Advanced Name Screening and Entity Linking with Existing NLP Models

Qiwenjing Jiang qj336@nyu.edu
Jialing Li jl9716@nyu.edu
Erqian Wang ew1708@nyu.edu
Kristine Zeng yz4792@nyu.edu

Abstract

In collaboration with Solytics Partners, we contributed to a machine learning system for assessing credit risk in banking. The system identifies high-risk individuals by extracting names from negative news reports and matching them against a database. Our research included evaluating various NLP models and name matching techniques such as BERT, RoBERTa, XLNet, TF-IDF Similarity, and Fuzzy Name Matching. The best results were achieved with BERT-CRF and XLM-RoBERTa models for NER and the Naive TF-IDF method for Name Matching. The project successfully applied AI in risk assessment despite challenges like data scarcity, indicating the potential for large language models in future improvements.

1 Introduction

In the banking sector, evaluating credit risks for new applications is crucial. High-risk individuals, often highlighted in negative news for criminal activities or fraud, pose a significant challenge. Our project, in collaboration with Solytics Partners, aims to address this issue by developing a machine learning system.

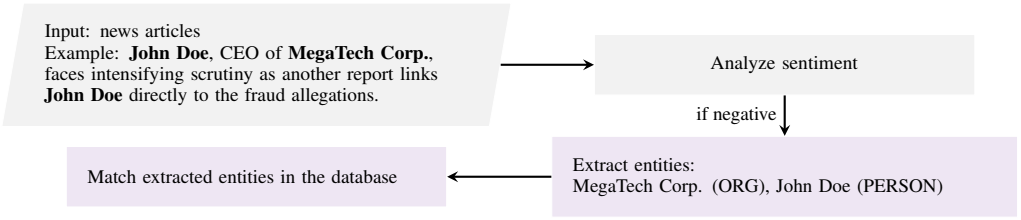

Figure 1: Flow Chart of the System

The system, as shown in the flow chart, takes news articles as input. It first evaluates the sentiment of each article. If the sentiment is negative, the system proceeds to extract named entities. It then verifies whether the extracted names are already in the database.

Our focus lies in two key areas: Named Entity Recognition (NER) and Name Matching. The NER component is responsible for identifying individuals and organizations within news articles, while the Name Matching part compares these extracted names with those in a database, flagging potential risks.

Preprint. Under review.

2 Related Works

2.1 Named Entity Recognition

The field of NER has seen significant advancements with the introduction of Natural Language Processing models such as BERT and its variants like BERT-BiLSTM-CRF and BERT-CRF [1]. These models have been successfully applied across various industries, demonstrating their ability to recognize and classify named entities in large datasets. Recent studies have also highlighted the potential of Large Language Models (LLMs), especially in the context of fine-tuning and in-context learning, with prompt engineering providing competitive results in NER tasks [2].

Despite the multitude of options available, due to the limited time and tools that we had, our group decided to concentrate on the implementation and fine-tuning of pre-trained NLP models and their variants. This approach enabled us to efficiently assess a diverse array of models, identifying and deploying the most effective one within a short timeframe.

2.2 Name Matching

The TF-IDF Similarity Metric is a key technique in Name Matching, noted for its effectiveness across diverse domains, as discussed by Cohen[3]. In addition, the Fuzz Score is based on the Levenshtein distance, which measures the minimal edits needed to change one string into the other and is commonly used as a metric for string similarity evaluation[4]. The collective clustering, has also been utilized for identity matching effectively. Li's 2015 study provides insights into the application of clustering methods for individual identity matching[5].

3 Problem Definition and Algorithm

3.1 Task

3.1.1 NER

Named entity recognition, a natural language processing subtask, identifies and categorizes key text information. In this project, news articles are input, and entities (words) with types are output. We focused on two entity types: PER (person) and ORG (organization). Due to confidentiality, our sponsor could not provide sufficient data for model training from scratch. Thus, we fine-tuned pretrained models using training data and adjusted hyperparameters.

3.1.2 Name Matching

Name Matching determines whether two different names represent the same person or organization. The same individuals or organizations could be under different names because of spelling variations, naming conventions, nicknames, initials, or typos. We implemented Name Matching models to measure whether the new incoming test names already exist in the database.

3.2 Algorithm

3.2.1 NER Algorithm

In this project, we have implemented six distinct pretrained models for NER, specifically spaCy Large, spaCy Transformer, BERT-Large, BERT-CRF, XLM-RoBERTa, and XLNet-Base. The overarching architecture of these models is encapsulated in the flow chart provided below, offering a high-level overview of their structural nuances.

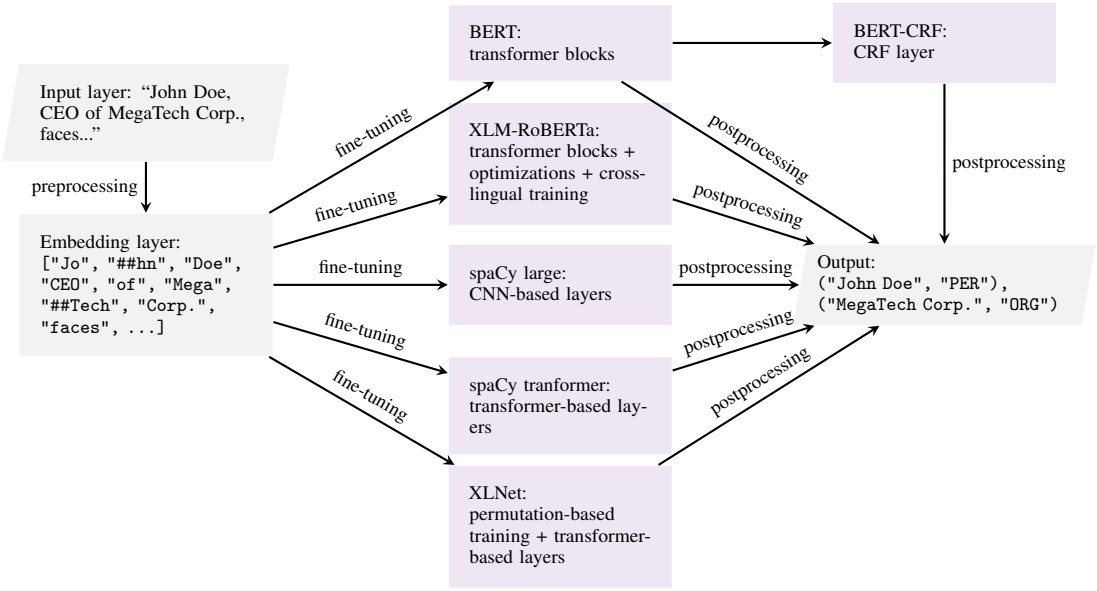

Figure 2: Architecture and Processing Flow of NER Models

BERT BERT leverages a deep transformer architecture to understand contextual relationships between words. For NER, it predicts entity labels for each token in a sentence using a token classification layer on top. Its bidirectional nature allows it to capture context from both directions, enhancing entity recognition accuracy [6].

BERT-CRF BERT-CRF combines BERT's contextual understanding with a CRF layer for sequence tagging. While BERT provides contextualized token embeddings, the CRF layer models the dependencies between labels in sequences, improving the coherence of entity predictions in NER tasks [7].

XLM-RoBERTa XLM-RoBERTa, an extension of RoBERTa, is trained on a large multilingual corpus. For NER, it effectively handles diverse languages, leveraging deep transformer layers. It excels in cross-lingual NER, identifying entities in a language-agnostic manner [8].

spaCy large spaCy's large model uses a CNN-based architecture for NER. It employs word vectors, context window features, and transition-based parsing for entity detection. This model is efficient for large-scale NER tasks, offering a good balance between speed and accuracy [9].

spaCy transformer The spaCy transformer model incorporates transformer-based features into spaCy's pipeline. For NER, it leverages pre-trained transformer models like BERT to enhance contextual understanding, leading to more accurate entity recognition across diverse text contexts [9].

XLNet XLNet employs a permutation-based transformer model, providing context-aware token representations. For NER, it captures bidirectional context and complex word dependencies, outperforming traditional models in recognizing nuanced entity types in text [10].

3.2.2 Name Matching Algorithm

To match names with considerations to conventions and nicknames, we adopted the Fuzzy Name Matching algorithm. Figure 3 shows a generalized Name Matching algorithm with examples. Here, names **John T. Doe** and **Doe, John** are matched with database name **John Doe**. After identifying the word components John and Doe, we calculate the Fuzz score between the input names and the database name. As both matches achieve a high similarity score, they are both identified as matches to name **John Doe**.

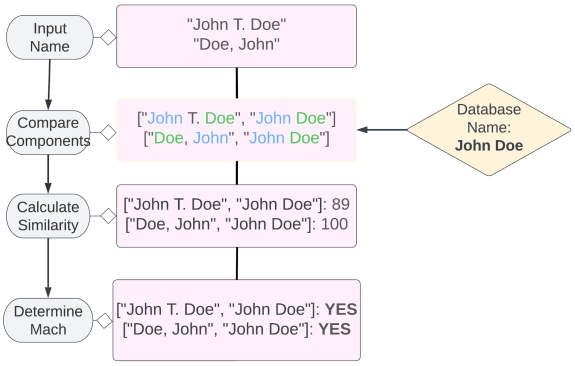

Figure 3: Diagram of Name Matching algorithm

Three name matching models based on Fuzzy were implemented:

Naive TF-IDF Name Matching In this model, we first created a TF-IDF matrix from database names, using a specified n-gram range and analyzer. This matrix represents the importance of each word/n-grams in name matching. Then, for each test name, the cosine similarities between this name and all names in the database were calculated and represented in the TF-IDF matrix. It then applied Fuzzy matching using token ratios to identify the top N similar names. The algorithm is complex enough to handle variations in name spellings and formats, yet it remains intuitive by using common Name Matching techniques like cosine similarity and Fuzzy matching.

Clustered Name Matching For this model, the unique database names were first vectorized with a character-based TF-IDF vectorizer. We then clustered the vectorized database names with the KMeans model. The number of clusters $\mathbf{n}$ was set to be the number of unique database names / 40, but no smaller than 20 and no larger than 500, to balance between the cluster size and cluster number based on the database size, avoiding overly sparse or dense clusters. We trained KMeans with the vectorized database matrix, and database names with similar segments would likely be assigned to the same cluster.

We then vectorized and clustered the new incoming test names with the trained TF-IDF vectorizer and KMeans clusters. The Fuzz scores between the test names and database names from the same cluster were computed. That is, if the test name 'Beth Wright' was assigned to Cluster 10, the Fuzz scores between it and every database name of Cluster 10 would be computed, and same for all other test names assigned to Cluster 10. The database names with the highest Fuzz scores were considered as the top matches.

Phonetic Name Matching This model is based on the pronunciation of the names. We encoded the database names and test names with the NYSIIS (New York State Identification and Intelligence System) phonetic code and computed the Fuzz score between the names with partially or fully identical phonetic encodings. For instance, as shown below, we will compute the Fuzz score between the test name 'Jone Rablen' and the database name 'Johnette Rossetti' since they share the part 'JANA' in their phonetic encodings. Similar to the previous two models, the database names with the highest Fuzz scores are taken as the top matches.

	actual name		phonetic encoding
Test Name:	Jone Rablen	→	**JANA** RABLAN
Database Name:	Johnette Rossetti	→	**JANA**TA RASAT

4 Experimental Evaluation

4.1 Data

NER The dataset for NER, provided by our sponsor, includes 221 training and 39 test examples. Each example consists of a news article annotated with entities and their respective labels.

Name Matching For the database names, there are 904 person names and 3261 organization names. For the test names, there are 100 person names 292 organization names. We processed the names by stripping the extra white space.

4.2 Methodology

4.2.1 NER

The NER process consists of data preprocessing, model fine-tuning, inference, post-processing, and evaluation.

Pre-processing The preprocessing phase initiates with tokenization, where the text in our data samples is segmented into tokens. This tokenization aligns with the specific settings of each model. For each word, we assign corresponding entity labels, word IDs, input IDs, and attention mask IDs.

Fine-tuning The preprocessed texts are input into various models, where fine-tuning is conducted by adjusting several hyperparameters to optimize performance. Key hyperparameters include learning rate, dropout rate, weight decay, batch size, and the number of training epochs. We employed grid search to find the optimal hyperparameters.

Inference After fine-tuning, these models are applied to make inferences on the test data.

Post-Processing In the post-processing phase, token-level predictions with label IDs are transformed into readable entity-level predictions, thereby rendering the model outputs into a format suitable for practical evaluation.

Evaluation Our evaluation approach emphasizes two critical aspects: accuracy and efficiency.

- **Accuracy:** We measure accuracy using two distinct F_1 scores.
  - **Exact Match $\mathbf{F_1}$ Score:** This metric is crucial for hyperparameter tuning and requires the model to identify every instance of named entities in the text.
  - **Unique Entity $\mathbf{F_1}$ Score:** Employed for the overall model evaluation and comparison, this score focuses on the model's ability to identify unique entities. It does not penalize for missing repeated instances of the same entity.
- **Efficiency:** Efficiency is a key consideration due to our sponsor's operational need to process millions of news articles daily. In this regard, we assess models based on their inference time on test data. Models that deliver accurate results but with suboptimal processing speeds are deemed less suitable for our high-volume, real-time application context.

4.2.2 Name Matching

To test Name Matching's effectiveness, we compared its outputs against the ground truth set of names. The performance is measured by how accurately it matches the top-n similar names. We implemented Mean Average Precision at K (MAP@k) to measure accuracy. In specific, we measured MAP@1 and MAP@3 for person names and organization names. As for efficiency, we used training time and inference time as metrics.

4.3 Results

4.3.1 NER

Table 1: Comparison of Selected NER Model Performances

Model	Entity $\mathbf{F_1}$	PER $\mathbf{F_1}$	ORG $\mathbf{F_1}$	Inference Time (s)
spaCy-lg	0.60	0.67	0.56	7.08
spaCy-trf (pre-trained)	**0.87**	0.88	0.87	145
BERT-large	0.62	0.63	0.61	**1.64**
BERT-CRF	0.69	0.71	0.64	2.25
XLM-RoBERTa	0.66	0.70	0.63	1.66
XLNet-base	0.76	0.83	0.72	4.41

Table 1 shows the comparative performances of the selected NER models. The metrics shown include the Unique Entity F1 scores for overall entities, PER, and ORG categories, as well as the inference time for each model. These results provide an initial overview of the models' balance between accuracy and efficiency.

4.3.2 Name Matching

Table 2: Comparison of Name Matching Approaches

Method	Per Top1	Per Top3	Org Top1	Org Top3	Training Time (s)	Inference Time (s)
Naive TF-IDF	**0.78**	**0.77**	**0.59**	**0.75**	N/A	3.45 + 32
Clustered	0.68	0.64	0.46	0.61	3.25 + 50.10	**0.97 + 11.02**
Phonetic	0.66	0.64	0.45	0.61	N/A	2.50 + 12.27

Table 2 shows that Naive TF-IDF has the highest accuracy but relatively low efficiency regarding the inference time. The performances of the Clustered and Phonetic methods are satisfactory and are more efficient. It should be noted that while the Clustered method demands additional cluster training time, this training is less frequent, occurring only when the database is fundamentally updated.

4.4 Discussion

NER The results reveal a trade-off between F_1 score and inference time in the NER models. While the spaCy-transformer model achieves the highest F1 scores, its prolonged inference time limits its practicality for high-volume processing. BERT-large emerges as the most efficient model, yet with moderate accuracy. Notably, both our team and our sponsor have a preference for the BERT-CRF and XLM-RoBERTa models, as they exhibit a commendable balance between accuracy and efficiency. This equilibrium aligns well with our operational requirements.

Name Matching As presented in the result section, there is a trade-off between efficiency and accuracy in Name Matching. However, since the provided datasets for Name Matching are small, the difference between the models might be subtle. Considering the potential application on much larger datasets, we also tested our model efficiency on another dataset of person names, choosing 20,000 names as a database and 10,000 names for testing. Naive TF-IDF and Clustered models had similar inference times, approximately 420 seconds, whereas the Phonetic method demonstrated a faster inference, completing in 340 seconds. While all three of the solutions are ready to be deployed in production, the choice depends on business needs.

5 Conclusion

In summary, our project successfully enhanced the machine learning system for credit risk assessment in banking, focusing on NER and Name Matching. We determined that the BERT-CRF and XLM-RoBERTa models offer an optimal balance of efficiency and accuracy for NER. For Name Matching, the Naive TF-IDF method showed the highest accuracy, with Phonetic and Clustered methods also proving effective. Our sponsor plans to adopt and deploy these methods in their future operations. Despite challenges like data scarcity, our project demonstrates the practical application of AI in risk assessment, providing a foundation for further advancements in the field. Future directions include exploring the potential of large language models in NER, and overcoming current limitations of time and resources.

6 Lessons learned

Our project faced two primary limitations. First, data scarcity was a significant challenge, as our sponsor could only provide a limited dataset due to confidentiality concerns. The second challenge was our exploration of Large Language Models (LLMs). We were constrained to small-sized, open-source LLMs for NER tasks due to confidentiality requirements and limited computational resources. Despite those models' accessibility, they underperformed compared to existing NER models.

7 Student contributions

QJ: NER spaCy pipelines; Name Matching Naive TF-IDF
JL: NER BERT-base, BERT-CRF, distilBERT
EW: NER BERT-large, XLM-RoBERTa, XLNet-base
KZ: Name Matching Clustered method & Phonetic method
QJ + JL + EW + KZ: Instructor & Mentor Meetings, Midterm Presentation, Poster, Poster Presentation, Report

References

[1] Milad Baghalzadeh Shishehgarkhaneh, Robert C. Moehler, Yihai Fang, Amer A. Hijazi, and Hamed Aboutorab. Transformer-based named entity recognition in construction supply chain risk management in australia, 2023.

[2] Shuhe Wang, Xiaofei Sun, Xiaoya Li, Rongbin Ouyang, Fei Wu, Tianwei Zhang, Jiwei Li, and Guoyin Wang. Gpt-ner: Named entity recognition via large language models, 2023.

[3] William Cohen and Jacob Richman. Learning to match and cluster entity names. In *ACM SIGIR-2001 Workshop on Mathematical/Formal Methods in Information Retrieval*. Citeseer, 2001.

[4] Ran Ziv, Ilan Gronau, and Michael Fire. Companyname2vec: Company entity matching based on job ads. In *2022 IEEE 9th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–10. IEEE, 2022.

[5] Jiexun Li and Alan G Wang. A framework of identity resolution: evaluating identity attributes and matching algorithms. *Security Informatics*, 4:1–12, 2015.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[7] Fábio Souza, Rodrigo Nogueira, and Roberto Lotufo. Portuguese named entity recognition using bert-crf, 2020.

[8] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale, 2020.

[9] Explosion AI. spacy: Industrial-strength natural language processing, 2023. Accessed: 2023-12-10.

[10] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding, 2020.