MACHINE LEARNING PROJECT DOCUMENTATION

20XW58 – MACHINE LEARNING LAB

TOPIC: HATE SPAN DETECTION with crf-based nlp

**INTRODUCTION:**

The hate span detection project aims to develop an automated system for identifying and predicting the BIO tags of hate speech in text data using Conditional Random Fields (CRF) and Natural Language Processing (NLP) techniques.

Hate speech poses a significant social issue in the online world, contributing to harassment, discrimination, and hostility.

Its primary objective is to create a model that can accurately detect hate spans in text, ultimately aiding in the moderation of online content.

**PROBLEM STATEMENT:**

Given a training dataset of text with hate speech spans annotated in BIO format, develop a machine learning model that can identify the hate span and returns the BIO tag.

**PURPOSE:**

The main purpose is to develop a machine learning model to detect hate spans (text fragments that express hatred or bigotry towards particular group or individual) in text.

**SCOPE:**

The model will be trained on a dataset of annotated text samples that contains hate speech

We have also developed a web service to make the model available to users.

**WHY IS THIS PROJECT?**

This project is chosen to address the problem of online hate speech.

By developing a machine learning model to detect hate spans, it helps to make social media, internet a safer and more inclusive place for everyone.

**DATASET:**

Kaggle dataset – hatenorm23

<https://www.kaggle.com/competitions/hatenorm23/data>

**CODE STRUCTURE:**

1. Importing libraries and dependencies
2. Data preparation (loading and preprocessing)
3. Feature Engineering using CRF for NLP tasks.
4. Model training and evaluation.
5. Web application development with Flask
6. Model deployment to a Flask API.

**DEPENDENCIES:**

The project relies on the following Python libraries and tools:

* ***google.colab*** for Google Co-laboratory integration.
* ***pandas*** for data handling and manipulation
* ***sklearn\_crfsuite*** for CRF based machine learning
* ***joblib*** for model persistence
* ***Flask*** for web application development
* Other standard Python libraries for data science and web development.

**DATA PREPROCESSING:**

* *Data loading:* Reads data from a CSV file using pandas’ library.
* *Data cleaning:* Duplicates and missing values are removed from the dataset.
* *Text normalization:* Text is converted to lowercase for uniformity.

**MODEL TRAINING AND EVALUATION:**

* *Data splitting:* Dataset is split into training and evaluation sets.
* *CRF Model Training:* A CRF model is trained using the training data.
* *Model Evaluation:* The model performance is evaluated using various classification metrics using F1-Score, Recall, Precision, Accuracy score

**MODEL DEPLOYMENT:**

* Trained CRF model saved using joblib library.
* The model is loaded in the Flask application for real time predictions.

**USAGE:**

1. Run the flask application by executing the Python script in google co-laboratory.
2. Access the application through a web browser (running in local host) to enter text or hate speech and detect hate spans using BIO tags.

**CONCLUSION:**

This project addresses the critical issue of hate speech in online platforms by developing a machine learning model for accurate detection.

The project code, along with model deployment, provides a practical tool to enhance online content moderation.

**FUTURE IMPROVEMENTS:**

* Model refinement to achieve improved accuracy
* Enhanced UI/UX
* Integration with various online platforms for real time moderation

**CONTRIBUTORS:**

21PW06 – HARITHRA R

21PW08 – JENISA MERLIN D

**PROJECT REPOSITORY:**

The code and resources for this Hate Span detection project can be found in the HateHurter-ExposingToxicTexts repository on GitHub.

**Repository: HateHurter-ExposingToxicTexts**

[**https://github.com/Jenisa-Merlin/HateHurter-ExposingToxicTexts.git**](https://github.com/Jenisa-Merlin/HateHurter-ExposingToxicTexts.git)

Explore the code, data, and other project materials available in the repository.