CS626 - Speech, Natural Language Processing, and the Web

Named Entity Identification

Group Id- 53
Shruti Patel, 24M0825 , CSE
Kaustubh Shivshankar Shejole, 24M2109, CSE
Tushar Katakiya, 24M2110, CSE
Kush Mangukiya, 24M0769, CSE

Date: 4th November 2024

Problem Statement

- Perform Named-Entity Identification using SVM classifier with appropriate feature engineering

- **Technique to be used**: SVM classifier

- **Dataset**: CoNLL-2003 NER Data; https://paperswithcode.com/dataset/conll-2003 and https://huggingface.co/datasets/conll2003

(they are same data, but have common and distinct information)

Problem Statement

- **Input**: A sentence
- **Output**: Name-No Name tagged for each word in the sentence

- **Example**:
  - **Input**: Washington DC is the capital of United States of America
  - **Output**: Washington_1 DC_1 is_0 the_0 capital_0 of_0 United_1 States_1 of_1 America_1

Data Processing Info (Pre-processing)

1. **Tokenization:** Used pre-tokenized format from the CoNLL-2003 dataset; each token represents a word or punctuation.
2. **Binary NER Labeling:** Converted entity tags (`PER`, `ORG`, etc.) into binary labels:
   a. 1 for named entities.
   b. 0 for non-entities.
3. **Case Preservation:** Retained original casing to capture capitalization patterns, used lowercase only in certain features.
4. **POS Tag Mapping:** Converted POS tags into unique numerical identifiers for binary feature representation.
5. **Removing Possessive Suffixes:** Used `get_root_word` function to remove possessive endings (`'s` or `'`) for standardization.
   a. No further tokenization or stop-word removal was applied due to the dataset's pre-tokenized format.

Feature Engineering (1/2)

- **Currency features:**

  is_currency: Checks if the term refers to a currency.

- **Capitalization Features:**

  is_title: Checks if the word is in title case (e.g., names of people or places are often capitalized).

  is_upper: Indicates if the word is in uppercase (commonly used for acronyms or special terms).

  is_lower: Identifies lowercase words, helping differentiate ordinary words from proper nouns.

- **Word Lowercase Representation:** The lowercase form of each word (word.lower()) helps standardize text, reducing variability caused by case differences.

Feature Engineering (2/2)

- **Date and time checking:** *We check whether a word refers to date or time.*
- **Character-Based Features:**

  suffix-1: Captures the last character of the word, which can indicate specific categories (e.g., "s" for plurals or "y" for some names).

  prefix-1: Captures the first character, useful for identifying patterns related to specific entity types.
- **POS Tagging:**

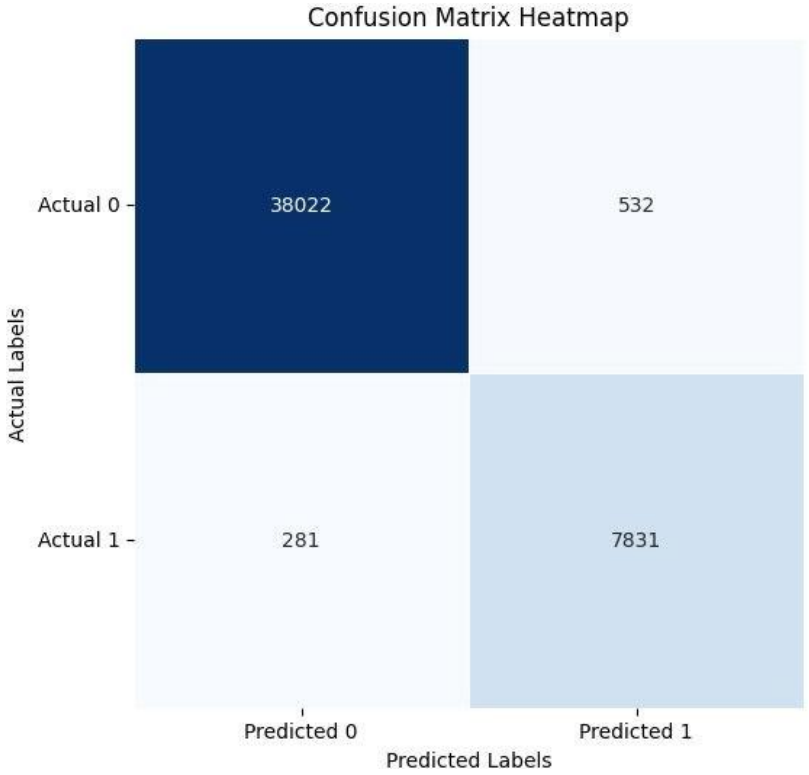
  pos: Part-of-speech (POS) tag of the word, which provides context about its grammatical role, aiding in distinguishing entities from non-entities.
- **Root Word Analysis**:

  root_word: Provides the base form of the word, useful for reducing variations (e.g., "running" → "run").

SVM Implementation

- Utilized scikit-learn's SVM implementation for simplicity and performance.
- Preprocessed text data to generate features for each token, such as word embeddings, part-of-speech tags, and contextual cues.
- Trained the SVM model on CONLL datasets, with features engineered to enhance recognition accuracy.
- Used libraries like NLTK for tokenization, POS tagging, and feature extraction.

Confusion Matrix



Overall performance

	precision	recall	f1-score	support
0	0.99	0.99	0.99	38554
1	0.94	0.97	0.95	8112
accuracy			0.98	46666

Error analysis

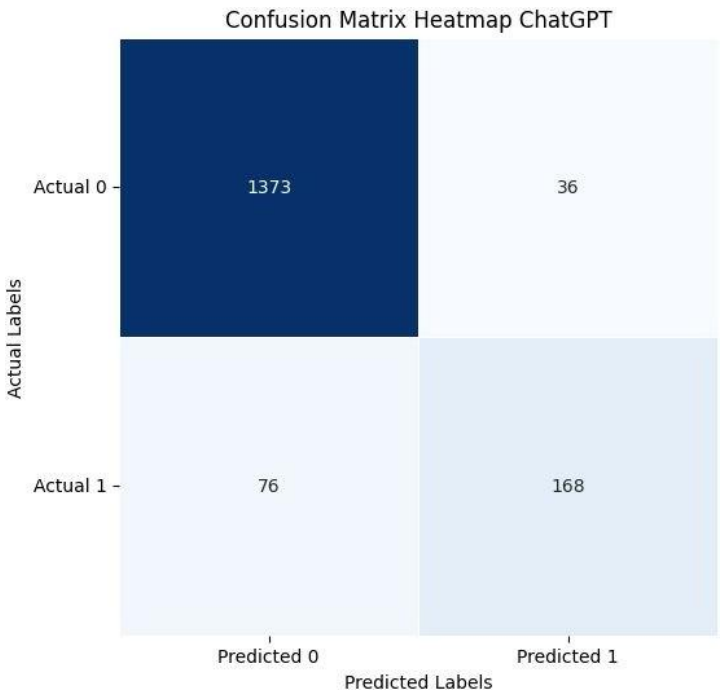
- **Ambiguous Names:** Words that can function as both named and non-named entities are challenging. For instance:

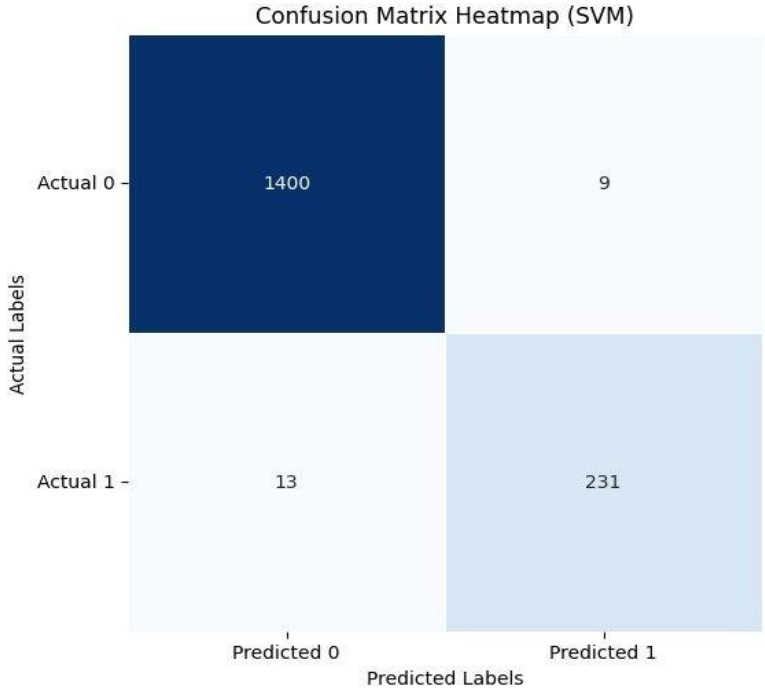
  "May": Interpreted as either a month (DATE) or a verb, depending on context. It was present as ‘O’ in the dataset.

  "Apple": Misclassified as a general noun instead of a named entity (ORG) when used without clear business context.

- **Lack of Capitalization:** Named entities are often uncapitalized in informal text, leading to confusion with non-named entities.
- **CAPITALIZATION:** Normal words capitalized give errors.

Comparison with ChatGPT





Learnings

- Understood the role of preprocessing techniques, such as lowercasing and lemmatization, to standardize text and reduce noise.
- Used NLTK for POS tagging and root word extraction, enhancing the model’s ability to recognize entities accurately.
- Gained hands-on experience with scikit-learn's SVM implementation, which proved effective for binary-class text classification.

Project

Aim: The aim of this project is use multilingual language models to classify Gujarati text into different sentiment categories, such as positive, negative, or neutral. The best performing model will be used for testing. This project will help in understanding the emotional tone of Gujarati text data, enabling applications in customer feedback analysis, social media monitoring, and other NLP-based tasks for Gujarati language.

Dataset : https://github.com/MG1800/gsac/tree/main (total 6575 tweets)

Input :

Gujarati text samples, such as user comments, reviews, or tweets.

e.g. *"આ સેવા ખરાબ છે" ("This service is bad").*

Output:

The sentiment label for the input text (e.g., **Positive**, **Negative**, **Neutral**).
e.g. for "આ સેવા ખરાબ છે", the model should output

"Negative". **Reference:** [GSAC: A Gujarati Sentiment Analysis

Corpus from

Twitter](https://aclanthology.org/2023.wassa-1.12) (Gokani & Mamidi, WASSA 2023)

Evaluation Scheme

- Demo working- 10/10 (if not working or no GUI - 0)
- SVM implementation and Feature Selection - 10/10
- Confusion matrix drawn and error analysed- 10/10
- Overall F1-score
  - > 90 - 10/10
  - >80 & <=90 - 8/10
  - >70 & <=80 - 7/10
  - so on.
- Comparison with ChatGPT (10)
- **Note: Must have GUI, otherwise no mark will be given for demo.**