

HTML Based Cursor Classifier

Muhammad Ahmad Nazir
370307
BEE-13B

School of Electrical Engineering
and Computer Sciences

Abdul Hadi
374120
BEE-13B

School of Electrical Engineering
and Computer Sciences

Hamza Atiq
368099
BEE-13B

School of Electrical Engineering
and Computer Sciences

Abstract

This project proposes a solution for assessing student engagement by monitoring and analyzing cursor movement patterns on digital documents. By leveraging cursor data such as movement, scrolling and time the system classifies whether a student is actively engaged or disengaged while viewing content. The tool will be compatible with an HTML-based environment, allowing for integration into various web-based educational platforms. This cursor-based engagement classifier aims to enhance learning analytics by providing real-time insights into student interaction, thus improving educational outcomes through adaptive content delivery and personalized feedback.

1. Introduction

As online learning becomes increasingly prevalent, educators face challenges in accurately assessing student engagement during remote sessions. Traditional methods such as video monitoring are intrusive and can raise privacy concerns, whereas attention-detection algorithms that rely on direct interaction are often limited in accuracy. Cursor tracking offers a non-intrusive method of understanding user behavior, as it provides data on attention and engagement through patterns in cursor movement, clicks, and scrolling. This project aims to develop a web-based cursor engagement classifier that can determine the likelihood of a student being engaged with educational content, helping educators and developers understand and adapt to user needs effectively.

2. Problem Statement

Online education platforms often lack the ability to accurately measure student engagement. Standard methods such as quizzes and time-based activity tracking do not provide real-time insights into student focus or engagement with the content. There is a need for an effective, non-intrusive way to monitor student behavior while they interact with digital content, allowing educators to respond to engagement levels appropriately.

3. Project Requirements

The requirements for this project involve tracking and analyzing cursor patterns and ensuring smooth operation in an HTML-based environment.

3.1. Track Cursor Movement

Utilize software capable of logging cursor coordinates (x, y) to capture movement, scroll and time actions on a webpage.

3.2. Analyze Cursor Patterns

Classify cursor data to determine whether the student is engaged or disengaged.

3.3. HTML Compatibility

Ensure the solution works smoothly within an HTML browser environment, with minimal impact on page load times or performance.

4. Methodology and Results

The methodology for this project involves several stages, including data collection, feature extraction, and machine learning model training. The system works as follows:

4.1. Data Collection

The cursor movement and time are tracked on an HTML-based webpage designed to display a PDF document. This data is captured through a combination of mouse events and time intervals to monitor user engagement. The key components of the data collection process are as follows:

- **Cursor Movement:** The position of the cursor is tracked as the user moves it over the PDF viewer. The cursor's x and y coordinates are recorded whenever the cursor moves by a significant distance (greater than a defined threshold).
- **Tracking Interval:** The cursor position is logged at regular intervals (every 100 milliseconds) to create a detailed record of the user's interaction with the content.
- **Time Stamps:** Each movement is associated with a timestamp, calculated relative to the session's start time. This allows for tracking the duration of user activity and identifying periods of inactivity.
- **Data Storage:** The collected cursor data, including the x and y coordinates and corresponding timestamps, are stored in an array called `cursorData`. This array is periodically updated during the tracking session.
- **Data Export:** At the end of the tracking session, the cursor data is saved as a JSON file. This file contains the full set of cursor positions and timestamps, which can be downloaded for further analysis. The download is triggered by the user clicking the "End Session" button.

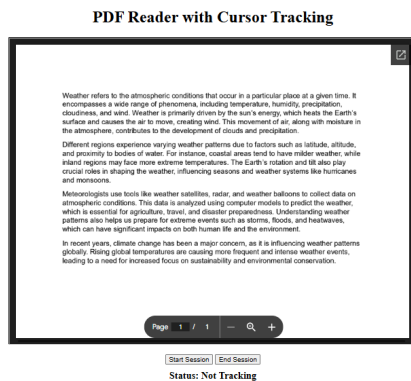


Figure 1. Web Based Interface

Data Collection Type: The cursor movement, time, and scroll data are tracked on an HTML-based webpage. This data is stored in a JavaScript (.js) file upon interaction with the page. The collection process is divided into two categories based on the level of user engagement: **engaged** and **disengaged** data.

1. Engaged Data:

Engaged data is collected when the user demonstrates focused interaction with the content. Specifically, it is captured when the user moves the cursor in a controlled manner, such as from one text block to another horizontally. This type of movement typically reflects a user's attention to the content, as they are actively reading or interacting with specific elements on the page. The cursor is expected to follow a smooth, deliberate path with minimal random movement.

total_dist	average_s	speed_std	num_mov	idle_time	direction	coverage	label
5691.771	165.396	391.4561	300	4.192	21	0.001021	1
5656.07	182.4421	377.1723	265	4.231	23	0.00092	1
5490.868	169.9275	360.5734	268	5.258	31	0.001167	1
5692.966	164.5271	375.3678	299	4.27	19	0.000965	1

Figure 2. Engaged Data

2. Disengaged Data:

Disengaged data is collected under two conditions:

- **Idle State:** The user does not move the cursor at all for a prolonged period, indicating a lack of interaction with the content.

total_dist	average_s	speed_std	num_mov	idle_time	direction	coverage	label
842.6856	20.35227	221.6143	7	40.508	7	0.000222	0
696.7722	13.12164	156.0234	5	51.996	4	9.53E-05	0
836.8328	13.73974	170.4791	6	60.097	4	5.63E-05	0
736.1359	20.49038	185.0753	7	36.005	5	0.000212	0

Figure 3. Disengaged Idle Data

- **Erratic Movement:** The user moves the cursor rapidly or randomly, often without focusing on specific content. This could resemble behaviors like playing with the mouse or moving the cursor erratically across the screen, which suggests a disengaged or distracted state.

total_dist	average_s	speed_std	num_mov	idle_time	direction	coverage	label
16452.68	462.1279	342.1685	342	0.601	331	0.00118	0
19201.29	601.7515	561.4725	297	1.799	245	0.000906	0
26016.3	508.0515	385.9563	501	0.914	444	0.001528	0
20140.38	357.0863	233.9369	534	2.174	523	0.001723	0

Figure 4. Disengaged Erratic Data

The data collection is initiated by the user clicking the "Start Session" button. Once the session begins, the system continuously monitors the cursor position, recording relevant data until the "End Session" button is clicked. The

collected data is then available for download in JSON format, allowing educators or researchers to analyze the engagement levels based on cursor movement patterns. The interface is given below.

4.2. Data Preprocessing and Features Extraction

After collecting the cursor data, it is processed into a structured format to facilitate effective analysis. The raw data, stored in JSON files, contains information about cursor movements and timestamps. The preprocessing steps outlined below help to transform this raw data into meaningful features that can be analyzed to assess user engagement.

1. **Data Loading:** The raw cursor data from each session is loaded into a Pandas DataFrame. Each session consists of a series of cursor positions (x, y), timestamps, and other relevant information. The data is read from JSON files and organized for further processing.

```
[{"x": 459, "y": 144, "timestamp": 202}, {"x": 508, "y": 475, "timestamp": 312}, {"x": 459, "y": 480, "timestamp": 320}, {"x": 559, "y": 344, "timestamp": 340}, {"x": 518, "y": 284, "timestamp": 363}, {"x": 483, "y": 249, "timestamp": 379}, {"x": 463, "y": 217, "timestamp": 380}, {"x": 451, "y": 186, "timestamp": 411}, {"x": 448, "y": 177, "timestamp": 428}, {"x": 429, "y": 150, "timestamp": 460}, {"x": 449, "y": 145, "timestamp": 463}, {"x": 469, "y": 133, "timestamp": 474}, {"x": 484, "y": 116, "timestamp": 486}, {"x": 399, "y": 113, "timestamp": 512}, {"x": 396, "y": 108, "timestamp": 520}, {"x": 396, "y": 106, "timestamp": 540}, {"x": 396, "y": 104, "timestamp": 562}, {"x": 395, "y": 101, "timestamp": 570}, {"x": 395, "y": 99, "timestamp": 580}]
```

Figure 5. Collected Data from Webpage

2. Feature Extraction:

- **Movement Metrics:** The change in cursor position between consecutive points is calculated. Additionally, the time difference between movements is computed. The Euclidean distance traveled by the cursor is derived from these values.
- **Total Distance:** The total distance covered by the cursor during the session is computed by summing the individual movement distances. This provides an overall measure of cursor activity.
- **Average Speed:** The average speed of the cursor is calculated by dividing the total distance by the total time of the session. This metric provides insight into the overall pace of cursor movements.
- **Speed Variability:** The standard deviation of cursor speed is calculated to capture the variability in movement speed during the session.
- **Movement Count:** The number of significant movements is counted, where a movement is defined as a cursor displacement greater than a pre-defined threshold (5 pixels).
- **Idle Time:** Idle time is calculated as the total time the cursor remains stationary, i.e., when the distance between consecutive positions is zero.

- **Direction Changes:** The angle of movement between consecutive cursor positions is calculated. Significant changes in direction (greater than a threshold, 30 degrees) are counted to assess the frequency of direction shifts during the session.

3. **Data Structuring:** After the features are extracted, they are organized into a structured format. Each session's features, including metrics such as total distance, average speed, speed variability, number of movements, and idle time, are stored in a feature list. This list is then converted into a Pandas DataFrame for easy analysis.
4. **Feature Labeling:** Each session is labeled based on the engagement level. Sessions from the "engaged" directory are labeled as 1, while those from the "disengaged" directory are labeled as 0. This labeling allows for the classification of sessions into engaged and disengaged states, facilitating supervised learning.
5. **Final Output:** The processed feature data for all sessions is compiled into a CSV file, which can then be used for further analysis or machine learning model training. This CSV file contains all the extracted features, with missing values filled in, ensuring the data is ready for subsequent stages of analysis.

By performing these preprocessing steps, the raw cursor data is transformed into a structured dataset with meaningful features that can be used to evaluate user engagement levels during PDF reading sessions.

total_dist	average_s	speed_std	num_mov	idle_time	direction	coverage	label
5691.771	165.396	391.4561	300	4.192	21	0.001021	1
5656.07	182.4421	377.1723	265	4.231	23	0.00092	1
5490.868	169.9275	360.5734	268	5.258	31	0.001167	1
5692.966	164.5271	375.3678	299	4.27	19	0.000965	1
842.6856	20.35227	221.6143	7	40.508	7	0.000222	0
696.7722	13.12164	156.0234	5	51.996	4	9.53E-05	0
836.8328	13.73974	170.4791	6	60.097	4	5.63E-05	0
756.1359	20.49038	185.0753	7	36.005	5	0.000212	0
16452.68	462.1279	342.1685	342	0.601	331	0.00118	0
19201.29	601.7515	961.4725	297	1.799	245	0.000906	0
26016.3	508.0515	385.9563	501	0.914	444	0.001528	0
20140.38	357.0863	233.9369	534	2.174	523	0.001723	0

Figure 6. Final Features

4.3. Model Training

The collected and preprocessed data is used to train a machine learning model to classify user engagement based on cursor behavior. The process involves several key steps:

1. Loading the Dataset:

The processed dataset, saved in CSV format, is loaded into the system. This dataset contains features extracted from cursor movement data, along with corresponding labels indicating engaged or disengaged states.

2. Data Preparation:

The dataset is divided into features (input variables) and labels (target variable). Features represent the extracted metrics such as total distance, average speed, number of movements, and idle time, while labels indicate the engagement level.

3. Splitting the Data:

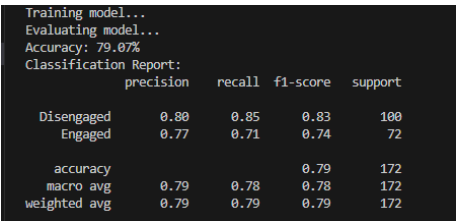
The data is split into training and testing sets to evaluate the model's performance. A stratified split ensures that both engaged and disengaged data are proportionally represented in the training and testing sets. 80% of the data is used for training, and 20% is reserved for testing.

4. Model Training:

A Random Forest Classifier is used as the machine learning model. This ensemble-based algorithm is trained on the training set to learn patterns that distinguish engaged from disengaged behavior. Hyperparameters such as the number of trees in the forest are set to default values for training.

5. Model Evaluation:

The trained model is evaluated on the testing set. Metrics such as accuracy and a detailed classification report (including precision, recall, and F1-score for both classes) are generated to assess the model's performance.



```
Training model...
Evaluating model...
Accuracy: 79.07%
Classification Report:
      precision    recall  f1-score   support

Disengaged      0.80      0.85      0.83       100
Engaged         0.77      0.71      0.74        72

 accuracy: 0.79
macro avg: 0.79      0.78      0.78       172
weighted avg: 0.79      0.79      0.79       172
```

Figure 7. Trained Model

6. Model Saving:

The trained model is saved in a serialized format using joblib. This allows the model to be loaded and used later for real-time predictions in the browser-based application.

By following this process, the Random Forest Classifier achieves a reliable performance in distinguishing between engaged and disengaged cursor behaviors, enabling accurate predictions in the deployed application.

4.4. Real-Time Engagement Prediction

The engagement prediction system utilizes the trained Random Forest Classifier to classify user engagement based on real-time cursor movement data. The process involves the following steps:

1. Loading the Model:

The pre-trained cursor classifier model is loaded from its serialized form. This model was previously trained on features extracted from cursor movement data.

2. Session Data Input:

Real-time session data, recorded as a JSON file, is loaded. This file contains cursor movement information such as x and y coordinates, timestamps, and other interaction details.

3. Feature Extraction:

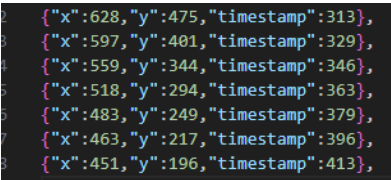
The session data is processed to calculate key metrics, including:

- **Total Distance:** The cumulative distance traveled by the cursor.
- **Average Speed:** The cursor's average movement speed during the session.
- **Speed Standard Deviation:** Variability in the cursor's speed.
- **Number of Movements:** The count of significant cursor movements exceeding a defined threshold.
- **Idle Time:** The total time the cursor remained stationary.
- **Direction Changes:** The number of significant directional shifts in cursor movement.

4. Prediction:

The extracted features are arranged in the order expected by the classifier. The model then predicts the engagement state of the user:

- **Engaged:** The user exhibits deliberate and consistent cursor movement patterns, such as reading or interacting with the content.



```
[{"x":628,"y":475,"timestamp":313},
{"x":597,"y":401,"timestamp":329},
{"x":559,"y":344,"timestamp":346},
{"x":518,"y":294,"timestamp":363},
{"x":483,"y":249,"timestamp":379},
{"x":463,"y":217,"timestamp":396},
{"x":451,"y":196,"timestamp":413},
```

Figure 8. Engaged Datapoints

```
PS D:\CursorClassifier> &
predict_engagement.py
Prediction: Engaged
```

Figure 9. Result

- **Disengaged:** The cursor movement is erratic, excessively fast, or idle for extended periods, indicating a lack of engagement.

```
1 [{"x":605,"y":471,"timestamp":200},
2 {"x":501,"y":303,"timestamp":300},
3 {"x":526,"y":246,"timestamp":406},
4 {"x":494,"y":261,"timestamp":503},
5 {"x":600,"y":180,"timestamp":600},
6 {"x":527,"y":297,"timestamp":700},
7 {"x":648,"y":242,"timestamp":800},
```

Figure 10. Disengaged Erratic Datapoints

```
PS D:\CursorClassifier> & C
predict_engagement.py
Prediction: Disengaged
```

Figure 11. Result

```
2 {"x":625,"y":190,"timestamp":303},
3 {"x":625,"y":183,"timestamp":407},
4 {"x":625,"y":183,"timestamp":511},
5 {"x":625,"y":183,"timestamp":617},
6 {"x":625,"y":183,"timestamp":713},
7 {"x":625,"y":183,"timestamp":813},
```

Figure 12. Disengaged Idle Datapoints

```
PS D:\CursorClassifier> & C:
predict_engagement.py
Prediction: Disengaged
```

Figure 13. Result

5. **Result Output:** The prediction result is displayed, providing immediate feedback on the user's engagement state. This output can be used to adapt the content or interface dynamically based on the user's engagement level.

This system demonstrates how cursor behavior can be effectively leveraged to infer user engagement in real-time, making it a valuable tool for applications such as e-learning, usability testing, and interactive content delivery.

5. Challenges

5.1. Data Privacy

Collecting cursor data raises privacy concerns, especially in educational settings. Measures will be taken to anonymize data and ensure compliance with relevant privacy regulations.

5.2. Data Quality

The quality and consistency of cursor data may vary depending on the user's behavior and device. Handling noisy or missing data will be a challenge.

5.3. Model Accuracy

While the Random Forest algorithm is robust, ensuring high accuracy across diverse student populations and various content types will require extensive testing and fine-tuning.

6. Conclusion

The development and testing of the HTML-based cursor tracker for assessing student engagement have demonstrated significant potential to revolutionize remote learning environments. The system's high accuracy, detailed through precision, recall, and F1 scores, underscores its capability to discern between engaged and disengaged states effectively. With real-time feedback and non-intrusive monitoring, educators can adapt teaching strategies dynamically, thus fostering an interactive and personalized learning experience. As online education continues to evolve, this tool offers a promising avenue for enhancing student engagement and, consequently, educational outcomes. Future work will focus on refining the model's predictive capabilities and expanding its application across diverse learning platforms.