Gebze Technical University
Department of Computer Engineering
CSE 654 / 484
Fall 2024

Homework 2
Due date: Jan 2 2024

Turkish Question Answering System for Gebze Technical University Official Student Rules

Project Overview

This project aims to build a Question Answering (QA) system that can extract relevant answers from Gebze Technical University's official student rules document, as provided on the university's website (https://www.gtu.edu.tr/icerik/1479/592/lisans-yonetmelik-ve-yonergeler.aspx). Students will work on implementing a natural language understanding system specifically tailored for university rules and regulations in Turkish.

The project will involve preprocessing official documents, training or fine-tuning a Turkish-language transformer-based model, and deploying a system capable of answering natural language questions about the rules.

Project Objectives

1. Parse and preprocess the Gebze Technical University student rules document to create a QA dataset.
2. Build a QA system to answer Turkish questions about university regulations.
3. Fine-tune a pre-trained Turkish transformer model for extracting precise answers.
4. Create a user-friendly interface where students can ask questions and receive answers

directly from the rules.

Dataset and Preparation

Step 1: Collect Data
- Download the **Student Rules and Regulations Document** from Gebze Technical University's official website:
  - [Link to GTU Student Rules and Regulations](https://www.gtu.edu.tr/icerik/1479/592/lisans-yonetmelik-ve-yonergeler.aspx)
- Extract the text using libraries like `PyPDF2` or `BeautifulSoup`.

Step 2: Annotate Data
- Convert the extracted content into a structured QA dataset with context, question, and answer fields.

System Design

Step 1: Model Selection
- Use a pre-trained Turkish transformer model such as BERTurk or XLM-RoBERTa.

Step 2: Fine-Tuning
- Fine-tune the chosen model on the annotated dataset with span-based loss.

Step 3: Evaluation
- Evaluate the model using Exact Match (EM) and F1 Score metrics.

Features

Core Features
1. Answer Extraction: Provide precise answers to natural language questions about the student rules.
2. Searchability: Extract answers from multiple sentences or sections.

Optional Features
1. Rule Summarization: Summarize sections of the rules document.
2. Interactive Interface: Web-based interface for user interaction.

Implementation Steps

Step 1: Preprocessing
- Tokenize and clean the text, considering Turkish characters and casing.

Step 2: Fine-Tune Transformer
- Use Hugging Face Transformers library to fine-tune a pre-trained model.

Step 3: Build the Interface
- Implement a preferably web-based QA interface using Flask or FastAPI.

Suggested Timeline

Week 1

- **Day 1-2:** Extract and preprocess the GTU student rules document.
- **Day 3-4:** Annotate the dataset with questions and answers.
- **Day 5-7:** Fine-tune a pre-trained Turkish transformer model on the annotated data.

Week 2

- **Day 8-9:** Build the QA system back-end using Python and Hugging Face Transformers.
- **Day 10-12:** Develop a front-end for user interaction.
- **Day 13-14:** Test the system, evaluate its performance, and refine based on feedback.

Deliverables

1. **Annotated Dataset**: GTU-specific QA dataset with questions and answers.
2. **Source Code**: Fully documented code for the QA system.
3. **Interactive Demo**: A functional web-based QA interface.
4. **Final Report**: Description of methodology, challenges, and evaluation results.

Recommended Tools

1. **Libraries**:
   - Transformers by Hugging Face for model implementation.
   - NLTK or Zemberek for Turkish text preprocessing.
   - Flask or FastAPI for API development.
2. **Datasets**:
   - GTU rules document (annotated).
   - TurQuAD or Boun-QA for supplementary training data.
3. **Platform**:
   - Google Colab or your resources.

Submission and Rules:

Prepare your report in PDF format and submit it along with your code (either as a ZIP file or a notebook) via the submission platform.

You may use any programming language for the implementation.

We encourage the use of tools like ChatGPT for assistance, but your homework should reflect more original work than automated parts.

Remember: No report, no points! Similarly, no code, no points!