AN AUTOMATED VIDEO CONTENT CUSTOMIZATION SYSTEM USING EYE TRACKING AND ARTIFICIAL INTELLIGENCE

Yuyang Lou[1] and Yu Sun[2]

[1]Charles Wright Academy, 7723 Chambers Creek Rd W, Tacoma, WA 98467
[2]California State Polytechnic University, Pomona,
CA, 91768, Irvine, CA 92620

ABSTRACT

In the past few years, the internet and online social networks developed drastically, promoting the development of online learning programs. These programs provided opportunities for a digital learning experience that allows students to explore beyond what's taught in school. However, having a clear understanding of what topic might interest the user and motivate the user to further explore that topic is hard for both the user and the learning program. This paper proposes to create one centralized method of predicting what the user would be interested in and provide them with educational content recommendations. Our design builds upon the eye-tracking techniques, which allows us to capture users' eye movements, and object recognition achieved by machine learning, which allows us to examine the specific object that the users are looking at and provide data for the users' interest analysis [3]. Our results show a success rate of 90% of analyzing what the user is truly looking at. We used our decision heuristic, etc.

KEYWORDS

Eye Tracking, Deep learning, computer vision.

1. INTRODUCTION

Moving into the next decade, individual interest and the inclusion of various communities became increasingly important when companies are designing products [1][2]. Content recommendation based on the analysis of these individual interests, as a result, grows more and more popular in many fields [3]. In the industry of entertainment, various companies use content recommendations to attract their users and keep them entertained [6]. In the industry of online shopping, content recommendations are utilized to help the customers discover their favorite item and increase the likelihood that an item is going to be purchased. In the advertisement industry, content recommendation has become one of the most important aspects to attract users to their potential interests. However, in the field of education, the potential of content recommendation has not yet been fully developed. While a person might not be fully aware of what they are interested in the most during their learning process, wandering on learning platforms aimlessly and browsing content from various topics is very time-consuming and not helpful in determining their path of interest. Analyzing their interest throughout their learning process by tracking their behaviors and recommending them on more topics based on the analysis, can direct the users to their interest and speed up the exploration process drastically. This project, therefore, attempts to build a content recommendation system based on analysis of users' interests, aiming to provide a

David C. Wyld et al. (Eds): AIAP, IT, MoWiN, CCSEIT, CNSA, ICBB - 2022
pp. 39-48, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.120403

learning experience that is intriguing to the users and allows them to further study and exceed on topics they are truly passionate about.

Before the existence of the internet, advertisements, commercials on televisions and radios, and billboards were the only access companies had to market their products [7]. To improve its efficiency while bound by the technological limitations at the time, people started to deliver those advertisements by geographical interests, the first form of a content recommendation system based on users' interests. Several decades later, some of the most popular methods of creating such a system focus on the analysis of users' social networks and the construction of an evaluation system that categorizes different products for different needs [8]. These methods are heavily implemented in real life, making impacts daily. Entertainment platforms such as YouTube and Netflix rely heavily on their evaluation system that categorizes movies and videos into different genres, which in turn allows them to recommend to users similar products that they have been previously enjoying. On the other hand, social media such as Twitter and Instagram, while also implementing an evaluation system, also analyzes its users' social network information. By portraying a social network graph, social media can capture users' connections with each other, similar interests among a group of users. However, these techniques are never used alone. Companies such as amazon utilize a combination of the two, recommending users similar products they have previously bought based on their evaluation metrics of the products, and products favored by similar users through their analysis of the user's social network graph. These techniques are relatively cheaper and practical yet consist of disadvantages too. Programs utilizing these techniques, at best, can only give estimations of a user's potential interests, because the data it is processing are all secondary data that does not directly represent users' true intention, and sometimes waste time and effort marketing the wrong product to the wrong group [9].

In this paper, we follow the same line of research to construct an evaluation metric that can help us categorize our products and better deliver them to our users of similar interests. However, the process of determining the users' potential interest is different from the existing ways elaborated in the former paragraph. Here, we took a more direct approach, extracting information regarding the users' eye movement utilizing eye-tracking equipment, and a video that the users are watching [4]. We then process the video through a machine learning model, recognize the objects occurring in the video, and analyze the objects that the users' eyes are focusing on, in turn predicting the users' potential interest. As human eye movement is a way to express their emotion, this method is a more direct way of predicting users' potential interest and delivering them similar products. In the future, there are many ways in which this concept can be put into practical use and benefit our community. In the field of education, this technique can help children develop their interest and explore their topics of interest at a young age and help them learn about vocabularies by staring at different pictures corresponding to different words.

In two application scenarios, we demonstrate the practicality and accuracy of the model in terms of predicting the user's interest after watching a video. In the first experiment, we tested our model after adjusting the parametric to analyze its performance at recognizing objects in the video. We make amends to the number of batches the model is trained on, the number of epochs the model will iterate through, and train the model with a dataset with different sizes [10]. Its accuracy is shown by the graphs and data collected after the experiment. In this case, the model yielded a result of over 80% of accuracy and around 90% of precision. In the second experiment, we tested the practicability of the model when analyzing the users' interest through the data collected when they are watching the video. A questionnaire to the users is performed after the model makes its prediction, to reflect on the percentage of error between users' intended interests and the prediction. By linking the program to the Google search engine, the program reached a considerable amount of accuracy at predicting users' interest and recommending them further

content of interest. The model indeed exhibits a percentage of error that is not negligible, and future adjustment of the evaluation metric will likely solve this problem. Through the above two experiments, we can show the accuracy and practicality of the program, given its performance on object recognition and user interest analysis. We are also aware of the necessary future improvement to withstand every scenario.

The rest of the paper is organized as follows: Section 2 gives the details on the challenges that we met during the experiment and designing the sample; Section 3 focuses on the details of our solutions corresponding to the challenges that we mentioned in Section 2; Section 4 presents the relevant details about the experiment we did, following by presenting the related work in Section 5. Finally, Section 6 gives the concluding remarks, as well as points out the future work of this project.

2. CHALLENGES

To build the project, a few challenges have been identified as follows.

2.1. Implementing/combining the YoloV5 established model into the rest of our codebase can often lead to issues

Integrating other pre-developed programs into our code can be both very time-consuming and troublesome. The YoloV5 libraries that we are using have pre-built programs that perform object detection. To extract information for our use in the eye-tracking system, it is necessary to make adaptations to both systems, which requires a lot of work [5]. Generally, the solution is to write the program around the libraries that we intended to integrate into our system. In our case, however, the eye-tracking program that we built is a new system. Our solution to this problem then is to scan through the YoloV5 library, interpret the functions of its code, and make changes to its scripts to integrate it into our eye-tracking system.

2.2. Designing a content viewing tracking system with the mouse and human eyes is tough to do without it being slow/hard-to-use

Building a real-time content viewing tracking system is very hard because object detection consumes a lot of time and that leads to a sub-optimal user experience. Real-time object detection is often too slow for proper viewing experiences due to constraints in hardware. If a real-time tracking system is implemented, frames of the videos will drop significantly, leading to negative user experiences. Generally, there are several solutions to the problem. On one hand, we can add more computational power, such as multiple processors, to the system to support the calculation needed for object detection or find better algorithms that improve the efficiency of the program. On the other hand, we can pre-process the video to not cause the slow-down from real-time object detection. Due to our limitation in hardware resources and the intended purpose of this program to be publicized, we decided to pre-process the videos and implement a real-time eye-tracking system, so that the user is ensured fluency in their experience without requiring too much computational power.

2.3. Designing a decision heuristic network requires a lot of training and adaptations to accurately analyze the user's interests

Due to the inconsistency of Users' behavior, decision heuristic networks will need a lot of data to adapt to various conditions. In our real-time eye-tracking system, data regarding users' eye movement must be captured in real-time and analyzed. It imposed a challenge for us to grasp the

data and analyze it in a way that is not too time-consuming to affect user experiences. Generally, to solve the problem the data of users' eye movement are recorded and analyzed at the end of the video, so the analysis will not affect the user experience. In addition to the collection of the data and an overall analysis of the data at the end of the video, we built a user interface using Tkinter that allows us to create a display right next to the video, displaying the result when the program performs real-time interest analysis.

3. SOLUTION

The system is a user interest analysis system based on object recognition and eye-tracking techniques. The system integrates the yolov5 machine learning model to perform object detection on the videos and output the video's corresponding data. The model is trained on various topics of interest, to perform accurate object detection on videos regarding those topics. This object detection is then pre-run before creating the system to decrease real-time object recognition delays. When the users begin to interact with the video and give input with their eyes or the mouse, the system then retrieves the corresponding datasets of the video acquired during the pre-running process and monitors the users' eye or mouse movements. While the user is watching the video, the system will be responsive to the user's eye movement/mouse movements, the objects they are currently looking at by referring to the corresponding datasets and giving real-time recommendations to articles and web links based on the objects. In addition to the real-time recommendations, the system also records users' eye or mouse movements and performs overall analysis at the end of the video, to give more accurate suggestions based on a heuristic network. The system also provides a modern user interface and convenient function for users to manipulate the system and grant them a great user experience (See Figure 1 for the system's overall structure).

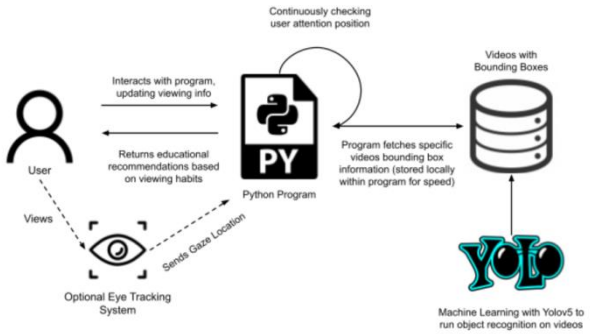

Figure 1. An outline of the Interest Analysis System

To train the model, we searched for videos of different topics of interest such as biology, astronomy, rocket science, and extracted frames out of those videos as our training set using OpenCV libraries. We also added additional distorted versions of the images to create a stronger dataset using a tool called Roboflow. We then hand-labeled the images, generated a yolov5 file, and trained the model with it. To output the video's data during the pre-running process, which contains the bounding box information of the objects, we made changes in the yolov5 file and extracted that information to a separate text file. To achieve real-time interaction between the user and the system, we built a python application using Tkinter and OpenCV, which provided convenient library features to track the user's eye or mouse movements and integrated the Tobii eye-tracking system into the application, which allows us to access data regarding the users' eye movements. We also made adaptations in the yolov5 files so that every time the system detects a change in the users' eye or mouse movements, it refers back to the text file generated during the pre-running process, access the bounding box information, and determine the objects the users

are currently looking at. To achieve real-time recommendation, we integrated google search into the function so that when the program receives data of the objects that the users are looking at, it looks for information regarding the objects on google and shows five web links, based on their popularity, in the application. The links are integrated into a clickable label and the user will be able to access the web link instantly as they are looking at the object. To make this process more efficient, we also built a dictionary that keeps track of searched items, and their corresponding search results, and make sure that no repeated work has been done to interrupt the users' experience. The program also stores the data to the end of the video and performs more sophisticated analysis on the users' eye movement, in terms of the percentage of the time the users are looking at a specific object, or how many consecutive seconds the users are looking at a single object. The users will be able to access such data through a stats button. This information is also utilized by the recommendation system to produce a more personalized recommendation regarding the users' major interests. Finally, to make the UI modern and convenient to use, we designed the layout of the application, so the widgets are aligned with each other to bring comfort to the users aesthetically. The media player occupies most of the screen space in the application, and the real-time recommendations and the data appear on the right column of the application. There is also a list of videos the users can play at the bottom of the application. These features add to the simplicity of the application and make it easier to interact.

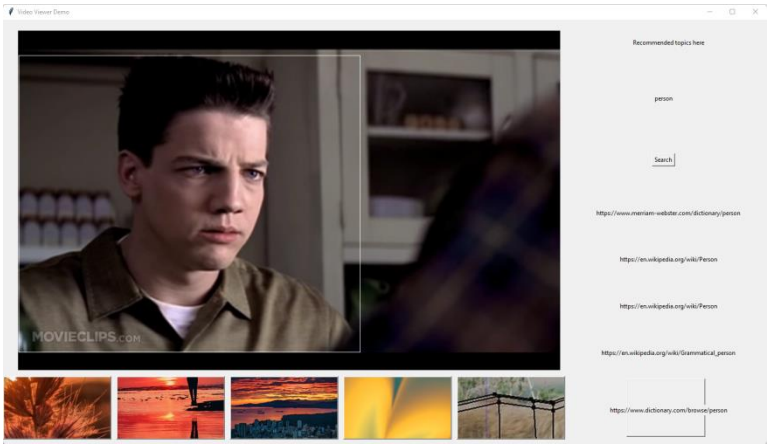

Figure 2. A screenshot of the application

```
def motion(self, event):

    x = event.x - 5
    y = event.y - 5
    for bbox in self.curr_bboxes:
        min_x = bbox[0]
        min_y = bbox[1]
        max_x = bbox[2]
        max_y = bbox[3]

        if x >= min_x and x <= max_x and y >= min_y and y <= max_y:
            print("Currently looking at {}".format(bbox[4]))
            if self.watch_timer.get(bbox[4]):
                self.watch_timer[bbox[4]] += 1
            else:
                self.watch_timer[bbox[4]] = 1
            if bbox[4] != self.mostrecentlylookedat:
                print(self.seen_words)
                if self.seen_words.get(bbox[4]):
                    self.search_label.configure(text=str(bbox[4]))
                    numlink = 5
                    links = []
                    eval_link = lambda x: (lambda p: webbrowser.open_new(x))
                    for i in range(numlink):
                        links.append(Label(self.root, text=self.seen_words[bbox[4]][i]))
                        links[i].grid(row=i + 3, column=5)
                        links[i].bind('<Button-1>', eval_link(self.seen_words[bbox[4]][i]))
                    print(links)

                else:
                    self.mostrecentlylookedat = bbox[4]
                    self.search_label.configure(text=str(bbox[4]))
                    data = str(self.mostrecentlylookedat)
                    results = search(data)
                    self.seen_words[bbox[4]] = results
                    numlink = 5
                    links = []
                    eval_link = lambda x: (lambda p: webbrowser.open_new(x))
                    for i in range(numlink):
                        links.append(Label(self.root, text=results[i]))
                        links[i].grid(row=i + 3, column=5)
                        links[i].bind('<Button-1>', eval_link(results[i]))
```

Figure 3. A segment of the code that allows the program to track users' eye and mouse movements

```
def videoLoop(self):
    try:
        timer = FPS().start()
        while not self.stopEvent.is_set() and not self.paused:


            self.grabed, self.frame = self.vs.read()

            if self.frame is None:
                break
            self.frame = imutils.resize(self.frame, width=1080)

            image = cv2.cvtColor(self.frame, cv2.COLOR_BGR2RGB)
            image = Image.fromarray(image)
            #Update the bounding boxes for said image
            self.update_curr_bboxes(self.framenumber)
            for bbox in self.curr_bboxes:
                min_x = bbox[0]
                min_y = bbox[1]
                max_x = bbox[2]
                max_y = bbox[3]
                draw = ImageDraw.Draw(image)
                draw.line((min_x, max_y, max_x, max_y))
                draw.line((min_x, min_y, max_x, min_y))
                draw.line((max_x, min_y, max_x, max_y))
                draw.line((min_x, min_y, min_x, max_y))

            image = ImageTk.PhotoImage(image)
            self.framenumber += 1
            if self.panel is None:
                self.panel = Label(image=image)
                self.panel.image = image
                self.panel.grid(column=0, row=0, columnspan=5, rowspan=7, padx=self.display_padx, pady=self.display_pady)

            else:
                self.panel.configure(image=image)
                self.panel.image = image
        timer.stop()
        print(timer.fps())
        print(timer.elapsed())
```

Figure 4. A segment of the code that streams the video

4. EXPERIMENT

4.1. Experiment 1

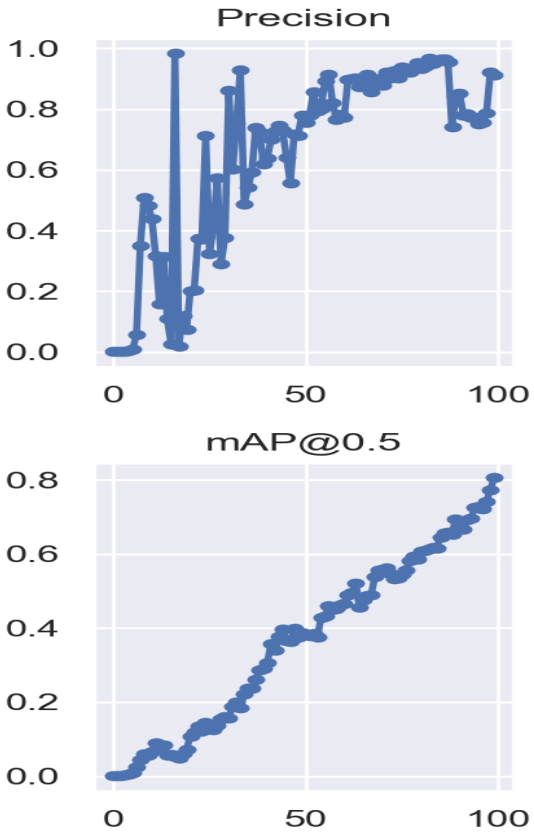

Figure 5. Result of the experiment

After adjusting the parameters multiple times, the results show that a relatively lower number of batches and a higher number of epochs will result in the most optimally trained model for our experiment. We initially started training the model on only 1 batch and 10 epochs, yielding an accuracy of at most 10%. When we tried to train the model with over 16 batches, the program crashed due to the limitations of our hardware. Eventually, we decided to increase the number of epochs and keep the number of batches around 16. As the number of epochs increased, accuracy and precision improved and when the number of epochs reached 100, the model yielded a result of over 80% of accuracy and around 90% of precision, which is a major improvement over our initial trial. The accuracy and precision of the model only showed slight improvement when the number of epochs increased over 100.

Finding training metrics that result in the most optimally trained model allows us to train our model with more unique objects with higher object recognition accuracy, eventually improving the overall accuracy at analyzing users' interests. We trained on the model with a specific dataset. We adjust the parametric regarding the number of epochs and batches we are training the model on and analyze the performance of the model when the training is complete, and tests are performed on the model. Tests are performed multiple times each time we adjust the parametric, to ensure that the outliner will have a lesser impact on the overall result.

4.2. Experiment 2

With higher accuracy at analyzing users' interests, the system will be able to produce more sophisticated and personalized recommendations, improving users' experience when interacting with the system and having a further understanding of their interests.

To measure the system's extent of success in analyzing the users' interests, we run the system and record how accurate the program is at tracking users' eye or mouse movements and recognizing the objects they are focused on. Analyzing the program's success at cultivating interest and the relevance between the system's recommendations and the users' real interests is done by user feedback.

When interacting with the system, the program shows major success at tracking the users' eye or mouse movement quickly and accurately, in turn allowing successful object recognition. The program's interpretation of the users' interests, however, wasn't as successful. At the end of each video stream, I usually found 40% of the suggestions that the program interpreted as my interests corresponding to my real interests, partly since the correlation between what the users are looking at and their real interests aren't always stable, and people's interests can be affected by measures other than their vision. The weblinks that the program provides also show low relevance between the links and the users' real interests, primarily because we are choosing the most popular web pages that have low correspondence to personal background.

Based on the two experiences we have done; we can prove that our solution has both stable performance and high accuracy.

5. RELATED WORK

In this article, the team utilizes SNS (social network service) data and eye-tracking data to create a preference metric, and yield user preferences for categories [11]. While this research leans more on the analysis of SNS data using eye-tracking techniques, my work focuses primarily on the eye-tracking data extracted directly from the media by detecting the objects the users are looking at using machine learning. Compared to their work, my research proposed a more direct approach toward user preference analysis but overlooks the effect of SNS on user preference analysis.

In this article, the team attempts to achieve accurate event detection, that is, the process of analyzing event streams to recognize the event types [12]. They implemented machine learning to train the classifiers and performed event detection based on the classifier. Compared to their work, my research approaches event detection by pre-training the media that the users are looking at, and yielding results based on their eye-movement data. My work is more efficient in terms of time, however, less flexible when performing live updates.

In this article, the team utilizes analysis on the shape and the color features of the objects to achieve object detection [13]. The team extracts the length=width ratio as the shape information of the object and extracts the color histogram to determine the color of the object. Eventually, a video surveillance system is implemented. While they focus on the color and shape of the objects to achieve object detection, my project focuses on the heaving training and the large dataset built into the system, which allows it to extract features itself and classify the object based on those features. My work is more accurate and more flexible, however, requires a lot of data training [14].

6. CONCLUSIONS

In this project, we proposed a way of analyzing users' interest while browning videos and articles, by constructing a program that can analyze users' eye movement on the screen and recognize the objects their eyes are focusing at through implementation of the Yolov5 machine learning model and the use of eye-tracking equipment. Through real-time recommendations and post-video analysis, we intend to provide the users with a comfortable and personal entertainment experience while exploring their field of interest. The program will also implement a modern user interface, so it is suitable for everyone. In this project, we designed two experiments to evaluate the success of our program. We tested its accuracy at recognizing objects in the video by training the model with different parametric, specifically the batches the model is running with and the number of epochs the model will be trained on. We also managed to measure the margin of error between the model's prediction of the user's interests and the user's real interests by performing a set of questionnaires to the users after the program delivered them recommendations. The result of these two experiments proved the effectiveness and the practicality of the program. The program was able to recognize objects at a precision of over 90% and an accuracy over 80%, it also predicts users' interests at a considerable accuracy, over 50% accuracy on average.

However, current limitations of the program exist and are not negligible in the future development of the program. Due to hardware limitations, the machine learning model is not able to train on many batches, which in turn undermines its accuracy at performing object recognition [15]. The program is also limited in its current prediction of user's interest, and with the percentage of error displayed from the experiments, The program will not exceed other measures to analyze users' interest. Currently, there still exists a gap between what the user's real intentions are when watching the video and what the program interprets the users' eye movement as an indicator of their preference.

To make sure that the application's hardware requirement is comfortable and affordable to the majority, we will tackle the technical limitations by controlling the quality of the training dataset and possibly replacing our current machine learning model to reach a higher accuracy when recognizing objects from the video. We will also begin to construct evaluation metrics using machine learning in the future to process users' eye movement data and give them more reasonable recommendations based on the processed data.

REFERENCES

[1] Mitchell, J. Clyde. "Social networks." Annual review of anthropology 3.1 (1974): 279-299.
[2] Twigg, Carol A. "Models for online learning." Educause review 38 (2003): 28-38.
[3] Tanenhaus, Michael K., and Michael J. Spivey-Knowlton. "Eye-tracking." Language and Cognitive processes 11.6 (1996): 583-588.
[4] Holmqvist, Kenneth, et al. Eye tracking: A comprehensive guide to methods and measures. OUP Oxford, 2011.
[5] Krafka, Kyle, et al. "Eye tracking for everyone." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
[6] Vogel, Harold L. Entertainment industry economics: A guide for financial analysis. Cambridge University Press, 2020.
[7] Beaver, William H. "Market efficiency." Accounting Review (1981): 23-37.
[8] Arreola, Raoul Albert. Developing a comprehensive faculty evaluation system. Madison, WI: Magna Publications, 2004.
[9] Vartanian, Thomas P. Secondary data analysis. Oxford University Press, 2010.
[10] Epstein, Joshua M. "Why model?" Journal of artificial societies and social simulation 11.4 (2008): 12.
[11] Song, Hyejin, and Nammee Moon. "Eye-tracking and social behavior preference-based recommendation system." The Journal of Supercomputing 75.4 (2019): 1990-2006.

[12] Zemblys, Raimondas, et al. "Using machine learning to detect events in eye-tracking data." Behavior research methods 50.1 (2018): 160-181.
[13] Wu, Jun, and Zhitao Xiao. "Video surveillance object recognition based on shape and color features." 2010 3rd International Congress on Image and Signal Processing. Vol. 1. IEEE, 2010.
[14] Pazzani, Michael J., and Daniel Billsus. "Content-based recommendation systems." The adaptive web. Springer, Berlin, Heidelberg, 2007. 325-341.
[15] Zhang, Xian-Da. "A matrix algebra approach to artificial intelligence." (2020): 803.