Systems Analysis & Design
Semester 2025-I
Course Project Description

Eng. Carlos Andrés Sierra, M.Sc.
Computer Engineering
Universidad Distrital Francisco José de Caldas

Overview

This course project integrates the primary lessons covered in previous workshops:

i. **Workshop #1: Analysis of a Kaggle Competition**.

ii. **Workshop #2: System Design and Architecture**.

iii. **Workshop #3: Computational Simulation and Validation**.

In this final phase, you will incorporate a **Large Language Model (LLM)** available on Kaggle to:

- Automatically generate or refine submissions for the competition you analyzed.
- Evaluate how these LLM-based solutions perform in real competition settings.
- Document your findings in a final report that references insights from prior workshops.

Project Scope and Objectives

- **Integrate an LLM:** Choose or fine-tune an available model (e.g., a specialized notebook or Kaggle kernel) that can assist with data preprocessing, feature engineering, or direct predictions.
- **End-to-End Pipeline:** Combine your analysis (Workshop #1), design decisions (Workshop #2), and simulation approach (Workshop #3) into a coherent workflow.

Carlos Andrés Sierra, Computer Engineer, M.Sc. in Computer Engineering, Titular Professor at Universidad Distrital Francisco José de Caldas.

Any comment or concern about this document can be sent to Carlos A. Sierra at: *cavirguezs@udistrital.edu.co*.

- **Competition Submission:** Prepare at least one valid submission to the chosen Kaggle competition using the LLM-based approach.
- **Performance Evaluation:** Compare the LLM-driven submission to any baseline or traditional models used in previous workshops. Highlight improvements or drawbacks.
- **Project Report:** Summarize the entire process, referencing all prior workshop results and detailing the LLM application, Kaggle leaderboard insights, and recommended next steps.

Project Steps

1. **LLM Setup:**
   - Identify a pre-trained model in Kaggle or upload your own model version.
   - Justify the choice of model (e.g., domain relevance, multilingual capabilities, or known successes in text-based feature generation).
2. **Connector or API:**
   - Implement a straightforward *interface* or script to link your system design (Workshop #2) with the chosen LLM.
   - Ensure that data can be seamlessly fed into the LLM for inference, transformation, or other manipulations (e.g., generating answers, rewriting code blocks).
3. **Simulation and Testing:**
   - Reuse or extend the simulation strategies from Workshop #3 to evaluate how the LLM interacts with your system architecture.
   - Track performance (time, resource usage) and system behavior under varied scenarios (e.g., small data vs. large data).
   - Confirm that the system remains robust regarding chaotic inputs or unexpected user prompts.
4. **Competition Submission:**
   - Generate predictions or post-processed solutions using the LLM, then submit results to the Kaggle competition.
   - Document the submission process, including data preparation, any hyperparameter tuning, and error handling.
5. **Performance and Discussion:**
   - Compare the LLM-based submission's score to previous approaches (traditional ML or deep learning methods).

- Explain any disparities between expected performance and actual leaderboard results.
- Propose potential next steps or model improvements.

6. **Final Report:**
   - Combine your writing from all workshops (analysis, design, simulation, LLM approach).
   - Include an **experiments section** detailing personal attempts to improve LLM outputs, data transformations, and relevant Kaggle codes.
   - Provide final remarks on the project's overall efficiency, user experience, and future research or development ideas.
   - Save your final PDF and relevant source codes in a dedicated folder (e.g., `Final_Course_Project`) in your GitHub repository.

Deliverables and Deadlines

- **LLM Implementation Notebook or Script:** Show how the model was integrated (upload to Kaggle or store in your repository).
- **Final Report (PDF)**: Summarize the entire project, highlighting LLM design choices, Kaggle outcomes, and any code references.
- **Supporting Files:** Any Dockerfiles, `requirements.txt`, or environment.yml definitions for reproducibility.
- **Submission Deadline:** Refer to your course schedule or official announcements for the exact date and time.

Notes and Recommendations

- Coordinate with your team to ensure consistent integration of analysis, design, and simulation artifacts from prior workshops.
- Document any challenges faced when combining the LLM with your pipeline (e.g., large memory usage, slow inference times).
- If the competition results do not meet your expectations, discuss possible reasons and propose specific improvements (e.g., advanced prompting, more training epochs, or better structured data inputs).

This concludes your course project guidelines. By uniting analysis, design, simulation, and an LLM-based Kaggle submission, you will gain a rich, hands-on experience in modern systems engineering. Good luck!