**Title: Classification of User Desktop Activities using Eye-Movement Data and Supervised Machine Learning**

Author: Alireza Feyzabadi Farahani

Supervisor: Dr. Shiva Kamkar

**Abstract**

This research project investigates the efficacy of supervised machine learning techniques for classifying a user's desktop activity based on eye-tracking data. The primary objective was to develop a robust predictive model using a public dataset of gaze coordinates recorded during eight distinct tasks (e.g., *READING*, *BROWSING*, *DEBUGGING*). A comprehensive data processing pipeline was established, beginning with the consolidation and cleaning of raw time-series data. A sliding-window approach was employed for feature engineering, transforming gaze sequences into a structured feature set by computing statistical metrics such as the mean, standard deviation, and maximum of gaze coordinates and velocity.

A **Random Forest** classifier was selected as the baseline model, achieving an initial accuracy of **62.97%**. To systematically optimize performance, **GridSearchCV** with 5-fold cross-validation was implemented, improving the model's accuracy on the unseen test set to **63.60%**. Furthermore, an experiment to incorporate more complex, event-based features derived from fixations and saccades was conducted. Contrary to the initial hypothesis, this did not yield further improvement, suggesting that macro-level gaze statistics were more discriminative for this particular task set. Comparative analyses with an **XGBoost** classifier and the **SMOTE** technique for class balancing also did not surpass the tuned Random Forest's performance. The findings confirm the viability of using statistical gaze features for activity recognition but highlight the critical importance of feature selection and the task-dependent nature of feature efficacy.

**1. Introduction**

**1.1. Background and Motivation**

In the field of Human-Computer Interaction (HCI), the development of intelligent systems that can understand and adapt to a user's context is of paramount importance. Eye-tracking technology offers a powerful, non-intrusive window into a user's cognitive processes, providing high-fidelity data on their focus of attention. The analysis of gaze patterns—where a user looks, for how long, and in what sequence—can be used to infer their current task, intent, or cognitive state (e.g., concentration, confusion, or fatigue).

The ability to automatically recognize a user's desktop activity has significant practical applications. It can enable the creation of adaptive user interfaces that modify layouts or present relevant information based on the task at hand. In high-stakes environments, such as control rooms or medical diagnostics, it can be used to monitor operator workload and prevent errors. Furthermore, it holds potential for developing next-generation assistive technologies that provide proactive support to users.

**1.2. Problem Statement**

While the potential of eye-tracking is clear, translating raw gaze data into a high-level understanding of user activity remains a complex challenge. The data is a high-velocity time series of coordinates, and the patterns that define a specific activity are often subtle. This project addresses this challenge by framing activity recognition as a supervised machine learning classification problem.

**1.3. Project Objectives**

The primary objectives of this project are as follows:

1. To implement a complete data processing pipeline for a public eye-tracking dataset.
2. To develop and train a baseline machine learning model (Random Forest) and evaluate its performance.
3. To systematically optimize the baseline model through hyperparameter tuning.
4. To investigate the impact of more advanced, event-based features (fixations and saccades) on model performance.
5. To compare the optimized model against an alternative advanced model (XGBoost) and investigate techniques for handling class imbalance (SMOTE).
6. To analyze the results, identify the most predictive features, and discuss the model's strengths and limitations.

**2. Methodology**

**2.1. Dataset Description**

The study utilized the "Eye Movement Data Set for Desktop Activities" from Kaggle. This dataset contains recordings from multiple participants performing eight common desktop tasks. The data was provided as 192 individual .csv files, each corresponding to a unique user-task session. The raw data primarily consisted of timestamped x and y gaze coordinates.

**2.2. Data Preprocessing and Velocity Calculation**

The initial step involved programmatically loading all 192 CSV files and consolidating them into a single DataFrame. During this phase, metadata such as the user ID and task label were extracted from each filename. A unique session\_id was generated for each file.

To capture the dynamics of eye movement, **gaze velocity** was computed as the Euclidean distance between consecutive gaze points. This feature is critical as it distinguishes between a static fixation and a rapid saccade. The calculation was grouped by individual user sessions. To handle noise, the velocity was clipped at the 99.9th percentile.

**2.3. Feature Engineering**

As traditional machine learning classifiers require a fixed-size feature vector, the time-series gaze data was transformed using a **sliding-window feature extraction** method.

* **Window Size:** 50 samples.
* **Step Size:** 25 samples (50% overlap).
* **Extracted Features:** For each window, 12 statistical descriptors were calculated:
  + Mean, standard deviation, minimum, and maximum of x and y coordinates.
  + Mean, standard deviation, and maximum of velocity.
  + A count of samples within the window.

This process converted the raw time series into a structured dataset of 59,954 instances.

**2.4. Model Selection and Training**

The **Random Forest** algorithm was selected as the primary classification model due to its high accuracy, robustness against overfitting, and its ability to calculate feature importance. For comparison, **XGBoost**, an implementation of gradient boosting, was also selected due to its reputation for high performance on tabular data.

**2.5. Experimental Setup and Evaluation**

The feature set was partitioned into an 80% training set and a 20% testing set using **stratification** to maintain class proportions. Model performance was assessed using **Accuracy**, **Precision**, **Recall**, **F1-Score**, and a **normalized confusion matrix**.

**3. Results and Discussion**

**3.1. Baseline Model Performance**

The initial Random Forest model, trained with default parameters, established a strong performance baseline. On the unseen test data, it achieved an overall **accuracy of 62.97%**. The model performed exceptionally well on the READ task (91% recall) but struggled with tasks like SEARCH (46% recall).

**3.2. Hyperparameter Tuning for Optimization**

To systematically improve upon the baseline, a **Grid Search with 5-fold Cross-Validation (GridSearchCV)** was performed. This technique exhaustively searches a predefined parameter grid to identify the optimal hyperparameter combination.

The tuning process yielded the following optimal parameters:

* **Best Parameters:** {'n\_estimators': 300, 'max\_depth': None, 'min\_samples\_leaf': 1}.

This configuration suggests the model benefits from a larger number of deep, unpruned trees. When the model with these optimized parameters was evaluated on the test set, it achieved an improved **accuracy of 63.60%**. This result demonstrates the value of systematic tuning.

**3.3. Experiment on Advanced Feature Engineering**

Following the initial modeling, an experiment was conducted to determine if more granular, event-based eye-tracking features could improve classification accuracy. An **I-DT (Identification by Dispersion-Threshold)** algorithm was implemented to detect **fixations** and **saccades**. From these identified events, a new set of theoretically more powerful features was extracted, including mean fixation duration, saccade amplitude, and scan-path length.

These advanced features were added to the original feature set and used to train the optimized Random Forest model. **Contrary to the initial hypothesis, this did not result in a significant improvement in model accuracy.** This important negative result suggests several possibilities:

1. **Parameter Sensitivity of I-DT:** The dispersion and duration thresholds of the I-DT algorithm were not tuned, which may have led to inaccurate event detection, introducing noise that offset any informational gain.
2. **Feature Redundancy:** The new features may be highly correlated with the original statistical features. For example, a low standard deviation of coordinates (y\_std) already implies long fixations in a small area. The new features may not have provided enough novel, uncorrelated information for the model to leverage.
3. **Dominance of Macro-Patterns:** For this specific set of tasks, the **macro-level** gaze patterns (i.e., the general location and spread of gaze, captured by the baseline statistics) appear to be more discriminative than the **micro-level** dynamics of individual fixations and saccades. The key difference between WATCHING a video (gaze in a central box) and WRITING code (gaze scanning vertically) is better captured by coordinate variance than by fixation duration.

This finding suggests that for this problem, the added complexity of event-based feature extraction does not justify its computational cost.

**3.4. Comparative Analysis: SMOTE and XGBoost**

Further experiments were conducted to explore other avenues for improvement:

* **Class Imbalance (SMOTE):** The **SMOTE** technique was applied to the training data to correct for class imbalance. This did not improve the Random Forest's performance, resulting in a slightly lower accuracy of 62.21%.
* **XGBoost Model:** The XGBoost classifier achieved an accuracy of 60.56% on the original data and 60.15% on the SMOTE data. In this application, it did not outperform the optimized Random Forest.

**3.5. Final Model Analysis**

The tuned Random Forest model (63.60% accuracy) remains the best-performing model. Its confusion matrix confirms that the primary challenge is distinguishing between tasks with similar screen-scanning behaviours like SEARCH, BROWSE, and DEBUG. The feature importance analysis consistently showed that **velocity-based metrics** were the most decisive, highlighting that the dynamics of eye movement are more informative than static gaze location.

**4. Conclusion and Future Work**

This project successfully developed and optimized a machine learning pipeline to classify user desktop activities, achieving a final accuracy of **63.60%** with a tuned Random Forest classifier. The study confirms that statistical features derived from a sliding window can effectively capture activity-specific gaze patterns.

A key finding was that simple statistical features describing the macro-level distribution and dynamics of gaze were more effective than a more complex, event-based feature set. This highlights that feature efficacy is highly task-dependent and that increased complexity does not guarantee improved performance.

Based on these findings, several directions for **future work** are recommended:

* **Advanced Sequential Modeling:** Employ deep learning models designed for time-series data, such as **Long Short-Term Memory (LSTM)** networks. These models could learn temporal dependencies directly from the raw gaze data, potentially capturing subtle patterns that our current feature engineering approach misses.
* **Systematic I-DT Parameter Tuning:** A more rigorous investigation into event-based features should include a systematic tuning of the I-DT algorithm's dispersion and duration parameters to ensure the most accurate possible detection of fixations and saccades.
* **Larger and More Diverse Datasets:** Building a more generalized and robust model would require expanding the dataset to include more participants, a wider variety of tasks, and different screen resolutions and layouts.