IndiaAI CyberGuard AI Hackathon

Project Title: Crime Report Classification Using NLP

Team Name : Jay_Khodiyar

Institute Name : Monark University

Team Members:

1. **Amit Gurjar:**
    - **Role:** Team Leader
    - **Profession:** BE-IT Student
    - **Expertise:** AI/ML, Computer Vision
    - **Contact:** amitgurjar8155@gmail.com
2. **Ronak Gohil:**
    - **Role:** Team member
    - **Profession:** BE-IT Student
    - **Expertise:** AL/ML, Python
    - **Contact:** ronakgohil240@gmail.com
3. **Harshil Sangani:**
    - **Role:** Team member
    - **Profession:** BE-IT Student
    - **Expertise:** AL/ML, Data Visualization
    - **Contact:** harshilsangani07@gmail.com
4. **Yash Sapra:**
    - **Role:** Team member
    - **Profession:** BE-IT Student
    - **Expertise:** AL/ML, Python
    - **Contact:** yprajapati276@gmail.com

Date: 2024-11-21

1.Executive Summary

Project Overview

Our project tackles the challenge of subcategory prediction in a crime-related dataset for the India AI Cyberguard Hackathon. This dataset contains raw, unstructured text information and is classified into three primary fields: **Category**, **Subcategory**, and **Crime Additional Info**. The primary task is to build a robust NLP model that can accurately predict the subcategory based on the crime additional info, which often includes messy text descriptions.

Objective

The goal is to effectively classify these text entries into predefined subcategories to enable more streamlined and actionable insights into crime-related data. Accurate subcategory classification is vital for aligning outputs with Government of India regulations and provides a foundation for improved data analysis and reporting.

Approach

Our approach to this challenge includes experimenting with various machine learning models, with a primary focus on **GRU** due to its effectiveness in handling complex data. Additional models explored include traditional machine learning methods such as **Naive Bayes** and **SVM**, along with advanced techniques like **BERT** for capturing contextual information in crime-related descriptions.

To optimize model performance, we implemented a detailed pipeline that includes:

- **Text Preprocessing**: Handling noisy text data with typos, abbreviations, and inconsistent phrasing.
- **Feature Engineering**: Transforming text data into meaningful vectors, primarily using TF-IDF and embeddings.
- **Model Tuning**: Fine-tuning model parameters to maximize classification accuracy and balance across categories.

Results

After extensive model testing, GRU demonstrated a strong balance of accuracy and efficiency, with further refinements yielding improved precision and recall for most subcategories. Our experiments with XGBoost and other complex models also highlighted areas where deeper semantic understanding significantly boosted subcategory prediction, particularly in ambiguous cases.

2. Introduction

Theme and Context

This project is part of the India AI Cyberguard Hackathon, centered on developing an NLP model to assist citizens in filing cybercrime reports more accurately on the **National Cyber Crime Reporting Portal (NCRP)**. By enabling real-time analysis of text descriptions and incident-related media files uploaded by citizens, the project aims to guide users in correctly categorizing their reports, making the reporting process more efficient and less error-prone.

Problem Statement

One of the primary challenges in reporting cybercrimes is ensuring that users accurately describe incidents, including selecting the correct subcategories for streamlined investigation and action. Misclassification or incomplete information in reports can lead to inefficiencies and delays in response. This project addresses this challenge by focusing on **automatic classification of crime descriptions** into appropriate subcategories based on raw, unstructured text data.

Dataset Overview

The dataset provided contains approximately **1 lakh rows** of raw, unstructured text entries. Each row includes:

- **Category**: The main classification of the crime.
- **Subcategory**: More specific classifications within each category.
- **Crime Additional Info**: Descriptive text detailing the incident.

The dataset is unprocessed, containing significant noise in the form of typos, inconsistencies, and ambiguities. This unstructured nature adds complexity, as the text requires extensive preprocessing to enable accurate classification.

Objectives

Our objective is to build a reliable NLP model that:

- **Classifies each crime description into appropriate subcategories** to ensure the report aligns with Government of India guidelines.
- Provides real-time classification to assist users in choosing correct subcategories, thereby reducing the potential for misreporting.

3. Methodology

Data Preprocessing
Given the unstructured nature of the dataset, extensive preprocessing was crucial to prepare the text data for classification. Key steps included:

1. **Normalization**: Standardizing text to improve consistency and model performance. This included converting text to lowercase and removing extraneous symbols and punctuation that did not contribute to semantic meaning.

2. **Tokenization**: Splitting text into individual words or tokens, enabling analysis of linguistic structure and making it easier to apply transformations like TF-IDF.

3. **Regular Expressions (RegEx)**: RegEx patterns were used to identify and clean common text issues such as excessive whitespace, repeated characters, and informal language often seen in user-generated content.

4. **EDA (Exploratory Data Augmentation)**: To account for the imbalances in the dataset, EDA techniques were applied to expand certain categories. This included generating synthetic examples for underrepresented subcategories, which improved model performance on less frequent classes.

Feature Engineering
To improve the representation of text data, word embeddings were utilized instead of traditional TF-IDF vectors. Word embeddings offer dense and context-aware vector representations of words, capturing semantic relationships more effectively. This shift enhanced the performance of deep learning models, especially the GRU architecture, by enabling them to leverage the contextual meaning of words. Key updates include:

- **Word Embeddings:** Pre-trained word embeddings (e.g., GloVe or Word2Vec) were used to initialize the embedding layer. This approach provided richer contextual information and improved model generalization by leveraging knowledge from large external corpora.

- **Fine-tuned Embeddings:** During model training, the embeddings were fine-tuned on the dataset to capture domain-specific nuances in the crime descriptions.

- **N-grams:** Unigrams and bigrams were still considered during exploratory analysis, but their usage shifted from explicit feature engineering to model interpretation and understanding patterns in the crime descriptions.

Model Selection

To classify text into categories and subcategories, multiple models were tested for performance and suitability. Two models—GRU and XGBoost—emerged as the primary choices due to their ability to handle the dataset's complexity and deliver robust performance.

1. **GRU (Primary Model):**
   A Gated Recurrent Unit (GRU) model was employed as the primary classifier for its ability to capture sequential patterns in text data efficiently. The GRU architecture effectively handled the contextual and temporal relationships within crime descriptions.

    - **Architecture Highlights:** The model consisted of an embedding layer initialized with pre-trained word embeddings, followed by GRU layers, dense layers, and a softmax output for classification.
    - **Regularization:** Dropout and batch normalization were incorporated to prevent overfitting and ensure stability during training.
    - **Training Strategy:** A combination of early stopping and learning rate scheduling was used to optimize performance without excessive computation.

2. **XGBoost:**
   XGBoost was chosen as a complementary model due to its interpretability and strong performance with structured features.
    - **Gradient Boosting:** XGBoost's iterative boosting mechanism provided robust predictions, particularly for rare and imbalanced subcategories.
    - **Parameter Tuning:** Hyperparameters such as learning rate, max depth, and the number of estimators were fine-tuned to optimize performance.
    - **Integration:** While primarily focused on structured features, XGBoost was also experimented with feature embeddings, bridging traditional machine learning with deep learning representations.

1. **Additional Models for Benchmarking**: Several other models were tested for comparison, including:
    - **Naive Bayes**: Chosen for its simplicity and speed in handling text classification tasks, it served as a baseline model.
    - **BERT (Bidirectional Encoder Representations from Transformers)**: Applied to assess the benefits of using a contextual language model. While BERT yielded promising results in capturing contextual nuances, it was computationally intensive and challenging to integrate within the hackathon timeframe.

Training and Validation
The dataset was split into training and validation sets to evaluate model performance. Cross-validation was employed to assess stability, particularly for imbalanced subcategories. The model was evaluated based on accuracy, precision, recall, and F1-score to ensure balanced performance across all classes.

Mapping: Subcategories to Categories

This mapping organizes the subcategories into three broad categories: Other Cyber Crime, Financial Fraud Crimes, and Women/Child Related Crime. Each category consolidates relevant subcategories to simplify analysis and classification

1. Other Cyber Crime

- Any Other Cyber Crime
- Cheating by Impersonation
- Cyber Bullying/Stalking/Sexting

- Cyber Terrorism
- Damage to Computer Systems
- Data Breaches
- Defacement of Websites or Unauthorized Changes
- Defacement/Hacking
- Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks
- Email Hacking
- Email Phishing
- Fake/Impersonating Profile
- Impersonating Email
- Intimidating Email
- Malicious code attacks (e.g., Virus, Worm, Trojan, Bots, Spyware, Ransomware, Crypto miners)
- Online Cyber Trafficking
- Online Job Fraud
- Online Matrimonial Fraud
- Profile Hacking/Identity Theft
- Provocative Speech of Unlawful Acts
- Ransomware
- SQL Injection
- Tampering with Computer Source Documents
- Unauthorized Access/Data Breach

2. Financial Fraud Crimes

- UPI-Related Frauds
- Internet Banking-Related Fraud
- E-Wallet Related Frauds
- Debit/Credit Card Fraud or SIM Swap Fraud
- Fraud Call/Vishing
- Cryptocurrency Crime
- Demat/Depository Fraud
- Online Gambling/Betting Fraud

- Business Email Compromise/Email Takeover

3. Women/Child Related Crime

- Rape/Gang Rape-Sexually Abusive Content
- Sale, Publishing and Transmitting Obscene Material/Sexually Explicit Material
- Child Pornography/Child Sexual Abuse Material (CSAM)

4. Results

Model Performance Metrics (With GRU):

- **Subcategory Accuracy: 0.9840**
- **Subcategory Macro Precision: 0.9835**
- **Subcategory Macro Recall: 0.9838**
- **Subcategory Macro F1-score: 0.9840**

Precision and Recall for Each Subcategory:

SUB CATEGORY	PRECISION	RECALL	F1 SCORE
ANY OTHER CYBER CRIME	0.9099	0.8296	0.8679
BUSINESS EMAIL COMPROMISE/EMAIL TAKEOVER	0.9987	1.0000	0.9993
CHEATING BY IMPERSONATION	0.9759	0.9910	0.9834
CHILD PORNOGRAPHY/CHILD SEXUAL ABUSE MATERIAL (CSAM)	0.9971	0.9947	0.9959
CRYPTOCURRENCY CRIME	0.9970	1.0000	0.9985
CYBER BULLYING/STALKING/SEXTING	0.9787	0.9764	0.9775
CYBER TERRORISM	0.9951	1.0000	0.9975
DAMAGE TO COMPUTER SYSTEMS	1.0000	1.0000	1.0000
DATA BREACHES	1.0000	1.0000	1.0000
DEBIT/CREDIT CARD FRAUD OR SIM SWAP FRAUD	0.9400	0.9453	0.9426
DEFACEMENT OF WEBSITES OR UNAUTHORIZED CHANGES	1.0000	1.0000	1.0000
DEFACEMENT/HACKING	1.0000	1.0000	1.0000
DEMAT/DEPOSITORY FRAUD	0.9793	1.0000	0.9896
DENIAL OF SERVICE (DOS) AND DISTRIBUTED DENIAL OF SERVICE (DDOS) ATTACKS	1.0000	1.0000	1.0000
E-WALLET RELATED FRAUDS	0.9704	0.9799	0.9751
EMAIL HACKING	0.9432	1.0000	0.9707
EMAIL PHISHING	0.9990	0.9977	0.9983
FAKE/IMPERSONATING PROFILE	0.9785	0.9924	0.9854

FRAUD CALL/VISHING	0.9520	0.9701	0.9610
IMPERSONATING EMAIL	1.0000	1.0000	1.0000
INTERNET BANKING-RELATED FRAUD	0.9354	0.9505	0.9429
INTIMIDATING EMAIL	1.0000	1.0000	1.0000
MALICIOUS CODE ATTACKS (E.G., VIRUS, WORM, TROJAN, BOTS, SPYWARE, RANSOMWARE, CRYPTO MINERS)	1.0000	1.0000	1.0000
ONLINE CYBER TRAFFICKING	0.9985	0.9980	0.9983
ONLINE GAMBLING/BETTING FRAUD	0.9968	0.9917	0.9943
ONLINE JOB FRAUD	0.9869	0.9995	0.9931
ONLINE MATRIMONIAL FRAUD	0.9969	1.0000	0.9985
PROFILE HACKING/IDENTITY THEFT	0.9886	0.9881	0.9884
PROVOCATIVE SPEECH OF UNLAWFUL ACTS	0.9966	0.9964	0.9965
RANSOMWARE	0.9998	1.0000	0.9999
RAPE/GANG RAPE-SEXUALLY ABUSIVE CONTENT	0.9999	1.0000	0.9999
SQL INJECTION	1.0000	1.0000	1.0000
SALE, PUBLISHING AND TRANSMITTING OBSCENE MATERIAL/SEXUALLY EXPLICIT MATERIAL	0.9870	0.9951	0.9910
TAMPERING WITH COMPUTER SOURCE DOCUMENTS	1.0000	1.0000	1.0000
UPI-RELATED FRAUDS	0.9179	0.8289	0.8711

5. Challenges Faced

1. Handling Messy Text

The dataset presented a significant challenge with its raw and unstructured text, which included:

- **Typos and Inconsistencies**: Frequent typographical errors and varying phrasing made it difficult to standardize descriptions across similar cases.

- **Abbreviations and Slang**: The use of abbreviations and informal language required careful handling to avoid loss of meaning during preprocessing.

- **Special Characters and Unwanted Symbols**: These were removed using regular expressions, although some instances required further fine-tuning to retain meaningful content.

- **Ambiguous Terminology**: Certain terms and phrases were open to interpretation, which added complexity to subcategory classification. Techniques like normalization and tokenization were used to create a more uniform data structure, allowing the model to interpret the input text more reliably.

2. Managing Imbalanced Data

The dataset was highly imbalanced, with some categories and subcategories underrepresented. This imbalance posed two main issues:

- **Biased Model Predictions**: The model tended to favor the majority classes, reducing its accuracy for underrepresented subcategories. To address this, we employed **class weighting** and **sampling techniques**. For example, we used **oversampling** of minority classes to increase their representation in the training set and **undersampling** of dominant classes to balance the data.

- **Challenge in Precision-Recall Balance**: Imbalanced data often leads to a trade-off between precision and recall, particularly for minor subcategories. To address this, we focused on tuning hyperparameters to achieve a more balanced F1-score, ensuring fair representation across all subcategories.

3. Handling Null Values

While the dataset contained mostly complete records, some fields were missing, impacting the model's learning process. We addressed this by:

- **Removing Rows with Significant Missing Data**: For cases where missing values significantly affected description quality, we excluded those rows from the dataset.

- **Imputation for Minor Null Cases**: For minor missing values, we used simple imputation techniques to fill in missing data, particularly for critical fields required by the model.

4. Model-Related Challenges

Due to the size and complexity of the dataset, additional challenges arose related to model training and performance optimization:

- **Overfitting**: Given the high variability in text, overfitting was a concern, especially with models like GRU. To mitigate this, we used techniques such as **early stopping**, **regularization** (L2 regularization), and **cross-validation**. These methods helped prevent the model from memorizing training data, improving its generalizability.

- **Optimization for Large Datasets**: The dataset's large size posed constraints on computational resources and processing time. We tackled this by:

- **Batch Training**: Breaking down the training process into manageable batches to reduce memory load.

Dimensionality Reduction: Applying Word Embedding selectively and capping the maximum features to prevent the feature space from becoming too large, which optimized both memory usage and model efficiency.

6. Conclusion

Summary of Accomplishments
In this project, we developed an NLP model capable of accurately classifying crime-related text data into specific subcategories based on raw, unstructured input. Through extensive preprocessing, feature engineering, and experimentation with various machine learning techniques, our final model (GRU) achieved strong performance despite significant data challenges, such as imbalance and high variability in text descriptions. The model demonstrated effective handling of complex and noisy data, showing robust accuracy across categories and subcategories.

Key Learnings
This project reinforced the importance of:

- **Thorough Data Preprocessing**: Managing unstructured and messy text data through normalization, tokenization, and regex proved essential to obtaining reliable results.
- **Class Imbalance Solutions**: Addressing data imbalance with sampling and class weighting contributed to fairer, more accurate classification, even for underrepresented subcategories.
- **Optimization for Large Datasets**: Efficient handling of large datasets was crucial for maintaining computational performance while preventing overfitting.

Impact and Real-World Potential
The developed model is a step toward enhancing the National Cyber Crime Reporting Portal (NCRP) by supporting citizens in filing accurate, well-categorized cybercrime reports. By guiding users in real-time, this model could help streamline the reporting process and facilitate more responsive action from authorities. The model's framework can also be adapted to similar tasks involving classification of unstructured data, making it a valuable asset in other government and legal applications.

Future Directions
To further improve model performance and adaptability, we propose:

- **Incorporating Advanced NLP Models**: Future iterations could integrate more complex NLP architectures, such as fine-tuned BERT or other transformer-based models, to enhance contextual understanding.
- **Expanding Feature Engineering**: Exploring additional feature engineering techniques, like entity recognition or sentiment analysis, could improve the model's ability to discern nuanced differences in crime descriptions.
- **Exploring Real-Time Deployment**: To maximize impact, the model could be optimized for deployment as a real-time classification service, with updates based on newly reported cases to keep classification criteria current.

Closing Statement
This project highlights the feasibility and utility of deep learning in addressing societal needs through intelligent automation. With continued refinements, our model holds the potential to improve the NCRP's operational efficiency and better support citizens in reporting cybercrimes, aligning with broader goals for data-driven governance and citizen engagement.

7. References

Data Processing:

- NumPy: Oliphant, T. E. (2006). A guide to NumPy. T. E. Oliphant.
- Pandas: McKinney, W. (2011). pandas: a foundational library for data analysis and manipulation. Python for Data Analysis.
- Scikit-learn: Pedregosa, F., et al. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825-2830.
- NLTK: Bird, S., Klein, E., & Loper, E. (2009). Natural language processing with Python. O'Reilly Media, Inc.

Model Training:

- GRU (Gated Recurrent Unit): Cho, K., et al. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.
- XGBoost: Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785-794.
- Scikit-learn: Pedregosa, F., et al. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825-2830.

Data Balancing:

- Scikit-learn Resampling: Scikit-learn documentation. (n.d.). Resampling strategies in scikit-learn. Retrieved from https://scikit-learn.org/stable/modules/classes.html#module-sklearn.utils

Exploratory Data Analysis (EDA):

- Matplotlib: Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90-95.
- Seaborn: Waskom, M. L., et al. (2020). Seaborn: statistical data visualization. Journal of Open Source Software, 5(51), 2411.

8. Appendices

Appendix A: Data Preprocessing Steps

- Detailed outline of preprocessing techniques, including:
  - **Normalization Techniques**: Explanation of how text normalization was applied and examples of typical transformations.
  - **Tokenization Details**: Information on the tokenization approach, such as word-based or subword tokenization.
  - **Regex Patterns**: Common regular expressions used to clean unwanted symbols and examples of text transformations.
  - **Handling Null Values**: Documentation of methods used to handle missing values.

Appendix B: Exploratory Data Analysis (EDA) Results

- **Category and Subcategory Distributions**: Visualizations or tables showing the imbalance among categories and subcategories.
- **Word Frequency Analysis**: Summary statistics on the most common words and phrases across different subcategories.
- **Data Imbalance Visuals**: Plots depicting the distribution of data across classes to illustrate the degree of imbalance.

Appendix C: Model Hyperparameters and Tuning

- **Hyperparameter Tuning Logs**: Detailed listing of parameters tested for GRU, including learning rate, maximum depth, and other key settings.
- **Best Parameter Configuration**: The final set of hyperparameters chosen for the primary model, along with rationale for each choice.
- **Grid Search or Cross-Validation Details**: Information on the validation techniques used for selecting optimal parameters.

Appendix D: Evaluation Metrics and Confusion Matrices

- **Confusion Matrices for Key Subcategories**: Confusion matrices displaying model performance across major subcategories.
- **Precision, Recall, and F1-Score Metrics**: Detailed performance metrics for each subcategory, offering insight into model accuracy and potential areas for improvement.

Appendix E: Code Snippets

- **Key Functions for Preprocessing**: Code snippets for custom functions used in data preprocessing, tokenization, and text cleaning.
- **Feature Engineering with TF-IDF**: Sample code illustrating how TF-IDF was applied to convert text into feature vectors.

- **Model Training Script**: Essential portions of the code used for training and tuning the XGBoost model.

Appendix F: Additional Experiments and Observations

- **Comparison of Model Performances**: Summary tables showing how different models (e.g., Naive Bayes, SVM, BERT, XGBoost) performed relative to GRU.
- **Insights from Failed Approaches**: Brief discussion of models or methods tested that did not yield significant improvements, along with reasons for their limitations.

Summary of Findings

The NLP analysis for crime report classification revealed significant insights:
1. Sentiment Trends : Most crime descriptions carried neutral or negative sentiments, aligning with the nature of cybercrime. Positive sentiments were rare and often linked to non-criminal contexts.
2. Common Themes and Topics :
  - High-frequency themes included financial fraud, phishing, and unauthorized access.
  - Women/child-related crimes predominantly discussed abuse and trafficking.
3. Visualization Insights:
  - Heatmaps of word frequencies showed distinct vocabularies for each subcategory.
  - Temporal trends indicated spikes in phishing and ransomware reports during specific months.
4. Classification Metrics:
  - Text classification accuracy reached 98.40% for subcategories.
  - Errors were attributed to ambiguous phrasing and overlapping terms between categories.

Model Evaluation

The GRU model demonstrated exceptional performance:
- Accuracy : 98.40% for subcategories.
- Precision and Recall: High scores across most subcategories, ensuring balanced classification.
- Error Analysis: Most misclassifications occurred in subcategories with overlapping descriptions. Improving feature engineering and incorporating advanced models like BERT can address these challenges

Implementation Plan

To enhance the system further, the following steps are proposed:

1. **System Changes**: Deploy the model as a microservice, enabling real-time classification.

2. **Further Analysis**: Integrate Named Entity Recognition (NER) for better contextual understanding.

3. **Deployment Plans**: Implement a user-friendly web interface for crime reporting. Continuous model updates will ensure alignment with new data trends.

Plagiarism Declaration

We declare that this project report is our original work and has not been copied from any other source. References to external works, libraries, and frameworks are appropriately cited.