Real-Time Inference (RTI) Simulator Software: Fatigue Detection Application

Kiman B. Park, Ph.D.

Overview: My RTI-Simulator Software is a sophisticated tool designed for detecting and analyzing fatigue using the Neurobiomonitor headset (NBM) by Freer Logic. This software leverages machine learning models to predict the Karolinska Sleepiness Scale (KSS) based on real-time data inputs.

Features:

1. **Dataset Integration:**
    - My software comes pre-loaded with a dataset provided by Eden Band Company.
    - Users can select specific subject IDs to visualize individual results.
2. **Machine Learning Models:**
    - **Random Forest Classifier:** This primary model is used for predicting KSS.
    - **K-Nearest Neighbor Classifier (KNN):** Although included, this model is not active in the current version due to its lower accuracy compared to the Random Forest Classifier.
3. **Real-Time Inference Simulation:**
    - A dynamic simulation feature displays real-time inferences in a dedicated blue window.
    - As data is captured, real-time KSS inferences are marked with green indicators.
    - If no signal is detected, the KSS is automatically set to 0.
    - An alert system is triggered if the KSS exceeds a threshold of 7, with both visual and auditory notifications. This threshold is user-adjustable.
    - The code can be transferred and integrated easily to other sensors for real-time inference.
4. **Customizable Interface:**
    - Users can personalize the application's appearance through the main menu option “Work Space” -> “Change the style”.
5. **Model Parameter Viewing:**
    - Detailed model parameters can be accessed via the menu by selecting “Parameters” and choosing either “1st Model” or “2nd Model”.
6. **Logging System:**
    - A comprehensive logging system is integrated, accessible through the menu for review.

Technical Specifications:

1. **Operating System:** Windows 10
2. **Programming Language:** Python 3.11

3. **Integration:** Neurobiomonitor headset (NBM) by Freer Logic
4. **Dataset:** Provided by Eden Band Company
5. **Real-Time Data Processing:** Visual and auditory alert systems based on threshold settings

User Manual:

- The operating manual is currently under development and will be included in future releases.

Conclusion: I designed the RTI-Simulator Software to provide an advanced and user-friendly solution for fatigue detection and analysis, featuring robust machine learning capabilities and real-time data processing. The application's customizable interface and adjustable parameters make it suitable for a wide range of users, from researchers to practitioners in various fields.

Future Direction: The future development of my RTI-Simulator Software will aim to enhance its accuracy, integrate various operating systems and mobile systems, expand its features, and improve user experience. These advancements will ensure the software remains at the forefront of fatigue detection technology, catering to the evolving needs of its users.

1. **Model Accuracy Improvement:**
    - Data Augmentation: Incorporate additional data from diverse subjects to train the models, improving their generalization and accuracy.
    - Feature Engineering: Enhance the feature extraction process to identify more relevant signals from the NBM headset data.
    - Advanced Algorithms: Explore and integrate more advanced machine learning models such as Deep Neural Networks (DNNs) and ensemble methods to surpass the current performance of the Random Forest Classifier.
2. **Cross-Platform Integration:**
    - Support of Multiple Platforms: Integrate the software compatibility in Mac OS, Linux, and mobile platforms to increase the variety of preference.
3. **Enhanced Real-Time Inference:**
    - Adaptive Thresholds: Develop algorithms to dynamically adjust thresholds based on individual user profiles and historical data.
    - Continuous Learning: Integrate mechanisms for the models to learn and adapt from new incoming data, improving over time with more usage.
4. **Expanded Feature Set:**
    - Additional Sensors: Support additional biosensors to capture more comprehensive physiological data, providing a holistic view of the user's fatigue state.

- User Feedback Integration: Implement a feedback system where users can manually input their perceived fatigue levels, allowing the software to validate and adjust its predictions.
- Predictive Analytics: Add predictive analytics features to forecast future fatigue levels based on current trends and historical data.

5. **User Experience Enhancements:**
    - Improved Interface: Continuously refine the user interface for better usability and aesthetics, ensuring a smooth user experience.
    - Mobile Integration: Develop mobile applications to allow users to monitor their fatigue levels on-the-go.
        - This software can be easily integrated into the mobile application using **Kivy** or **BeeWare** packages in Python.
    - Custom Alerts: Provide more customization options for alert notifications, including different alert tones, visual cues, and haptic feedback for wearable devices.
6. **Comprehensive Reporting and Analytics:**
    - Detailed Reports: Generate detailed fatigue analysis reports that can be exported and shared, including trends, predictions, and personalized recommendations.
    - Dashboards: Create interactive dashboards that visualize data trends, model performance, and user statistics in an intuitive format.
7. **Collaboration and Integration:**
    - API Development: Develop APIs to allow integration with other health and wellness applications, enabling a more comprehensive health monitoring ecosystem.
    - Research Collaboration: Collaborate with research institutions to validate the software's effectiveness and contribute to ongoing studies in fatigue and sleep research.