Evaluating different NLP models to detect personally identifiable information

Aref Hasan[1], *Franziska* Marb[1], *Jannik* Völker[1], and *Nik* Yakovlev[1]

[1] *Baden-Württemberg Cooperative State University (DHBW), Mannheim, Germany*

Abstract. This paper examines the importance of protecting Personally Identifiable Information (PII) in the digital age and evaluates several Natural Language Processing (NLP) models, including Flair, Spacy-Stanza, NLTK, and a Large Language Model (LLM). The study identifies limitations in existing frameworks and introduces a finetuned SpaCy model using a specialized PII dataset, which achieves high precision, recall, and F-score metrics. The SpaCy model, after being fine-tuned, proves to be a reliable solution for detecting PII. This demonstrates its potential for promoting responsible data management and preserving privacy in the constantly changing digital environment.

1 Contributions

The research was carried out by all authors with similar effort. For the sake of transparency, Aref Hasan researched Flair, Franziska Marb researched SpaCy-Stanza, Jannik Völker ran the Large Language Model (LLM) and Nik Yakovlev researched NLTK and fine-tuned a SpaCy model. The resulting application was a team effort. Aref Hasan programmed the interface between the Graphical User Interface (GUI) and the model, Franziska Marb designed and built the GUI, Jannik Völker connected the model to the GUI and introduced object orientation, and Nik Yakovlev trained the model and built an interface for the model.

2 Introduction

In the digital era, where online interactions and data exchanges have become omnipresent, the concept and importance of Personally Identifiable Information (PII) have taken center stage. PII, defined as any data that can be used to identify an individual, has become a cornerstone in discussions of privacy and data security. Its relevance is amplified in our current context, where the internet serves as a vast repository of personal information, making the protection of PII not just a regulatory requirement but a societal imperative.

The distribution of digital data necessitates advanced methods for its safe handling, particularly given the ease with which sensitive information can be disseminated and exploited. Here, Natural Language Processing (NLP) emerges as a key player. The role of NLP in identifying and managing PII is more crucial in in today's connected world. By leveraging NLP technologies, we can efficiently process large volumes of text data to identify, categorize, and protect PII, thereby upholding data privacy and complying with strict data protection laws.

This integration of NLP into PII detection underscores a broader trend: the convergence of technology and ethics in the digital age. As NLP evolves, so too does our ability to safeguard personal information, marking a significant stride in responsible data management and privacy preservation in the internet era.

In response to these challenges, the project started with the development of a PII detector. This undertaking involved a comprehensive investigation and evaluation of various NLP models and frameworks. Analyzing these tools carefully, we aimed to develop an application capable of detecting and protecting PII.

3 Models

To evaluate the models in our project, we consistently utilized the PII dataset available on Hugging Face, named pii-masking-200k, provided by ai4privacy. This comprehensive dataset is tailored for PII detection and masking, featuring an extensive range of 54 PII classes, as documented in [1]. Its wide array of personal information classes significantly enhanced our capability to assess various PII types, surpassing the standard entities recognized by the baseline model. This dataset was integral in our project, serving as the benchmark for evaluating all the models we investigated.

3.1 FLAIR

Flair is an innovative framework in the realm of NLP, offering a simple yet effective interface for a variety of language processing tasks. Developed by Akbik et al. (2019), it facilitates sequence labeling, text classification, and language modeling, supporting a wide range of word and document embeddings. Notably, its context-aware representation learning significantly enhances word representations, leading to superior performance in tasks such as Named Entity Recognition (NER) [2].

In an evaluation involving the first 15,000 lines of a dataset, Flair exhibited robust performance in recognizing entities such as persons, organizations, and locations. However, these categories, although important, are not comprehensive. Despite its strengths, Flair demonstrates limitations in the identification of PII. Specifically, the model's parameters for NER, particularly in the flair/ner-english version, are limited to four categories: persons, organizations, locations, and miscellaneous entities (MISC).

This categorization, while encompassing a wide range of entities, is somewhat inadequate in the context of PII. PII can include a variety of data types, such as email addresses, phone numbers, social security numbers, and other personal identifiers, which extend beyond the basic categories offered by Flair. The model's lack of specificity in its parameters may result in failing to identify many forms of sensitive data that are essential for PII detection and protection.

Other NLP models have extended their parameters to include a more diverse range of entity types, thereby enabling the recognition of more specific types of PII. This expansion in entity recognition provides a more comprehensive coverage in scenarios where the identification of sensitive information is paramount. Consequently, these models are more suitable for tasks that demand rigorous PII identification and protection.

In summary, while Flair delivers robust performance in general NER tasks, its constrained set of entity categories is inadequate for tasks requiring detailed and extensive recognition of personal and sensitive data. This limitation hampers the model's effectiveness in scenarios where an exhaustive identification of PII is necessary, making other models with expanded parameters more suitable for such specific applications.

3.2 SPACY-STANZA

Spacy-Stanza represents an advanced synergy of two prominent NLP libraries, SpaCy and Stanza. SpaCy is known for its unparalleled speed, ease of use, and efficiency in natural language processing. In parallel, Stanza, formerly known as "StanfordNLP", is an NLP library developed by the renowned Stanford University, which provides sophisticated functions for the syntactic and semantic analysis of texts.

The convergent implementation of spaCy and Stanza allows developers and researchers to exploit the full potential of both libraries. SpaCy provides an easy-to-use API for text processing, while Stanza offers advanced models and techniques for tokenization, part-of-speech (PoS) tagging, lemmatization, and NER.

A major advantage of SpaCy-Stanza is its language diversity, making it a highly versatile solution for NLP applications. The use of pre-trained models allows users to perform complex text analysis without having to train their own models from scratch. This makes it much easier to get started with NLP development and speeds up the development process.

In addition to the basic functionalities, SpaCy-Stanza also offers extension options and customization features that allow for flexible configuration according to individual requirements. As a result, the integration of Stanza into spaCy has created a comprehensive NLP library that appeals to both novice and experienced developers and covers a wide range of natural language processing applications.

SpaCy-Stanza incorporates privacy-conscious design principles when handling PII. The library's customizable nature allows developers to implement and fine-tune privacy measures to meet specific requirements. This is crucial for applications that deal with sensitive data, ensuring compliance with data protection regulations and safeguarding user privacy year [3] [4] [5] [6] [7] [8].

In the evaluation of 5000 selected data sets, SpaCy-Stanza achieved mixed results in terms of recognizing PII. Using NER, 20 different PII categories were recognized. However, there were problems with the automatic assignment of the different categories of SpaCy-Stanza to those present in the data set. A manual assignment was made for the evaluation in order to be able to evaluate the quality of the results in comparison with other models. Accuracy was the decisive factor in selecting the right model for the PII Detector application. With a maximum accuracy of 0.43 in the NAME category, the model is significantly behind the performance of other models, which are described below.

To summarize, SpaCy-Stanza is a comprehensive and powerful NLP library that offers a harmonious blend of user-friendly interface and advanced linguistic analysis features. However, it is not suitable for the use case of the PII detector.

3.3 NLTK

The Natural Language Toolkit (NLTK) is a prominent library in NLP that offers a variety of text processing tools, such as tokenization, parsing, tagging, and classification [9]. Its NER component is particularly relevant for detecting PII [cf. 10, p.281 et sqq.].

NLTK's NER toolkit is capable of identifying and categorizing entities in text, including names, organizations, and locations [cf. 10, p.281 et sqq.]. However, it is primarily designed for general entity recognition and not specifically for PII detection. This means that while NLTK can identify certain types of PII, such as names and locations, it may not effectively recognize more specific forms like social security numbers or email addresses without additional customization.

A key strength of NLTK is its comprehensive set of tools for text analysis, which enables the creation of customized solutions for various information extraction tasks, including PII detection [11]. However, it lacks dedicated models for PII detection, unlike specialized frameworks in this field. Users need to develop their own rules or models for PII, which can be complex and requires a deep understanding of both NLP and PII detection methodologies.

In comparison to other NLP frameworks with dedicated PII detection components, NLTK requires more user effort and expertise to achieve effective results in this area. It offers a robust foundation for general text processing but demands substantial customization for specific applications like PII detection.

3.4 LLM

With advances in NLP, LLMs such as GPT-4 have shown potential in recognising and categorising sensitive information such as PII. This section presents an innovative approach using a local instance of the llama2-13b model, specifically the *nous-hermes-lama2-13b* model, using the Python library GPT4All to detect PII in textual content.

The methodology involved the development of a specialised prompt structure to guide the LLM in identifying PII. This was designed using the one-shot principle and instructed the model to search for personal information within a given text and to respond in JSON format. The categories of personal information were defined and also given. An example was provided to illustrate the expected response format. The prompt was as follows *"You are searching for personal info in a given text. Respond by stating all personal information in json format. Personal information can be the following: ['PREFIX', 'FIRSTNAME', [...], 'SSN']. Example: Hello Margareth, how are you? Here is my IBAN DE05 8903 8742 12342 32. Response: 'FIRSTNAME': 'Margareth', 'IBAN':'DE05 8903 8742 12342 32'"*

After the instruction prompt, the model was given a prompt containing text that potentially contained PII. It looked like this: *This is the text with personal info: 'A students assessment was found on device bearing IMEI: 06-184755-866851-3. The document falls under the various topics discussed in our Optimization curriculum. Can you please collect it?'*

The LLM ran on a Windows 11 computer with 32GB of RAM and a Ryzen 5 3600 processor. On average, it took about 35 seconds to answer a prompt. Knowing that the dataset contained around 43 thousand records, a sample size of 3000 was chosen to be tested with the LLM. The following observations can be made when examining the responses given by the LLM.

First, the LLM likes to include the instruction in it's response, making it harder to parse the response into json format. This can be partially cleaned up. After cleaning and parsing the responses to JSON, about 32 % couldn't be parsed to JSON, leaving about 2000 samples with successfully parsed JSON.

Second, every given PII category in the dataset was detected at least once by the LLM, but it made up about 441 additional PII categories. These could include other ways of spelling the same category, such as telephone or telephone number, but also categories that were not present in the given text or in the instructions. Attempts were made to match different variants using Levenshtein distance in combination with Jaccard similarity, but this didn't give sufficiently accurate results. As a result, any category that didn't match the exact wording of the labelled category was considered as wrong.

The performance of the LLM was calculated in the following way. It was divided into three categories. Detected keys or PII categories (e.g. the acllm predicted the category IBAN), detected values (e.g. the LLM predicted +49 186 12345678) and detected key-value pairs (e.g. the LLM predicted both correctly (e.g. "age" = "20"). These categories were calculated by comparing the labels from the dataset with the predicted output from the LLM. Output that couldn't be parsed into JSON was still considered, but gave zero correct predictions.

The table 1 shows these scores. The number of detected values was very low, resulting in a very low accuracy. The accuracy has been calculated using the following formula. It ignores any prediction that the LLM made that was wrong in the sense that it predicted a wrong category or a wrong value, but still gives a general idea of how the LLM performed:

$$accuracy = \frac{detected}{undetected + detected}$$

Category	detected	undetected	Accuracy
Keys	432	5585	0.0718
Values	17	6000	0.0028
Pairs	7	6010	0.0012

Table 1. Accumulated results for prediction over 3003 sample texts

To investigate further, the first 20 examples were examined by hand. This was done to include examples such as phone or phone number and to get a different perspective on the LLM. The results shown in table 2 suggest that the LLM is capable of detecting PII, but that it is not useful in a productive environment where other components depend on standard input formatting.

Category	detected	undetected	Accuracy
Keys	12	57	0.1739
Values	13	56	0.1884
Pairs	9	60	0.1304

Table 2. Accumulated results for prediction over 20 sample texts (manually examined)

3.5 FINE-TUNED SPACY

Baseline Model: SpaCy's en_core_web_sm

In our pursuit of a robust Personally Identifiable Information (PII) detector, we initially selected SpaCy's `en_core_web_sm` model as our baseline [12]. This model is well-regarded for its efficiency in entity recognition, a critical feature for identifying PII. The `en_core_web_sm` model, part of SpaCy's collection of pre-trained models, supports a variety of entities which include [12]:

- Persons
- Organizations
- Locations
- Languages
- Dates
- Times
- Percentages
- Money
- Quantities
- Ordinals
- Cardinals

While this model provides a solid foundation for basic entity recognition, it falls short in covering the extensive range of PII categories necessary for comprehensive detection.

Finetuning and Testing

To address this limitation, we turned to the PII dataset pii-masking-200k. We focused on the English version of this dataset, which consists of approximately 43.5k entries [1]. To enhance our PII detection capabilities, we finetuned the standard SpaCy model on the first 35k entries of this dataset. This finetuning process was geared towards adapting the model to recognize a wider array of PII classes, thereby increasing its effectiveness in real-world scenarios.

The remaining 8.5k entries served as our testing ground to evaluate the finetuned model's performance. We paid special attention to the model's ability to accurately recognize and categorize 'location' and 'name' entities, crucial elements in PII detection.

Evaluation Results

The test results were highly encouraging, demonstrating the efficacy of our finetuning approach:

- Precision: 0.8841342486651411
- Recall: 0.9017075732233848
- F-score: 0.8928344470334495

These metrics indicate a high degree of accuracy in the model's ability to identify and correctly classify PII entities. Precision measures the model's ability to correctly identify PII entities as such, while recall assesses the model's capacity to detect all relevant PII entities within the text. The F-score provides a harmonic balance between precision and recall, offering a comprehensive measure of the model's overall performance.

Conclusion: Choosing the Finetuned Model for PII Detection

Given the robust test results and the expansive coverage of PII classes, we confidently selected this finetuned model for our PII detection application. Its enhanced capabilities make it well-suited for accurately identifying a wide range of PII in various contexts, thereby ensuring more effective privacy protection in data processing tasks. This approach exemplifies the integration of advanced NLP techniques with practical applications in data security and privacy, marking a significant stride in responsible data management.

References

[1] ai4privacy, *pii-masking-200k*, https://huggingface.co/datasets/ai4privacy/pii-masking-200k (2023)

[2] A. Akbik, D.A.J. Blythe, R. Vollgraf, *Contextual String Embeddings for Sequence Labeling*, in *International Conference on Computational Linguistics* (2018), `https://api.semanticscholar.org/CorpusID:52010710`

[3] *spacy · industrial-strength natural language processing in python* (2024), `https://spacy.io/`

[4] D. K. (2019)

[5] explosion GitHub, *spacy-stanza* (2023), `https://github.com/explosion/spacy-stanza`

[6] explosion GitHub, *spacy* (2024), `https://github.com/explosion/spaCy`

[7] Explosion, *spacy-stanza · spacy universe* (2024), `https://spacy.io/universe/project/spacy-stanza`

[8] S.S.N. Group, *Stanza - a python nlp package for many human languages* (2024), `https://stanfordnlp.github.io/stanza/`

[9] E. Loper, S. Bird, *Nltk: The natural language toolkit* (2002), `cs/0205028`

[10] S. Bird, E. Klein, E. Loper, *Natural Language Processing with Python* (O'Reilly Media, Inc., 2019)

[11] NLTK Project Contributors, *Natural language toolkit* (2024), accessed: 2024-01-14, `http://www.nltk.org`

[12] E. AI, *English multi-task cnn trained on ontonotes, with glove vectors trained on common crawl.*, SpaCy (2017), `https://spacy.io/models/en#en_core_web_sm`

Acronyms

GUI Graphical User Interface

LLM Large Language Model

NER Named Entity Recognition

NLP Natural Language Processing

PII Personally Identifiable Information