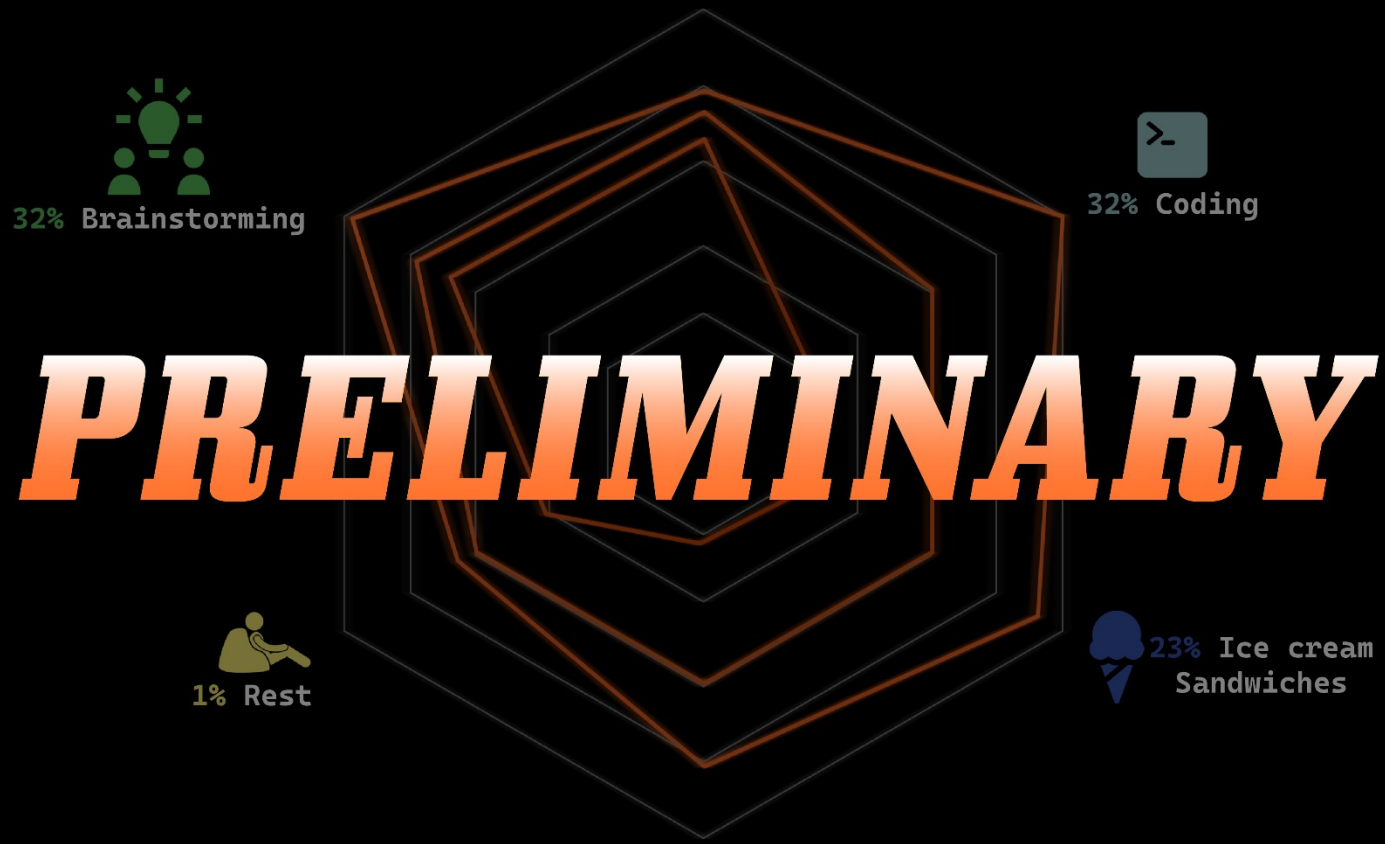
BITFEST 2025

HACKATHON



PRELIMINARY

প্রভু! আমায় জ্ঞান দাও

কুয়েট প্রকৌশল ও প্রযুক্তি বিশ্ববিদ্যালয়

Challenge 1: Banglish-to-Bengali Transliteration

Iqbal's laptop has been hacked by an evil hacker who uninstalled the Avro keyboard, leaving him unable to type in Bengali. To make matters worse, he can't reinstall the keyboard, and he desperately needs to write in Bengali to win a heated Facebook comment debate. Determined to overcome this challenge, Iqbal decides to train a model that can convert his Banglish text (Bengali written in English letters) into proper Bengali script.

Dataset - SKNahin/bengali-transliteration-data · Datasets at Hugging Face

Tasks

1. **Load the Dataset**

Use the provided Hugging Face dataset link to load the data.
Split the dataset into training and validation subsets (e.g., 80/20 or 90/10 split).

2. **Data Preprocessing**

Tokenize both Banglish and Bangla text.
Use appropriate tokenization techniques for sequence-to-sequence tasks.
Clean or filter the dataset (e.g., remove overly short or excessively long sentences if needed).

3. **Select a Model**

Choose a pre-trained sequence-to-sequence model from Hugging Face's Transformers library. Examples include mBART, T5, or any relevant translation model. Justify your choice of model based on performance, suitability for low-resource language tasks, and efficiency.

4. **Train the Model**

Fine-tune your selected model on the provided Banglish-to-Bangla dataset.
Set up a clear training pipeline using any deep learning framework (e.g., PyTorch, TensorFlow).
Use appropriate hyperparameters (e.g., learning rate, batch size, number of epochs) and justify your choices.

Challenge 2: Mofa's Kitchen Buddy

Mofa loves cooking and frequently collects recipe pictures or posts from social media. He saves his favorite recipes in a dedicated folder. Mofa wants to build a backend system powered by a Large Language Model (LLM) that helps him manage his ingredients and suggests recipes based on what he has at home. He plans to input available items in his house and update them when shopping. Additionally, he envisions a chatbot that can interact with him to suggest recipes tailored to his preferences, such as craving something sweet or a specific type of dish.

Tasks

1. **Database Design:**
    - Create a database schema to store available ingredients.
2. **Ingredient Management API:**
    - Develop APIs to input available items and update them after shopping or cooking.
3. **Recipe Retrieval:**
    - Build a system to parse and store recipe details from the already saved recipe images or texts into a combined *my_fav_recipes.txt* file.
    - Develop APIs to input new favorite recipe image or text.
4. **Chatbot Integration:**
    - Integrate an LLM-based chatbot that interacts with the user to understand preferences (e.g., "I want something sweet today").
    - The chatbot should process a combined recipe text file and recommend recipes based on the ingredients available at home.

Tips

Start by implementing text-based recipe retrieval first. If you have extra time, add OCR functionality to accept images as input too!

General Instructions

Participants are required to create a new GitHub repository at the start of the Preliminary Round. To solve these problems, feel free to use any stack or tools you like!

If you can't solve all the tasks, no worries—just submit whatever you've got!

Additional Considerations

- The combined *my_fav_recipes.txt* file can be very large.
- Feel free to use any LLM model you like for chatbot integration—local or API, free or paid!
- Focus on user-friendly API design for easy integration with other services. No need for UI.
- Maintain standard practices while preserving folder structure and file naming conventions.
- ***ELEGANCY IS THE KEY***.
- Emphasize team collaboration and time management for optimal performance.

Deliverables

- Source code repository containing -
    - **Challenge 1 Folder**
        - The Jupyter Notebook
    - **Challenge 2 Folder**
        - Solutions of the tasks
    - **README.md** containing concise API documentation
- Hosted model
    - Share the link to your trained model hosted on Hugging Face. If you can't upload it to Hugging Face, no problem—you can use Git LFS to share it instead!

README Format

- For API Documentation follow this example –

````
- Route: /notices
Method: GET
Sample Response:
```json
[
  {
    "id": 1,
    "date": "2024-12-23",
    "day": "Monday",
    "time": "8:30 AM",
    "notice": "Assignment Submission"
  }
]
```
- Route: /add-ct-announcement
Method: POST
Sample Payload:
```json
{
  "date": "2024-12-23",
  "day": "Monday",
  "time": "10:00 AM",
  "place": "Room 102",
  "course_id": "CSE3217",
  "topic": "Mobile Computing",
  "teacher_id": "1"
}
```
````