the classic linear scanning model, which proved useful to help improving SERP design by anticipating searchers’ engagement patterns for a given SERP arrangement. However, this model was not designed to measure effectively user attention to specific direct displays and does not exploit the latent information encoded in mouse cursor movements.

Another line of research considered simple, coarse-grained features derived from mouse cursor data to be surrogate measurements of user interest, such as the amount of mouse cursor movements [59] or mouse cursor’s “travel time” [18]. Follow up work adopted fine-grained mouse cursor features, which have been shown to be more effective. For example, Guo et al. [26, 27] computed within-distances between mouse cursor distances to disambiguate among informational and navigational queries, and could identify a user’s research or purchase intent based on aggregated behavioral signals that include, among others, mouse hovering and scrolling activity. Approaches like these have been directed at predicting general-purpose web-based tasks like search success [29] and satisfaction [49], user’s frustration [23], relevance judgements of search results [35, 63], and query abandonment [21, 36]. Eventually, they lack the granularity in predicting attention with particular direct displays of a SERP, such as advertisements, that our proposed modelling approach achieves.

2.2 User Attention in Web Search Tasks

Most research studies assume that eye fixation means examination [13]. However, Liu et al. [50] reports that about half of the search results fixated by users are not actually read, since there is often a preceding skimming step in which the user quickly scans the search results. Based on this observation, they propose a two-stage examination model: a first “from skimming to reading” stage and a second “from reading to clicking” stage. Interestingly, they

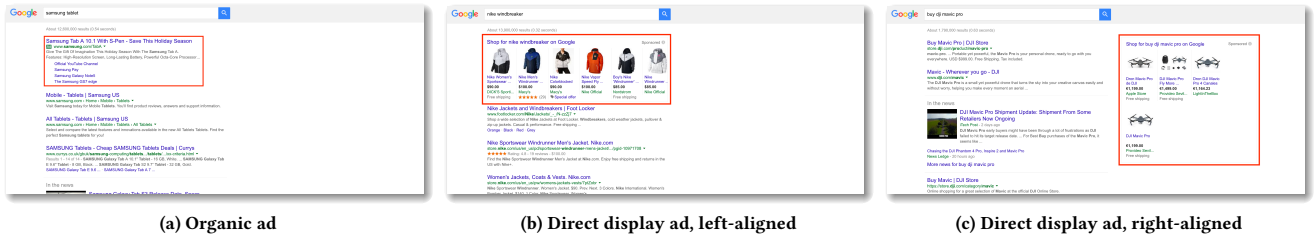


Figure 1: Examples of the ad formats, highlighted in red, and their positions on the Google SERP: Organic ad (a) vs. left-aligned (b) and right-aligned (c) direct display ads. In our experiments, only one ad format was visible at a time.

showed that both stages can be predicted with mouse movement behaviour, which can be collected at large scale.

Cursor movements can therefore be used to estimate user attention on SERP components, including traditional snippets, aggregated results, maps, and advertisements, among others. However, works that employ mouse cursor information to predict user attention with specific elements within a web page have been scarce. Despite these challenges, some of the early work by Arapakis et al. [3, 4] investigated the utility of mouse movement patterns to measure within-content engagement on news pages and predict reading experiences. Lagun et al. [45] introduced the concept of *motifs* for estimating results relevance. Similarly, Liu et al. [49] applied the motifs concept to SERPs to predict search result utility, searcher effort, and satisfaction at a search task level. Finally, Arapakis and Leiva [5] which investigated user engagement with direct displays on SERPs, concretely with the Knowledge Graph [6].

Our work differs significantly from previous art in several ways. First, we implement a predictive modelling framework to measure user attention to SERP advertisements, which are probably the most relevant instance of direct displays for search engines, from a business perspective. Second, previous work has used Machine Learning models which rely on ad-hoc and domain-specific features. As previously discussed, feature engineering requires domain expertise to come up with the best discriminative features. In contrast, we investigate several ANN architectures that use *raw* mouse cursor data, represented either as time series or as visual representations, and can predict user attention with competitive performance. Finally, we examine the performance of our predictive models w.r.t. sponsored ads served under different formats and different positions within a SERP and, thus, significantly expand on previous research and findings in the community.

3 USER STUDY

Online advertising comprises ads that are served under different formats (e.g. text, image, or video, or rich media), each with its unique look and feel. Some formats appear to be more effective than traditional online ads in terms of user attention and purchase intent [60], but also may cause “ad blindness” to a greater or a lesser extent [56]. Therefore, to understand how web search users engage with ads that appear under different formats and positions in SERPs, we conducted a user study through the FIGURE EIGHT¹ crowdsourcing platform. We collected feedback from participants who performed

brief transactional search tasks using Google Search and aimed to predict when users notice the ads that appear on SERPs, under different conditions. To mitigate low-quality responses, several preventive measures were put into practice, such as introducing gold-standard questions, selecting experienced contributors (Level 3) with high accuracy rates, and monitoring task completion time, thus ensuring the internal validity of our experiment.

3.1 Experiment Design

We used a between-subjects design with two independent variables: (1) ad format, with two levels (organic and direct display ads) and (2) ad position, with two levels (top-left and top-right position). Notice that organic ads are only shown in the left part of Google SERPs; see Figure 1. The dependent variable was ad attention.

Our experiment consisted of a brief transactional search task where participants were presented with a predefined search query and the corresponding SERP, and were asked to click on any element of the page that answered it best. All search queries (Section 3.2) triggered both organic (Figure 1a) and direct display ads (Figures 1b and 1c) on Google SERPs. Each participant was randomly assigned a search query and could perform the task only once, since inquiring at post-task about the presence of an ad would make them aware of it and could introduce carry over effects. In summary, each participant was only exposed to a unique combination of query, ad format, and ad position.

3.2 Search Query Sample

Starting from GOOGLE TRENDS,² we selected a subset of the Top Categories and Shopping Categories that were suitable candidates for the transactional character of our search tasks. From this subset of categories, we extracted the top search queries issued in the US during the last 12 months. Next, from the resulting collection of 375 search queries, we retained 150 for which the SERPs were showing at least one direct display ad (50 search queries for each combination of direct display ad format and position). Using this final selection of search queries, we produced the static version of the corresponding Google SERPs and injected the JavaScript code (Section 3.4) that allowed us to control the ads format and capture all client-side user interactions.

¹<https://www.figure-eight.com>

²<https://trends.google.com/trends/>

3.3 SERP Layout

All SERPs were all in English and were scraped for later instrumentation (Section 3.4). Participants accessed the instrumented SERPs through a dedicated server, which did not alter the look and feel of the original Google SERPs. This allowed us to capture fine-grained user interactions while ensuring that the content of the SERPs remained consistent and that each experimental condition was properly administered. All SERPs had both organic and direct display ads. Organic ads appeared both at the top-left and bottom-left position of the SERP, whereas direct display ads could appear either at the top-right or top-left position (but not both at the same time on the same SERP). Therefore, we ensured that only one ad was visible per condition and participant, since we are focusing on the single-slot auction case. This was possible by instrumenting each downloaded SERP with custom JavaScript code that removed all ads except the one that was tested in each of the experimental conditions (Figure 1). For example, bottom-most organic ads were not shown, since (i) users have to scroll all way down to the bottom of the SERP to reveal them and (ii) these ads have the same look and feel than the organic ads shown on the top-most position.

3.4 Mouse Cursor Tracking

We inserted JavaScript code that captured mouse cursor movements and associated metadata while users browsed the SERPs. We used EvTRACK,³ a general-purpose open-source JavaScript event tracking library that allows event capturing either via event listeners (the event is captured as soon as it is fired) or via event polling (the event is captured at fixed-time intervals). We captured mousemove events via event polling, every 150 ms to avoid unnecessary data overhead [47], and all the other browser events (e.g., load, click, scroll) via event listeners. Whenever an event was recorded, we logged the following information: mouse cursor position (x and y coordinates), timestamp, event name, and the XPath of the DOM element that relates to the event.

3.5 Self-Reported Ground-truth Labels

In a similar vein to previous work [5, 23, 45, 49], we collected ground-truth labels through an online questionnaire, which was administered at post-task and asked the user *to what extent* they paid attention to the ad using a 5-point Likert-type scale: “Not at all” (1), “Not much” (2), “I can’t decide” (3), “Somewhat” (4), and “Very much” (5). These scores would be collapsed to binary labels, but we felt it was necessary to begin with a 5-point Likert scale for several reasons. Unlike other scales that offer limited options (e.g. 2 or 3-point scales) and can result in highly skewed or neutral data [39], or scales with too many options (7-point scales) that are harder to understand, a 5-point Likert-type scale leaves room for “soft responses” while remaining fairly intuitive. Neutral scores were not considered for analysis.

3.6 Participants

We recruited 3,206 participants, of age 18 – 66 and of mixed nationality, through the FIGURE EIGHT platform. All participants were proficient in English and were experienced (Level 3) contributors,

i.e. they had a track record of successfully completed tasks and of a different variety, thus being considered very reliable contributors.

3.7 Procedure

Participants were informed that they should perform the search task from a desktop or laptop computer as long as they used a computer mouse. They were told to deactivate any ad-blocker before proceeding with the task, otherwise our JavaScript code would prevent them from taking part in the study. Participants were asked to act naturally and click on anything that would best answer the search query, e.g. result links, images, etc. An example of the search task descriptions provided to the participants is the following: “*You want to buy a Rolex watch and you have submitted the search query ‘rolex watches’ to Google Search. Please browse the search results page and click on the element that you would normally select under this scenario.*” The search task had to be completed in a single session and each query was performed on average by five different participants. The SERPs were randomly assigned to the participants and each participant could take the study only once. Upon concluding the search task, participants were asked to complete the post-task questionnaire which inquired about the presence of the ad (at that point the participants did not have access to the webpage). The payment was \$0.20 and participants could also opt out at any moment, in which case they would not be compensated.

3.8 Dataset

After excluding those logs with incomplete mouse cursor data (less than five mouse coordinates \approx one second of user interaction data), we concluded to 2,289 search sessions that hold 45,082 mouse cursor positions⁴. Of these search sessions, 763 correspond to the organic ad condition, 793 correspond to the left-aligned direct display ad, and 733 correspond to the right-aligned direct display ad. Ground-truth labels were converted to a binary scale using the following mapping: “Not at all” and “Not much” were assigned to the negative class, and “Somewhat” and “Very much” were assigned to the positive class, while neutral scores were not considered in subsequent analysis. We note that the class distribution was fairly balanced (66% of positive cases) across the experimental conditions. We then used 60-10-30 (%) disjoint stratified splits to assign the observations to the training, validation, and test set, respectively. The stratified sampling process was performed once per ad format and preserved the original class distribution in each data partition.

4 DATA REPRESENTATIONS

We framed the problem of ad attention prediction as a binary classification task: *given a user’s mouse cursor trajectory, did the user notice the ad?* To this end, as introduced in Section 1.1, we implemented both recurrent and convolutional neural networks to handle different representations of mouse cursor movements on SERPs. In what follows, we describe these data representations.

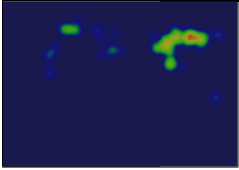
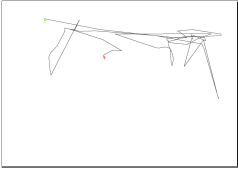

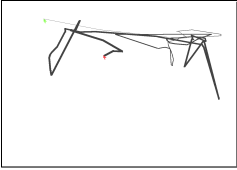
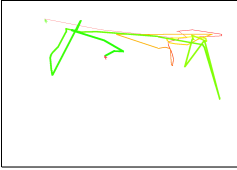
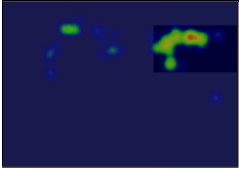
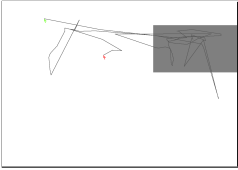
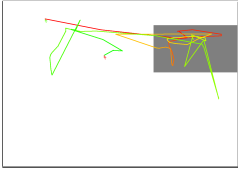
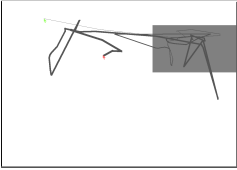
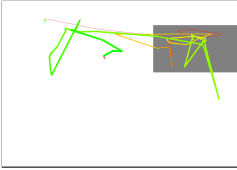
4.1 Time Series Representation

In our experiments, a mouse cursor trajectory is modeled as a multivariate time series of 2D cursor coordinates. The data is ordered

³<https://github.com/luileito/evtrack>

⁴The dataset is freely available here: <https://gitlab.com/iarapakis/the-attentive-cursor-dataset.git>

Table 1: Types of visual representations used to train the CNN models. The top row shows representations without the ad placeholder, whereas the bottom row shows representations with the ad placeholder.

	Heatmap	Trajectories	Colored trajectories	Trajectories with line thickness	Colored trajectories with line thickness
w/o ad placeholder					
	(1)	(2)	(3)	(4)	(5)
w/ ad placeholder					

by the time they were collected and there are no consecutively duplicated coordinates. A particular characteristic of this data representation is that events are *asynchronous*, i.e. contrary to regular time series, the sampling rate of mousemove events is not constant. This is so because web browser events are first placed in an event queue and then fired as soon as possible [58].

An inherent limitation while training RNN models is that the “memory” of the network must be fixed, i.e. the maximum number of timesteps that the model can handle must be set to a fixed length. This will impact model training in two ways. First, if we choose a small memory footprint (short sequence length) for our models, we would be truncating longer sequences that otherwise could bear rich behavioral information about the user. Second, if we choose a large memory footprint, then we would be wasting computational resources, as the model would require more weights to optimize and, in consequence, training time would be unnecessarily longer. Therefore, to make the training of our RNNs tractable, we inspected our data and decided to set the maximum sequence length to 50 timesteps, which roughly corresponds to the mean sequence length observed in our dataset plus one standard deviation. Since mouse cursor trajectories are variable-length sequences, shorter sequences were padded to such a fixed length of 50 timesteps and longer sequences were truncated. Finally, since each mouse cursor trajectory was performed on different web browsers with different screen sizes, the horizontal coordinates were normalised by each user’s viewport width. The vertical coordinates do not need to be normalised, since the SERP layout has a fixed width.

4.2 Visual Representation

According to our data, 90% of all mouse coordinates happened above the page fold, i.e. within the browser’s visual viewport. This was somewhat expected given the nature of our task: in crowdsourcing studies, users often proceed as quickly as possible in order to maximize their profit [22, 38]. However, this suggests that we can expect a visual representation to perform well, since most of the user interactions would be adequately represented in a fixed-size image. Therefore, we created five visual encodings (Table 1):

- (1) *Heatmap*. The influence of each mouse cursor coordinate is determined with a 2D Gaussian kernel of 25 px radius. When various kernels overlap, their values are added together.
- (2) *Trajectories*. Every two consecutive coordinates are joined with a straight line. The first and last coordinates are rendered as cursor-like images, in green and red color, respectively.
- (3) *Colored trajectories*. The trajectory line color is mapped to a temperature gradient, where green areas denote the beginning of the trajectory and red areas denote the end of the trajectory.
- (4) *Trajectories with variable line thickness*. The trajectory line thickness is proportional to the percentage of time, so that thick areas denote the beginning of the mouse trajectory and thin areas denote the end of the trajectory.
- (5) *Colored trajectories with variable line thickness*. A combination of the two previous representations described.

The mouse cursor data were rendered according to each visual encoding using the Simple Mouse Tracking system [48], which was operated via PhantomJS,⁵ a scriptable headless browser. Each mouse cursor trajectory was normalized according to the user’s viewport. Finally, no data augmentation or transformation techniques were applied, as often performed in computer vision tasks, and the images were saved as 1280x900 px PNG files.

5 DEEP LEARNING MODELS

5.1 Recurrent Neural Networks

In what follows, we provide an overview of the RNN units, or cells, that we investigated for our sequence classification task. These units cover the most popular choices by Deep Learning practitioners.

- (1) *SimpleRNN*. Vanilla recurrent cell, i.e. a fully-connected unit where its output is fed back to its input at every timestep.
- (2) *LSTM*. This unit introduces an output gate and a forget gate [32], to remember long-term dependencies within the data.
- (3) *GRU*. A simplification of the LSTM cell [16], where there is no output gate. GRU can outperform LSTM units in terms of convergence and generalization [17].

⁵<https://phantomjs.org/>

- (4) *BLSTM*. This unit uses both past and future contexts, by concatenating the outputs of two RNNs [25]: one processing the sequence from left to right (forward RNN), the other one from right to left (backward RNN). We note that any RNN unit can become bidirectional, however we used the LSTM variant because of their popularity and to keep our experiments consistent.

All RNN models have an input layer with 50 neurons (one neuron per timestep), followed by a hidden layer with $n \in [16, 24, \dots, 128]$ neurons and ReLU activation, a dropout layer with drop rate $q \in [0.5, 0.4, \dots, 0.1]$ to prevent overfitting, and a fully-connected layer of 1 output neuron using sigmoid activation. Each RNN model outputs a probability prediction p of the user’s attention to an ad, where $p > .5$ indicates that the user has noticed the ad.

We trained the models using binary crossentropy as loss function, the popular Adam optimizer (stochastic gradient descent with momentum) with learning rate $\eta \in [10^{-3}, 10^{-4}, \dots, 10^{-7}]$, and decay rates $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We set a maximum number of 90 epochs, using an early stopping of 20 epochs that monitors the validation loss, and tried different batch sizes $b \in [16, 32, 64]$.

As noted, we explored different combinations of hyperparameters (n, q, η, b). Our design space is thus quite large, rendering the classic grid-search approach unfeasible. Therefore, we optimized our RNN models via random search [12], which has a high probability of finding the most suitable configuration without having to explore all possible combinations [67]. The best configuration is determined via 3-fold cross-validation, i.e. a given hyperparameter combination is tested up to three times over the validation data partition and the final result is averaged. The best combination is the one with the minimum validation loss.

5.2 Convolutional Neural Networks

As anticipated in Section 1.1, we investigated four popular CNN architectures: AlexNet [44], SqueezeNet [37], ResNet50 [31], and VGG19 [61], all of which were trained on the ImageNet database (1M images with 1000 categories). Our choice of CNNs favoured diversity and was guided by the fact that the above architectures have been successfully tested in a wide range of computer vision applications, and have different designs, modules, and number of parameters. For example, ResNet50 and VGG19 employ a large number of layers, hence resulting in *deep* networks, (roughly twice as deep as AlexNet), use skip connections, and are among the early adopters of batch normalisation. On the other hand, AlexNet and its simplified variant SqueezeNet were the first to introduce ReLU activations and implement a shallow architecture while attaining high accuracy and requiring less bandwidth to operate.

Using the representations discussed in Section 4.2, we applied transfer learning to calibrate the CNNs to the particularities our images, which are quite different from the natural scenery images found in ImageNet. This way, we can reuse an existing architecture and apply it to our own prediction task because there are universal, low-level features shared among images. For each CNN model, we applied the following steps. First, we initialised the last layer of the CNN model with randomly assigned weights, while retaining the weights of the initial layers that were pre-trained on ImageNet. Next, we run a learning rate finder [62] that let η values to cyclically

vary by linearly increasing it for a few epochs. Training with cyclical learning rates instead of fixed values improves classification accuracy without the need to manual fine-tuning and often results in fewer iterations. We then unfreezed and re-trained all the layers of the CNN model for 300 epochs, using a per-cycle maximal LR. In addition, we used the early stopping method that terminated training when the monitored AUC stopped improving after 30 epochs. Batch size was adjusted accordingly to each architecture, to optimise for the use of GPU memory. Finally, we trained our CNN models with the Adam optimizer, using the same decay rates as in the RNN models and binary cross-entropy as loss function.

6 RESULTS

We report the performance of our ANNs trained on the different data representations, and for the different ad formats. We use the standard IR metrics of Precision, Recall, and F-Measure (F1 score), weighted according to the target class distributions in each case. The F-Measure provides an aggregated insight about the functionality of a classifier, however remaining sensitive to data distributions. Therefore, we also report the Area Under the ROC curve (AUC), which is insensitive to class distribution and error costs [30], and use it as our key metric to determine the top performing classifier for each setup. To investigate further the performance differences across models and conditions, we run Friedman’s ANOVA as an omnibus test and, if the result is statistically significant, we use the Wilcoxon signed-rank test for pairwise comparisons, with correction for multiple testing. Tables 2 to 4 show the results of our experiments, including the hyperparameter configuration used for each model. The Epoch column indicates the maximum number of epochs used for training each model, as we used early stopping to prevent overfitting. Gray table cells indicate the top performer in each data representation group, whereas the overall best performance result (across all representations) is denoted in bold typeface.

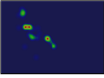
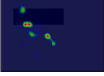

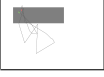



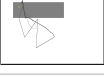


6.1 Effect of Model Type

We note that, under our experimental settings, CNN models outperform RNN models across all ad format conditions, sometimes by a large margin. When considering the best overall performing models for each type of ANN architecture, we can observe noticeable improvements in terms of the F1 and AUC metrics. More specifically, the best CNN model represents an increment over the best RNN model by 3.24% in terms of F1 and by 9.35% in AUC for the organic ads (Table 2). Similarly, we observe an increment of 13.91% (F1) and 26.42% (AUC) for the left-aligned direct display advertisements (Table 3). Lastly, we note an increment of 18.65% (F1) and 20.35% (AUC) for the right-aligned direct display advertisements (Table 4).

6.2 Effect of Ad Placeholder

We run statistical analysis to determine whether the presence of the ad placeholder had any effect on the models’ performance. For the organic ads and the right-aligned direct display advertisements, the Wilcoxon signed-rank test showed that the presence of the ad placeholder in the representations did not elicit a significant change in the AUC or F1 scores. However, in the left-aligned direct display condition, F1 scores were significantly higher for the models trained on the representations without the ad placeholder (Mdn=0.718),

Table 2: Experiment results for organic advertisements. Gray cells indicate the top performer in each representation group. The overall best performance result (across all groups) is denoted in bold typeface. The positive:negative ratio is 447:222.

Representation	Example	Architecture	Hyperparameters	Epoch	Adj. Precision	Adj. Recall	Adj. F-measure	AUC
Time series	$(x_1, y_1), \dots, (x_N, y_N)$	SimpleRNN	$\eta = 10^{-3}, q = 0.4, n = 64, b = 64$	90	0.556	0.697	0.603	0.529
		LSTM	$\eta = 10^{-3}, q = 0.3, n = 64, b = 32$	65	0.622	0.604	0.612	0.531
		BLSTM	$\eta = 10^{-4}, q = 0.3, n = 32, b = 16$	20	0.695	0.637	0.654	0.631
		GRU	$\eta = 10^{-3}, q = 0.2, n = 32, b = 32$	90	0.672	0.711	0.678	0.557
Heatmap		AlexNet	$\eta = 2.51E-7, b = 64$	38	0.572	0.667	0.583	0.547
		SqueezeNet	$\eta = 6.91E-7, b = 64$	41	0.602	0.524	0.540	0.543
		ResNet50	$\eta = 9.12E-7, b = 32$	46	0.652	0.679	0.659	0.638
		VGG19	$\eta = 1.90E-6, b = 16$	54	0.627	0.636	0.628	0.614
Heatmap with ad placeholder		AlexNet	$\eta = 2.51E-7, b = 64$	59	0.639	0.677	0.650	0.656
		SqueezeNet	$\eta = 2.51E-7, b = 64$	34	0.617	0.657	0.621	0.587
		ResNet50	$\eta = 8.06E-7, b = 32$	34	0.599	0.518	0.537	0.532
		VGG19	$\eta = 3.41E-7, b = 16$	44	0.601	0.622	0.606	0.598
Trajectories		AlexNet	$\eta = 4.78E-7, b = 64$	38	0.633	0.667	0.634	0.614
		SqueezeNet	$\eta = 3.31E-7, b = 64$	32	0.654	0.654	0.654	0.589
		ResNet50	$\eta = 3.41E-7, b = 32$	81	0.637	0.627	0.630	0.626
		VGG19	$\eta = 6.31E-7, b = 16$	33	0.658	0.693	0.645	0.600
Trajectories with ad placeholder		AlexNet	$\eta = 8.31E-6, b = 64$	62	0.620	0.639	0.626	0.610
		SqueezeNet	$\eta = 5.75E-7, b = 64$	59	0.614	0.619	0.615	0.578
		ResNet50	$\eta = 1.16E-6, b = 32$	37	0.632	0.464	0.449	0.690
		VGG19	$\eta = 1.16E-6, b = 16$	44	0.621	0.610	0.610	0.613
Colored trajectories		AlexNet	$\eta = 3.63E-7, b = 64$	57	0.594	0.633	0.607	0.561
		SqueezeNet	$\eta = 2.51E-7, b = 64$	36	0.630	0.658	0.639	0.605
		ResNet50	$\eta = 5.58E-7, b = 32$	50	0.619	0.534	0.550	0.601
		VGG19	$\eta = 3.41E-7, b = 16$	31	0.610	0.647	0.617	0.629
Colored trajectories with ad placeholder		AlexNet	$\eta = 3.63E-7, b = 64$	57	0.640	0.637	0.637	0.610
		SqueezeNet	$\eta = 3.98E-7, b = 64$	34	0.612	0.527	0.543	0.540
		ResNet50	$\eta = 3.41E-7, b = 32$	87	0.694	0.505	0.506	0.654
		VGG19	$\eta = 3.86E-7, b = 16$	50	0.563	0.623	0.580	0.582
Trajectories with line thickness		AlexNet	$\eta = 5.24E-7, b = 64$	31	0.599	0.484	0.495	0.539
		SqueezeNet	$\eta = 5.75E-5, b = 64$	35	0.559	0.555	0.562	0.546
		ResNet50	$\eta = 3.86E-7, b = 32$	49	0.657	0.660	0.654	0.612
		VGG19	$\eta = 4.36E-7, b = 16$	104	0.627	0.660	0.637	0.617
Trajectories with line thickness and ad placeholder		AlexNet	$\eta = 2.75E-5, b = 64$	43	0.630	0.642	0.631	0.616
		SqueezeNet	$\eta = 3.02E-7, b = 64$	36	0.674	0.665	0.700	0.657
		ResNet50	$\eta = 8.06E-7, b = 32$	31	0.669	0.539	0.548	0.638
		VGG19	$\eta = 4.93E-7, b = 16$	67	0.575	0.628	0.595	0.622
Colored trajectories with line thickness		AlexNet	$\eta = 2.51E-6, b = 64$	71	0.653	0.674	0.655	0.615
		SqueezeNet	$\eta = 5.24E-7, b = 64$	55	0.676	0.688	0.680	0.690
		ResNet50	$\eta = 4.93E-7, b = 32$	56	0.616	0.493	0.508	0.606
		VGG19	$\eta = 8.06E-7, b = 16$	39	0.591	0.618	0.603	0.557
Colored trajectories with line thickness and ad placeholder		AlexNet	$\eta = 2.51E-7, b = 64$	55	0.665	0.692	0.666	0.657
		SqueezeNet	$\eta = 2.51E-7, b = 64$	61	0.614	0.626	0.618	0.576
		ResNet50	$\eta = 4.36E-7, b = 32$	37	0.673	0.557	0.572	0.659
		VGG19	$\eta = 1.16E-6, b = 16$	62	0.549	0.591	0.565	0.572

as opposed to those trained on the representations with the ad placeholder (Mdn=0.691): $W = 50, p = 0.041, r = -0.27$.



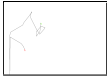
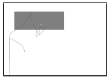






6.3 Effect of Ad Format

In organic ads, we note that the top performers are SqueezeNet (AUC=0.690), trained on the colored trajectories with varied line thickness, and ResNet50 (AUC=0.690), trained on the trajectories with ad placeholder. Considering the remaining performance metrics, SqueezeNet, despite its shallower architecture (3 hidden layers), seems to generalise better. In left-aligned direct display ads, the top performer is AlexNet (AUC=0.708), trained on the trajectories with ad placeholder, followed closely by VGG19 (AUC=0.694), trained on the heatmap with ad placeholder representation. The Wilcoxon signed-rank test showed that the presence of the ad placeholder in the representations did not elicit a significant change in the AUC or F-Measure scores, for either group. Also, in right-aligned direct display ads, where the advertisement is clearly separated from the

SERP results, the top performing model is the ResNet50, trained on trajectories without ad placeholder. This model holds the best AUC (0.739) and F-Measure (0.731) scores overall.

The Wilcoxon signed-rank test on all pairs of ad formats revealed a significant difference in terms of AUC between the organic ad (Mdn=0.610) and the left-aligned direct display (Mdn=0.634): $W = 185.5, p < .01, r = -0.62$. Similarly, we found a significant difference between the left-aligned (Mdn=0.634) and right-aligned direct displays (Mdn=0.594): $W = 716, p < .0001, r = -0.88$. When examining the F-Measure, the Wilcoxon signed-rank test showed a significant difference between organic ads (Mdn = 0.616) and the left-aligned direct display (Mdn=0.708): $W = 17, p < .0001, r = -1.15$. Furthermore, we observed a significant difference between the left-aligned (Mdn=0.708) and right-aligned direct displays (Mdn=0.629): $W = 788, p < .0001, r = -1.10$. The observed effect sizes are rather large, thus suggesting a practical importance of the results.

Table 3: Results for left-aligned direct display ads. Gray cells indicate the top performer in each representation group. The overall best performance result (across all groups) is denoted in bold typeface. The positive:negative ratio is 523:192.

Representation	Example	Architecture	Hyperparameters	Epoch	Adj. Precision	Adj. Recall	Adj. F-measure	AUC
Time series	$(x_1, y_1), \dots, (x_N, y_N)$	SimpleRNN	$\eta = 10^{-3}, q = 0.3, n = 64, b = 64$	73	0.575	0.758	0.654	0.508
		LSTM	$\eta = 10^{-3}, q = 0.4, n = 64, b = 64$	74	0.607	0.656	0.628	0.542
		BLSTM	$\eta = 10^{-3}, q = 0.5, n = 64, b = 16$	58	0.658	0.647	0.652	0.548
		GRU	$\eta = 10^{-3}, q = 0.2, n = 64, b = 32$	90	0.598	0.727	0.646	0.560
Heatmap		AlexNet	$\eta = 9.12\text{E-}7, b = 64$	84	0.661	0.714	0.683	0.606
		SqueezeNet	$\eta = 3.31\text{E-}5, b = 64$	31	0.732	0.692	0.706	0.613
		ResNet50	$\eta = 1.44\text{E-}6, b = 32$	46	0.697	0.743	0.715	0.668
		VGG19	$\eta = 7.58\text{E-}7, b = 16$	60	0.653	0.712	0.682	0.593
Heatmap with ad placeholder		AlexNet	$\eta = 2.51\text{E-}7, b = 64$	68	0.703	0.787	0.719	0.602
		SqueezeNet	$\eta = 2.51\text{E-}7, b = 64$	31	0.672	0.700	0.681	0.594
		ResNet50	$\eta = 3.63\text{E-}7, b = 32$	31	0.732	0.607	0.643	0.628
		VGG19	$\eta = 2.75\text{E-}7, b = 16$	66	0.749	0.712	0.728	0.694
Trajectories		AlexNet	$\eta = 3.02\text{E-}7, b = 64$	103	0.723	0.738	0.727	0.596
		SqueezeNet	$\eta = 5.75\text{E-}7, b = 64$	31	0.684	0.690	0.684	0.628
		ResNet50	$\eta = 7.35\text{E-}6, b = 32$	76	0.695	0.736	0.713	0.662
		VGG19	$\eta = 5.24\text{E-}7, b = 16$	36	0.692	0.732	0.709	0.687
Trajectories with ad placeholder		AlexNet	$\eta = 3.02\text{E-}7, b = 64$	103	0.745	0.745	0.745	0.708
		SqueezeNet	$\eta = 2.75\text{E-}6, b = 64$	31	0.712	0.690	0.695	0.632
		ResNet50	$\eta = 3.02\text{E-}7, b = 32$	39	0.717	0.755	0.729	0.629
		VGG19	$\eta = 7.58\text{E-}7, b = 16$	42	0.729	0.725	0.723	0.656
Colored trajectories		AlexNet	$\eta = 5.24\text{E-}4, b = 64$	68	0.745	0.745	0.745	0.677
		SqueezeNet	$\eta = 5.24\text{E-}7, b = 64$	31	0.715	0.528	0.568	0.597
		ResNet50	$\eta = 8.06\text{E-}7, b = 32$	139	0.728	0.665	0.688	0.665
		VGG19	$\eta = 3.63\text{E-}7, b = 16$	84	0.706	0.659	0.675	0.603
Colored trajectories with ad placeholder		AlexNet	$\eta = 2.51\text{E-}4, b = 64$	56	0.674	0.708	0.689	0.666
		SqueezeNet	$\eta = 2.51\text{E-}7, b = 64$	31	0.712	0.707	0.707	0.651
		ResNet50	$\eta = 2.51\text{E-}7, b = 32$	38	0.747	0.738	0.742	0.675
		VGG19	$\eta = 2.75\text{E-}7, b = 16$	72	0.703	0.766	0.714	0.670
Trajectories with line thickness		AlexNet	$\eta = 1.31\text{E-}6, b = 64$	62	0.690	0.720	0.703	0.606
		SqueezeNet	$\eta = 3.63\text{E-}7, b = 64$	31	0.725	0.738	0.737	0.637
		ResNet50	$\eta = 8.06\text{E-}7, b = 32$	39	0.692	0.732	0.709	0.604
		VGG19	$\eta = 2.51\text{E-}7, b = 16$	95	0.710	0.682	0.695	0.572
Trajectories with line thickness and ad placeholder		AlexNet	$\eta = 3.31\text{E-}5, b = 64$	31	0.705	0.732	0.717	0.664
		SqueezeNet	$\eta = 2.51\text{E-}5, b = 64$	90	0.715	0.752	0.725	0.654
		ResNet50	$\eta = 9.12\text{E-}7, b = 32$	72	0.709	0.740	0.720	0.653
		VGG19	$\eta = 1.31\text{E-}6, b = 16$	38	0.733	0.730	0.735	0.616
Colored trajectories with line thickness		AlexNet	$\eta = 1.20\text{E-}6, b = 64$	31	0.682	0.659	0.670	0.604
		SqueezeNet	$\eta = 9.12\text{E-}7, b = 64$	47	0.682	0.682	0.682	0.593
		ResNet50	$\eta = 3.41\text{E-}7, b = 32$	116	0.667	0.593	0.623	0.588
		VGG19	$\eta = 2.51\text{E-}7, b = 16$	50	0.760	0.615	0.652	0.668
Colored trajectories with line thickness and ad placeholder		AlexNet	$\eta = 1.73\text{E-}4, b = 64$	38	0.672	0.704	0.687	0.586
		SqueezeNet	$\eta = 1.00\text{E-}6, b = 64$	64	0.725	0.706	0.715	0.669
		ResNet50	$\eta = 4.36\text{E-}7, b = 32$	54	0.712	0.591	0.629	0.638
		VGG19	$\eta = 2.51\text{E-}7, b = 16$	46	0.717	0.726	0.722	0.652

7 DISCUSSION AND FUTURE WORK

This work has served as a first exploration on the feasibility of ANNs to predict user attention to ads on SERPs. We have shown that, using relatively few training data, it is possible to train RNN models from scratch and fine-tune existing CNNs via transfer learning. Our findings indicate that the mouse cursor representations and the tested model architectures achieve competitive performance in detecting user attention for all ad formats. Note that none of our models use handcrafted features, which require domain expertise, nor page-level information, since they are trained on raw sequences of mouse cursor movements. Taken together, our experiments raise the bar in the IR community and can inform researchers and practitioners when it comes to choosing one model or network configuration over another.

Having explored multiple representations of the same mouse cursor data, we have obtained several new insights and perspectives. For example, a times series representation of mouse movements



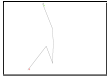
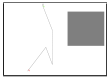
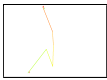


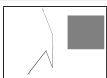


is the obvious choice, if we already know that user interactions consist of a small number of mouse movements. On the contrary, if we foresee that users are going to dwell for a relatively long time on a page, e.g. due to query difficulty or the nature of the search task, then an image-based representation would be a more apt choice.

Interestingly, our CNN models outperformed RNN models in most cases. However, we note that this might be due to the fact that our RNN models had a limited sequence length, in order to make training tractable on a single GPU,⁶ thereby limiting the learning capacity of these models. On the contrary, the CNN models had almost full coverage of the mouse cursor movements, since most user interactions happened above the fold, and they were rendered as a static image, which can be easily trained on commodity hardware.

Regarding the CNN models, our experimental results indicate that, in most cases, the presence of the ad placeholder in the visual

⁶Even if a computing cluster were used, a single GPU is still required to do a single forward/backward pass during training.

Table 4: Results for right-aligned direct display ads. Gray cells indicate the top performer in each representation group. The overall best performance result (across all groups) is denoted in bold typeface. The positive:negative ratio is 462:178.

Representation	Example	Architecture	Hyperparameters	Epoch	Adj. Precision	Adj. Recall	Adj. F-measure	AUC
Time series	$(x_1, y_1), \dots, (x_N, y_N)$	SimpleRNN	$\eta = 10^{-3}, q = 0.3, n = 32, b = 64$	56	0.577	0.677	0.572	0.530
		LSTM	$\eta = 10^{-3}, q = 0.4, n = 32, b = 32$	27	0.560	0.643	0.566	0.511
		BLSTM	$\eta = 10^{-3}, q = 0.5, n = 48, b = 16$	82	0.608	0.658	0.615	0.614
		GRU	$\eta = 10^{-3}, q = 0.4, n = 32, b = 64$	48	0.550	0.678	0.564	0.561
Heatmap		AlexNet	$\eta = 4.78\text{E-}7, b = 64$	60	0.743	0.721	0.630	0.566
		SqueezeNet	$\eta = 3.63\text{E-}7, b = 64$	59	0.647	0.708	0.636	0.599
		ResNet50	$\eta = 8.07\text{E-}7, b = 32$	39	0.668	0.698	0.668	0.599
		VGG19	$\eta = 3.02\text{E-}7, b = 16$	31	0.611	0.394	0.369	0.525
Heatmap with ad placeholder		AlexNet	$\eta = 4.78\text{E-}7, b = 64$	60	0.587	0.697	0.598	0.566
		SqueezeNet	$\eta = 6.91\text{E-}7, b = 64$	70	0.568	0.652	0.593	0.609
		ResNet50	$\eta = 1.44\text{E-}6, b = 32$	57	0.642	0.633	0.638	0.607
		VGG19	$\eta = 2.75\text{E-}7, b = 16$	32	0.679	0.685	0.681	0.680
Trajectories		AlexNet	$\eta = 5.75\text{E-}7, b = 64$	79	0.623	0.687	0.626	0.578
		SqueezeNet	$\eta = 2.75\text{E-}7, b = 64$	56	0.606	0.632	0.613	0.584
		ResNet50	$\eta = 1.49\text{E-}6, b = 32$	57	0.726	0.732	0.731	0.739
		VGG19	$\eta = 3.98\text{E-}7, b = 16$	73	0.618	0.673	0.626	0.581
Trajectories with ad placeholder		AlexNet	$\eta = 5.75\text{E-}7, b = 64$	32	0.612	0.662	0.629	0.561
		SqueezeNet	$\eta = 2.51\text{E-}7, b = 64$	39	0.616	0.612	0.609	0.607
		ResNet50	$\eta = 7.35\text{E-}6, b = 32$	31	0.646	0.676	0.658	0.602
		VGG19	$\eta = 4.93\text{E-}7, b = 16$	36	0.608	0.690	0.614	0.596
Colored trajectories		AlexNet	$\eta = 2.51\text{E-}7, b = 64$	118	0.620	0.676	0.632	0.607
		SqueezeNet	$\eta = 6.91\text{E-}7, b = 64$	39	0.634	0.527	0.550	0.564
		ResNet50	$\eta = 2.15\text{E-}6, b = 32$	47	0.639	0.687	0.636	0.658
		VGG19	$\eta = 5.58\text{E-}7, b = 16$	53	0.599	0.695	0.606	0.644
Colored trajectories with ad placeholder		AlexNet	$\eta = 2.51\text{E-}7, b = 64$	69	0.606	0.645	0.621	0.570
		SqueezeNet	$\eta = 2.75\text{E-}7, b = 64$	54	0.628	0.659	0.636	0.570
		ResNet50	$\eta = 1.49\text{E-}6, b = 32$	59	0.697	0.725	0.652	0.640
		VGG19	$\eta = 4.36\text{E-}7, b = 16$	59	0.683	0.712	0.688	0.679
Trajectories with line thickness		AlexNet	$\eta = 4.78\text{E-}7, b = 64$	38	0.584	0.639	0.604	0.598
		SqueezeNet	$\eta = 2.51\text{E-}7, b = 64$	62	0.708	0.678	0.683	0.601
		ResNet50	$\eta = 4.93\text{E-}7, b = 32$	57	0.626	0.690	0.632	0.582
		VGG19	$\eta = 5.58\text{E-}7, b = 16$	67	0.655	0.691	0.669	0.567
Trajectories with line thickness and ad placeholder		AlexNet	$\eta = 6.31\text{E-}5, b = 64$	31	0.646	0.454	0.460	0.572
		SqueezeNet	$\eta = 3.02\text{E-}7, b = 64$	47	0.669	0.688	0.676	0.576
		ResNet50	$\eta = 6.31\text{E-}7, b = 32$	33	0.593	0.661	0.615	0.596
		VGG19	$\eta = 2.43\text{E-}6, b = 16$	34	0.670	0.716	0.637	0.573
Colored trajectories with line thickness		AlexNet	$\eta = 5.75\text{E-}7, b = 64$	72	0.553	0.612	0.581	0.528
		SqueezeNet	$\eta = 2.51\text{E-}7, b = 64$	34	0.618	0.673	0.626	0.548
		ResNet50	$\eta = 1.03\text{E-}6, b = 32$	61	0.694	0.711	0.705	0.685
		VGG19	$\eta = 2.51\text{E-}7, b = 16$	73	0.626	0.590	0.605	0.588
Colored trajectories with line thickness and ad placeholder		AlexNet	$\eta = 4.36\text{E-}7, b = 64$	83	0.625	0.662	0.637	0.592
		SqueezeNet	$\eta = 1.90\text{E-}6, b = 64$	33	0.636	0.604	0.622	0.566
		ResNet50	$\eta = 3.86\text{E-}7, b = 32$	49	0.590	0.665	0.609	0.606
		VGG19	$\eta = 4.36\text{E-}7, b = 16$	63	0.646	0.675	0.654	0.633

representation seems to benefit the models' performance, although that finding was not always statistically significant. In addition, the visual representations based on *trajectories* and *colored trajectories with variable line thickness* are consistently found amongst the top-ranked performers. We presume that embedding the temporal dimension into the representations plays a role in accurate prediction of visual attention. Furthermore, we observe that the CNNs that implement shallow architectures (e.g. AlexNet and SqueezeNet) appear to perform equally well, if not better, than their *deeper* counterparts. This suggests that such CNN implementations can attain high accuracy while requiring less bandwidth to operate. Also, the application of transfer learning proved to be useful; hence, reusing existing architectures allows for a quick and inexpensive solution to visual attention prediction, with relatively few training data.

Finally, we should mention that our diagnostic technology was tested for the desktop setting, and currently half of the web traffic is mobile. However, user engagement is still higher on desktop [7]

and amounts for a profitable and sizeable percentage of web traffic. A potential extension is to account for touch-based interactions like, for example, zoom/pinch gestures and scroll activity [28]. Further ideas that could be explored in future work include: benchmark custom CNN architectures (or even combine RNNs and CNNs), analyze other color schema (e.g. for the ad placeholder color), improve the prediction capabilities of our RNNs models (e.g. stacking recurrent layers, using other activation functions, or implementing self-attention mechanisms), and train a general model to predict attention to any direct display. Ultimately, modeling user attention on SERPs has wide-ranging applications in web search ranking and UI design, and this work paves the way to many exciting future directions for research in this topic.

ACKNOWLEDGMENTS

I. Arapakis acknowledges the support of NVIDIA Corporation with the donation of a Titan Xp GPU used for this research.

REFERENCES

- [1] J. Accot and S. Zhai. 1997. Beyond Fitts' Law: Models for Trajectory-based HCI Tasks. In *Proc. CHI*. 295–302.
- [2] J. Accot and S. Zhai. 1999. Performance Evaluation of Input Devices in Trajectory-based Tasks: An Application of the Steering Law. In *Proc. CHI*. 466–472.
- [3] I. Arapakis, M. Lalmas, B. B. Cambazoglu, M.-C. Marcos, and J. M. Jose. 2014. User Engagement in Online News: Under the Scope of Sentiment, Interest, Affect, and Gaze. *J. Assoc. Inf. Sci. Technol.* 65, 10 (2014).
- [4] I. Arapakis, M. Lalmas, and G. Valkanas. 2014. Understanding Within-Content Engagement Through Pattern Analysis of Mouse Gestures. In *Proc. CIKM*.
- [5] I. Arapakis and L. A. Leiva. 2016. Predicting User Engagement with Direct Displays Using Mouse Cursor Information. In *Proc. SIGIR*.
- [6] I. Arapakis, L. A. Leiva, and B. B. Cambazoglu. 2015. Know Your Onions: Understanding the User Experience with the Knowledge Module in Web Search. In *Proc. CIKM*.
- [7] I. Arapakis, A. Penta, H. Joho, and L. A. Leiva. 2020. A Price-Per-Attention Auction Scheme Using Mouse Cursor Information. *ACM Trans. Inf. Syst.* (2020).
- [8] E. Arroyo, T. Selker, and W. Wei. 2006. Usability tool for analysis of web designs using mouse tracks. In *Proc. CHI EA*.
- [9] R. Atterer, M. Wnuk, and A. Schmidt. 2006. Knowing the User's Every Move: User Activity Tracking for Website Usability Evaluation and Implicit Interaction. In *Proc. WWW*.
- [10] J. Azcarraga and M. T. Suarez. 2012. Predicting Academic Emotions Based on Brainwaves, Mouse Behaviour and Personality Profile. In *Proc. PRICAI*. 728–733.
- [11] S. Bai, J. Z. Kolter, and V. Koltun. 2018. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *CoRR arXiv:1803.01271* (2018).
- [12] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. 2011. Algorithms for HyperParameter Optimization. In *Proc. NIPS*.
- [13] Brightfish, Profacts, and Lumen. 2018. From viewable to viewed: using eye tracking to understand the reality of attention to advertising across media. White paper. Retrieved: October 10 2019. Available: https://effectiveviews.be/files/White_Paper_From_Viewable_to_viewed.pdf.
- [14] S. K. Card, W. K. English, and B. J. Burr. 1987. Evaluation of Mouse, Rate-controlled Isometric Joystick, Step Keys, and Text Keys, for Text Selection on a CRT. In *Human-computer Interaction*, R. M. Baecker and W. A. S. Buxton (Eds.). Taylor & Francis, 386–392.
- [15] Y. Chen, Y. Liu, M. Zhang, and S. Ma. 2017. User Satisfaction Prediction with Mouse Movement Information in Heterogeneous Search Environment. *IEEE Trans. Knowl. Data. Eng.* 29, 11 (2017).
- [16] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches.
- [17] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling.
- [18] M. Claypool, P. Le, M. Wased, and D. Brown. 2001. Implicit Interest Indicators. In *Proc. IUI*.
- [19] B. C. Csáji. 2001. *Approximation with Artificial Neural Networks*. Master's thesis. Eötvös Loránd University.
- [20] F. Diaz, R. White, G. Buscher, and D. Liebling. 2013. Robust Models of Mouse Movement on Dynamic Web Search Results Pages. In *Proc. CIKM*.
- [21] A. Diriye, R. White, G. Buscher, and S. Dumais. 2012. Leaving So Soon?: Understanding and Predicting Web Search Abandonment Rationales. In *Proc. CIKM*.
- [22] C. Eickhoff and A. de Vries. 2011. How crowdsourcable is your task?. In *Proc. CSDM Workshop*.
- [23] H. A. Feild, J. Allan, and R. Jones. 2010. Predicting Searcher Frustration. In *Proc. SIGIR*.
- [24] R. Fishkin. 2019. Less than Half of Google Searches Now Result in a Click. Available at <https://sparktoro.com/>.
- [25] A. Graves, A. Mohamed, and G. Hinton. 2013. Speech Recognition with Deep Recurrent Neural Networks. In *Proc. ICASSP*.
- [26] Q. Guo and E. Agichtein. 2008. Exploring Mouse Movements for Inferring Query Intent. In *Proc. SIGIR*.
- [27] Q. Guo and E. Agichtein. 2010. Ready to Buy or Just Browsing?: Detecting Web Searcher Goals from Interaction Data. In *Proc. SIGIR*.
- [28] Q. Guo, H. Jin, D. Lagun, S. Yuan, and E. Agichtein. 2013. Mining Touch Interaction Data on Mobile Devices to Predict Web Search Result Relevance. In *Proc. SIGIR*.
- [29] Q. Guo, D. Lagun, and E. Agichtein. 2012. Predicting Web Search Success with Fine-grained Interaction Data. In *Proc. CIKM*.
- [30] D. J. Hand and R. J. Till. 2001. A Simple Generalisation of the Area Under the ROC Curve for Multiple Class Classification Problems. *Machine Learning* 45, 2 (2001).
- [31] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep Residual Learning for Image Recognition. In *Proc. CVPR*.
- [32] S. Hochreiter and J. Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997).
- [33] K. Hornik. 1991. Approximation Capabilities of Multilayer Feedforward Networks. *Neural Networks* 4, 2 (1991), 251–257.
- [34] J. Huang, R. White, and G. Buscher. 2012. User See, User Point: Gaze and Cursor Alignment in Web Search. In *Proc. CHI*.
- [35] J. Huang, R. W. White, G. Buscher, and K. Wang. 2012. Improving Searcher Models Using Mouse Cursor Activity. In *Proc. SIGIR*.
- [36] J. Huang, R. W. White, and S. Dumais. 2011. No Clicks, No Problem: Using Cursor Movements to Understand and Improve Search. In *Proc. CHI*.
- [37] F. N. Iandola, M. W. Moskewicz, K. Ashraf, S. Han, W. J. Dally, and K. Keutzer. 2016. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size. *CoRR arXiv:1602.07360* (2016).
- [38] P. G. Ipeirotis, F. Provost, and J. Wang. 2010. Quality Management on Amazon Mechanical Turk. In *Proc. HCOMP*.
- [39] S. Johnson, P. Smith, and S. Tucker. 1982. Response format of the job descriptive index: assessment of reliability and validity by the multitrait-multimethod matrix. *J. Appl. Psychol.* 67, 4 (1982).
- [40] R. Jozefowicz, W. Zaremba, and I. Sutskever. 2015. An Empirical Exploration of Recurrent Network Architectures. In *Proc. ICML*.
- [41] A. Kaklauskas, M. Krutinis, and M. Seniut. 2009. Biometric Mouse Intelligent System for Student's Emotional and Examination Process Analysis. In *Proc. ICALT*. 189–193.
- [42] A. Kapoor, W. Bursleson, and R. W. Picard. 2007. Automatic Prediction of Frustration. *Int. J. Hum.-Comput. Stud.* 65, 8 (2007), 724–736.
- [43] P. Kratky and D. Chuda. 2016. Estimating Gender and Age of Web Page Visitors from the Way They Use Their Mouse. In *Proc. WWW Companion*. 61–62.
- [44] A. Krizhevsky, I. Sutskever, and G. E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Proc. NIPS*.
- [45] D. Lagun, M. Ageev, Q. Guo, and E. Agichtein. 2014. Discovering Common Motifs in Cursor Movement Data for Improving Web Search. In *Proc. WSDM*.
- [46] L. A. Leiva. 2011. Restyling Website Design via Touch-based Interactions. In *Proc. MobileHCI*.
- [47] L. A. Leiva and J. Huang. 2015. Building a better mousetrap: Compressing mouse cursor activity for web analytics. *Inf. Process. Manag.* 51, 2 (2015).
- [48] L. A. Leiva and R. Vivó. 2013. Web Browsing Behavior Analysis and Interactive Hypervideo. *ACM Trans. Web* 7, 4 (2013).
- [49] Y. Liu, Y. Chen, J. Tang, J. Sun, M. Zhang, S. Ma, and X. Zhu. 2015. Different Users, Different Opinions: Predicting Search Satisfaction with Mouse Movement Information. In *Proc. SIGIR*.
- [50] Y. Liu, C. Wang, K. Zhou, J. Nie, M. Zhang, and S. Ma. 2014. From Skimming to Reading: A Two-stage Examination Model for Web Search. In *Proc. CIKM*.
- [51] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang. 2017. The Expressive Power of Neural Networks: A View from the Width. In *Proc. NIPS*. 6231–6239.
- [52] I. S. MacKenzie, T. Kauppinen, and M. Silfverberg. 2001. Accuracy Measures for Evaluating Computer Pointing Devices. In *Proc. CHI*. 9–16.
- [53] E. Martin and C. Cundy. 2018. Parallelizing Linear Recurrent Neural Nets Over Sequence Length. In *Proc. ICLR*.
- [54] D. Martín-Albo, L. A. Leiva, J. Huang, and R. Plamondon. 2016. Strokes of insight: User intent detection and kinematic compression of mouse cursor trails. *Inf. Process. Manag.* 52, 6 (2016).
- [55] V. Navalpakkam, L. Jentzsch, R. Sayres, S. Ravi, A. Ahmed, and A. Smola. 2013. Measurement and Modeling of Eye-mouse Behavior in the Presence of Nonlinear Page Layouts. In *Proc. WWW*.
- [56] J. W. Owens, B. S. Chaparro, and E. M. Palmer. 2011. Text Advertising Blindness: The New Banner Blindness? *J. Usability Stud.* 6, 3 (2011).
- [57] A. Pentel. 2017. Predicting Age and Gender by Keystroke Dynamics and Mouse Patterns. In *Proc. Adj. UMAP*.
- [58] J. Resig, B. Bibault, and J. Maras. 2016. *Secrets of the JavaScript Ninja* (2nd ed.). Manning Publications.
- [59] B. Shapira, M. Taieb-Maimon, and A. Moskowit. 2006. Study of the Usefulness of Known and New Implicit Indicators and Their Optimal Combination for Accurate Inference of Users Interests. In *Proc. SAC*.
- [60] Sharethrough and IPG. 2013. *Native Ads Vs Banner Ads: Native Ad Research from IPG & Sharethrough Reveals that In-Feed Beats Banners*. Technical Report.
- [61] K. Simonyan and A. Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *Proc. ICLR*.
- [62] L. N. Smith. 2015. Cyclical Learning Rates for Training Neural Networks.
- [63] M. Speicher, A. Both, and M. Gaedke. 2013. TellMyRelevance!: Predicting the Relevance of Web Search Results from Cursor Interactions. In *Proc. CIKM*.
- [64] R. D. Wright and L. M. Ward. 2008. *Orienting of Attention*. Oxford University Press.
- [65] T. Yamauchi. 2013. Mouse Trajectories and State Anxiety: Feature Selection with Random Forest. In *Proc. ACII*. 399–404.
- [66] T. Yamauchi and C. Bowman. 2014. Mining Cursor Motions to Find the Gender, Experience, and Feelings of Computer Users. In *Proc. ICDMW*. 221–230.
- [67] A. X. Zheng and M. Bilenk. 2013. Lazy Paired Hyper-Parameter Tuning. In *Proc. IJCAI*.
- [68] P. Zimmermann, S. Guttormsen, B. Danuser, and P. Gomez. 2003. Affective Computing – A Rationale for Measuring Mood With Mouse and Keyboard. *Int. J. Occup. Saf. Ergon.* 9 (2003), 539–51.