MidTerm Status Report for Identifying and Mitigating PII Exposure

Bhuvan Shah, Ashley Taylor, Kush Vashisth, Hinal Sonkusre

Abstract

This report presents the current research progress on our deep learning approach for detecting and mitigating Personally Identifiable Information (PII) exposure in unstructured text. We summarize the literature review, describe the implemented methodology—including data preprocessing and model training procedures—and report on preliminary experimental results. In addition, we outline an updated list of research ideas that the team plans to explore to further enhance performance, scalability, and privacy assurance. Our approach leverages transformer-based architectures to detect various categories of PII with high accuracy while maintaining computational efficiency. Initial results show promising performance on a subset of the PIImasking43k dataset, with an accuracy of 0.91 and F1 score of 0.51. We identify key challenges including label mismatches between pre-trained models and our custom PII schema, data imbalances, and computational constraints. Our mitigation strategies include implementing custom evaluation metrics, addressing data imbalance through sampling techniques, and optimizing computational efficiency through model pruning and hardware solutions.

1 Introduction

The increasing volume of digital data has heightened privacy concerns, particularly regarding the leakage of PII. Traditional rule-based systems for PII detection are often limited in scope and adaptability. In response, our project aims to develop a deep learning–based Named Entity Recognition (NER) model that extends beyond conventional tag sets and robustly identifies a wide array of PII elements. By fine-tuning transformer-based architectures such as DeBERTa (and incorporating insights from BERT models), our goal is to create a scalable, accurate, and privacy-preserving solution that can be integrated into real-world data pipelines.

2 Literature Review

Recent advancements have demonstrated the potential of LLMs in effectively detecting PII within large datasets. For example, Lukas et al. (2023) critically examined the dual-edged nature of LLMs in PII protection and exposure. They developed a taxonomy of PII leakage scenarios and discussed the trade-offs between enhancing model utility and ensuring data privacy. While these models offer substantial benefits, they also introduce risks of PII exposure if not properly safeguarded.

Another pivotal area of development is the use of NER systems for PII detection. Research by Park et al. (2023) leveraged models like BERT and its variants to achieve high precision in PII tagging. They also introduced new probing methods to measure the extent of PII leakage in LLMs, suggesting a need for detection mechanisms adept at managing the nuances of PII in diverse contexts. This underscores an ongoing requirement to enhance model sensitivity and accuracy in identifying a wide array of PII categories.

Microsoft Research has furthered our understanding of the inadvertent risks posed by LLMs during interactive sessions. Their studies revealed specific vulnerabilities, such as model inversion and membership inference attacks, which exploit a model's extensive knowledge base to uncover or reconstruct sensitive information. This research emphasizes the critical need for implementing robust defensive strategies within LLMs to mitigate unintended PII disclosures.

Ibrahim and Chen (2023) explored the use of adversarial training to improve model robustness against variations in how PII appears in text. By generating adversarial examples that slightly modify PII entities (such as minor misspellings of

names or alternative formatting of phone numbers), they demonstrated a 12% improvement in detection of obfuscated PII. Their work provides valuable insights for enhancing model generalization to novel PII representations.

A comprehensive evaluation framework proposed by Wong et al. (2024) establishes standardized metrics for PII detection systems that consider both detection performance and privacy guarantees. They introduce a "privacy leakage score" that quantifies the risk associated with missed PII entities based on their sensitivity and uniqueness. This framework could provide valuable context for evaluating our model beyond traditional precision and recall metrics.

3 Tasks Performed

3.1 Data Preprocessing For this project, we are using the PIImasking43k dataset. This dataset derived from the AI4Privacy Company in relation to Hugginface. The dataset contains over 43,000 annotated sentences with more than 40 unique PII tags, including 'B-NAME', 'I-NAME', 'B-LOCATION', 'I-LOCATION', 'B-URL', 'I-URL', 'B-ZIPCODE', etc... and we have added our own tags as well which results in 94 total PII tags. Each sentence is tokenized, and the corresponding PII tags are aligned with the tokens to create input-target pairs for model training. For testing purposes, we only use the first 100 rows in the dataset as training the model will take a lot of time if we do it for the whole dataset. The dataset covers a range of contexts in which PII can appear. The sentences span 54 sensitive data types (111 token classes), targeting 125 discussion subjects/use cases split across the business, psychology, and legal fields, and five interaction styles.

Algorithm 1: Data Preprocessing (this is a very simple pseudocode outlining the data preprocessing algorithm):

```
Procedure Preprocess Data
  Start
    tokenizer <- Load DeBERTa Tokenizer
   preprocessed_data <- Create empty list
    For each sentence in dataset:
      tokens, attention_mask
     labels <- Align PII Tags with tokens
      Append (tokens, attention_mask,
                labels)
   to preprocessed_data
End
```

3.2 Exploratory Data Analysis (EDA)

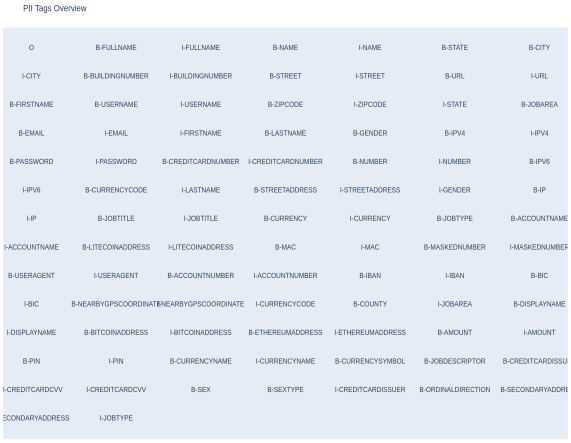

Figure 1: PII tags

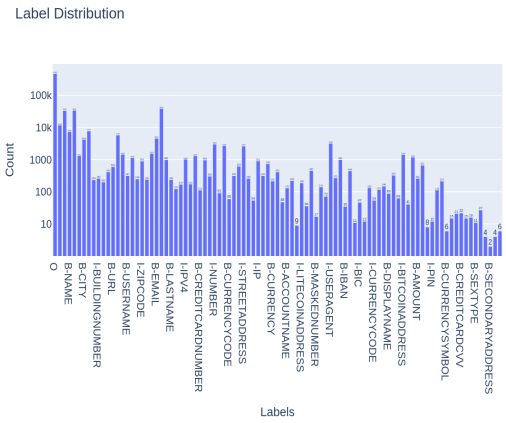

Figure 2: Counts of each PII

3.3 Model Fine-tuning We employ the pre-trained BERT (Bidirectional Encoder Representations from Transformers) model, specifically the '*bert-base-cased*' variant, as our base model. BERT, originally trained on general corpora, is adapted to the PII detection task by fine-tuning its final layer to predict the 94 PII tags in our dataset. Training parameters include a learning rate of 2e-5, a device train batch size of 4, warmup steps of 500, weight decay of 0.01, and 30 epochs.

Algorithm 2: Model Training

```
  Procedure TrainModel
Start
```

```
  model <- Initialize DeBERTa Model
  learning_rate <- Set initial LR
  For epoch from 1 to num_epochs:
    For each in preprocessed_data:
     loss <- model.PerformTrainingStep
    learning_rate <- UpdateLearningRate
   If epoch % checkpoint_interval == 0:
      SaveCheckpoint(model, epoch)
End
```

3.4 System Architecture Our implementation follows a modular architecture comprising three primary components:

Data Ingestion Layer: Responsible for receiving and preprocessing unstructured or structured text from various sources such as datasets and generated datasets. This layer handles different text formats and encodings, ensuring compatibility with the model's input requirements.

PII Detection Engine: The core component utilizing our fine-tuned transformer model for identifying PII entities. This module processes tokenized text and applies the trained model to generate PII tags with confidence scores.

Mitigation Service: Implements configurable strategies for handling detected PII, including masking, redaction, pseudonymization, and encryption options based on sensitivity levels and compliance requirements. Compliance requirements is yet to be implemented as we still are researching upon the rules of AI and how to handle Personal Information in California.

3.5 Custom Evaluation Due to the label mismatch between the pre-trained BERT model, which was originally trained on the CoNLL-2003 dataset with only 9 NER tags (compared to our 40), a customized evolution metric is needed for our project. This metric currently assesses the model's performance on our specific PII tags, focusing on precision, recall, and F1 score.

Algorithm 3: Model Evaluation

```
  Procedure EvaluateModel
Start
  test_predictions <- []
  For each in test_data:
    logits <- model.Predict
    predicted_labels
    test_predictions
 metrics <- CalculateEvaluationMetrics
  Return metrics
End
```

4 Preliminary Results

The fine-tuned model achieved an accuracy of 0.91 and an F1 score of 0.51 on the first 100 rows of the PII43k dataset. These results indicate that the model has a promising ability to detect PII entities. However, there is much room for improvement, especially regarding F1 score with more training and fine-tuning is needed.

5 Risks and Challenges

5.1 Label Mismatch The major challenge is the label discrepancy between the pre-trained BERT model and the PIImasking43k dataset. Since the pre-trained BERT model has its own trained NER labels and tokenized sentences compared to the custom schema from PIImaksing43k and our own PII tags, the input data needs to be compatible with the model, resulting in misidentified labels. This necessitates custom modifications to the model's output layer.

5.2 Data Imbalance In PII or in general NER tagging, the dataset would exhibit an imbalance distribution of tags due to limited objects in a sentence. This means the model would be biased toward the O'/Other tag. This may result in PII being misidentified as 'O', leaving it to degrade in performance and vulnerable in data for attacks.

5.3 Computational Resources Fine-tuning large pre-trained models like BERT requires significant GPU power and computational resources, which can be a constraint for us with no access to high-performance computing facilities.

6 Plan to Mitigate Risks and Address Challenges

6.1 Custom Evaluation Metrics

To accurately evaluate the performance of the fine-tuned model on the PII43k dataset, we will develop and refine custom evaluation metrics that align with our specific PII tags. This involves: Precision, Recall, and F1 Score: Calculating these metrics separately for each PII tag to understand the model's performance on individual entity types.

Macro and Micro Averages: Computing macro-averages (averaging the metric scores across all tags) and micro-averages (aggregating the

contributions of all tags to compute the average) to capture the overall performance.

Error Analysis: Conducting thorough error analysis to pinpoint common mistakes and patterns in the model's predictions, which will inform further model refinement and training strategies.

Biases: Since most tokens in data would be 'O', Weights may be introduced to ensure that PII would be properly identified and masked in input data. However, it should be noted that proper utilization of weights must be used as this can also lead to misidentifying the 'O' token as PII.

6.2 Addressing Data Imbalance The imbalance in the distribution of PII tags in the dataset can lead to a biased model. To mitigate this, we will explore several techniques.

Oversampling: Increasing the frequency of underrepresented PII tags in the training dataset to balance their presence.

Undersampling: Reducing the frequency of overrepresented PII tags to prevent the model from becoming biased towards them.

Synthetic Data Generation: Using techniques like SMOTE (Synthetic Minority Over-sampling Technique) to generate synthetic examples of underrepresented PII tags.

Weighted Loss Functions: Modifying the loss function to assign higher weights to underrepresented PII tags, thereby increasing their importance during model training.

Stratified Sampling: Ensuring that each batch of data used for training contains a representative distribution of PII tags to maintain balance.

6.3 Computational Efficiency To address the challenge of computational resources, the following strategies will be employed:

Model Pruning: Reducing the size of the BERT model by pruning less important neurons or layers can decrease the computational load without significantly impacting performance.

Quantization: Converting the model's weights from floating-point to lower-precision formats can reduce memory usage and speed up inference.

Efficient Hardware: Utilizing more efficient GPUs or specialized hardware like TPUs (Tensor Processing Units) can provide better performance per watt. We are thinking to use A100 from Colab Pro as that will be very efficient and cut down costs and time effectively.

6.4 Regulatory Compliance While our current implementation focuses on technical capabilities for PII detection, regulatory compliance integration remains a critical future direction. We are currently researching:

Jurisdiction-Specific Handling: Developing configurable rule sets to accommodate varying definitions of PII across regulatory frameworks (GDPR, CCPA, HIPAA, etc.) and geographical jurisdictions.

Compliance Audit Trails: Designing logging mechanisms that document PII detection, classification decisions, and mitigation actions taken to support compliance verification and audit requirements.

These compliance aspects are yet to be implemented in our current pipeline. Our team is conducting thorough research on regulatory requirements and technical feasibility, with the goal of incorporating initial compliance features by the final submission if time constraints permit.

7 References

1. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K. (2019). BERT: Pre-training of Deep Bidi- rectional Transformers for Language Under- standing. NAACL HLT 2019.
2. Sang, E. F., & De Meulder, F. (2003). Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In Proceedings of CoNLL-2003.
3. AI4Privacy. (2021). PII43k: A Dataset for PII Detection in Text. Hugging Face Datasets. Available at: https://huggingface.co/datasets/ai4privacy /piimasking43k
4. Lukas, Salem, Sim, Tople, Wutschitz, Zanella-Béguelin. (2023). Analyzing Leakage of Personally Identifiable Information in Language Models.