Extracting Software Mentions and Relations using Transformers and LLM-Generated Synthetic Data at SOMD 2025

Pranshu Rastogi
Independent Researcher
rastogipranshu29@gmail.com

Rajneesh Tiwari
MS student, CS, Georgia Institute of Technology
rtiwari37@gatech.edu

Abstract

Software is an essential building block of scientific activity, but it often does not receive official citation in scholarly literature. In order to enhance research accessibility and interpretability, we built a system that identifies software mentions and their properties (e.g., version numbers, URLs) as named entities, and classify relationships between them. We fine-tuned Deberta based models for the Named Entity Recognition (NER) task and handled Relation Extraction (RE) as a classification problem over entity pairs. Due to the small dataset size, we employed Large Language Models to create synthetic training data for augmentation. Our system achieved strong performance, with a **65%** F1 score on NER (ranking **2nd** in test phase) and a **47%** F1 score on RE and combined **56% F1 score**, showing significant performance of our approach in this area. Github:- pranshurastogi29/Named-entity-Relation-Extraction-SOMD-2025-ACL

1 Introduction

SOMD 2025[1] shared task focuses on software mention detection. Software which is firmly integrated into the fabric of contemporary scientific inquiry—not just as an experimental and analytic tool but also as a subject of theoretical discussion. In spite of its ubiquity, software regularly fails to receive systematic and formal citation in scholarly papers. Closing this gap calls for automated systems with the ability to detect and comprehend software mentions and their corresponding context in scholarly texts. In this paper, we introduce a system to address this task based on **Named Entity Recognition (NER)** and **Relation Extraction (RE)** methods. Our solution is built upon a heavily annotated dataset of **1,150** sentences (Schindler et al., 2021) of research articles, covering a wide variety of software-related entities and relationships. These are not limited to simple identifiers such as software names, but also encompass more involved constructions including versions, developers, licenses, and usage scenarios. The hierarchical annotation structure and BIO tagging scheme of the dataset allow for fine-grained entity recognition, and the relation annotations record informative relations—version relations and plugin relations—among entities.

For Named Entity Recognition (NER), we fine-tuned Deberta(He et al., 2021) (Base and Large) models on a provided dataset to detect different software-related entities. To enhance generalization, we created high-quality synthetic training data through instruction-guided Large Language Models (such Gemma-2-9b-it (Gemma Team et al., 2024), Mistral-7b-instruct-v0.1 (Jiang et al., 2023), Qwen2.5-7b-instruct(Yang et al., 2024)(Team, 2024).), from template-based on real examples. In Relation Extraction (RE), we cast the problem as a classification task over pairs of entities labeled by the NER model and fine-tunned Deberta (He et al., 2021) and ModernBERT (Warner et al., 2024) based encoders to predict the relationship type between software components. This end-to-end system obtained **65%** score on NER and **47%** for RE and **combined 56% F1 score**, testifying to the subtlety involved in identifying software references and their relationships within scientific texts.

2 Background

Software plays a fundamental role in research across many scientific disciplines, facilitating experimentation, simulation, analysis, and reproducibility. Despite its centrality, software is usually only informally discussed in academic literature and mostly absent of reference citations or proper metadata, which presents issues for subsequent indexing, reuse and reproducibility (Schindler et al., 2021). In order to address these issues, it is now common practice to automate the identification of

[1] https://sdproc.org/2025/somd25.html

Proceedings of the Fifth Workshop on Scholarly Document Processing (SDP 2025), pages 173–181

software mentions in academic writings and the various attributes of that mention. This commonly involves two natural language processing tasks: Named Entity Recognition (NER) and Relation Extraction (RE).

Commonly NER and RE have been looked at independently using pipeline-based approaches which lead to iterative errors, e.g. misidentified entities will lead to misidentified relationships (Zeng et al., 2014); (Zhang et al., 2017). In response to these limitations researchers have begun to shift away from independent modeling tasks to modeling both tasks, in a joint fashion, simultaneously and in turn improving performance and efficiency.

2.1 Related Work

One of the earliest joint extraction approaches was introduced by (Li and Ji, 2014), who proposed an incremental model that simultaneously identifies entities and their relations using shared contextual features. This joint modeling approach demonstrated clear advantages over pipeline systems in terms of accuracy and coherence.

Subsequent studies have leveraged transformer-based architectures to further improve the joint learning of NER and RE. (Wadden et al., 2019) presented a contextual span-based model that utilizes BERT-based embeddings to extract entities and relations jointly within a unified framework. Generative methods have also gained traction in this space. (Huguet Cabot and Navigli, 2021) proposed REBEL, a sequence-to-sequence model that reformulates relation extraction as a text generation task, simplifying the overall architecture and reducing dependency on complex feature engineering. Building upon these developments, (Hennen et al., 2024) introduced ITER, a transformer-based iterative refinement model for joint NER and RE that incrementally improves predictions through multiple passes, leading to state-of-the-art performance.

In the domain of software mention detection, the SoMeSci knowledge graph developed by (Schindler et al., 2021) represents a significant contribution. It provides a high-quality annotated dataset of software mentions from scientific articles, enabling the development and evaluation of machine learning models tailored to this specific use case.

2.2 Dataset or Task Description

The dataset is dominated by a high class imbalance on both relation and entity labels. Of the 2,680 relations in the dataset, only three types—Version of (33.7%), Developer of (23.2%), and Citation of (14.4%) total more than 70% of all instances. When you add URL of and PlugIn of, the top five relation types total almost 85% of the data. By comparison, less common relations such as Extension of and AlternativeName of occur much less often, indicating a long-tail distribution. Fig 2

At the entity level, about 82% of the 32,000 tokens have a non-entity tag ("O"). Among the real entity types, the most frequent ones are Application of (1,761 tokens) and Developer of (1,340 tokens), then come Version of (926 tokens) and Citation of (440 tokens).Table 1

2.3 Input and Output Format

The input consists of scholarly text (e.g., academic paper sentences), and the system outputs both entity-level annotations and inter-entity relations.

Example Input (text.txt): `{ "Input Text" : In this paper , we introduce the latest version of our computational analysis software , Comprehensive Analytical Software Tool ( CAST ) , now upgraded to version 5.2 . }`

NER Tags Output (train.entities.labels.txt): `{ "NER Label and Prediction Format" : O O O O O O O O O O O O O O O B-Application I-Application I-Application I-Application O B-Abbreviation O O O O O B-Version O }`

This shows how entities are labeled using the BIO tagging scheme. For instance, “Comprehensive Analytical Software Tool” is an `Application`, “CAST” is an `Abbreviation`, and “5.2” is a `Version`.

Relation Output (train.relations.labels.txt): `{ "RE Label and Prediction Format": abbreviation of 20 15 ; version of 27 15 }`

This indicates that the entity at token index 20 (“CAST”) is an abbreviation of the one starting at index 15 (“Comprehensive Analytical Software Tool”), and the entity at index 27 (“5.2”) is its version.

2.4 Evaluation Criteria

Submissions are evaluated on two tasks: Named Entity Recognition (NER) and Relation Extraction (RE), using macro-averaged F1-scores. It averages the F1-score of each class without considering the class imbalance. This means that each class is treated equally, regardless of how many instances it has in the dataset. For final ranking, submissions

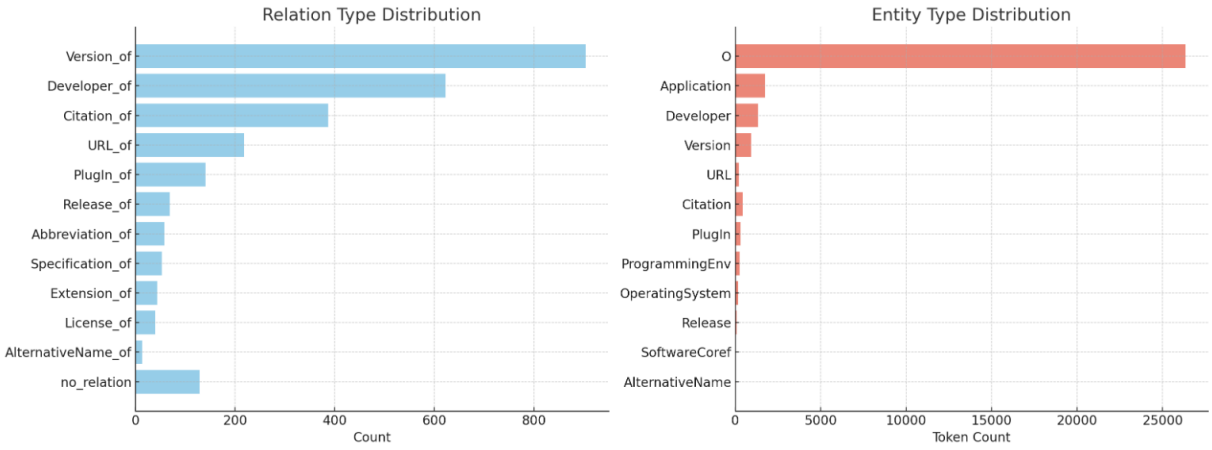

Figure 1: Label distribution in the dataset. Left: Relation Type Distribution. Right: Entity Type Distribution.

Entity Label	Count
O	26344
B-Application	1232
I-Application	529
B-Developer	616
I-Developer	724
B-URL	216
I-URL	6
B-Version	904
I-Version	22
B-PlugIn	211
I-PlugIn	109
B-Citation	382
I-Citation	58
B-Extension	43
I-Extension	7
B-ProgrammingEnvironment	234
I-ProgrammingEnvironment	28
B-OperatingSystem	146
I-OperatingSystem	14
B-Release	69
I-Release	24
B-Abbreviation	58
B-License	43
I-License	54
B-SoftwareCoreference	14
I-SoftwareCoreference	8
B-AlternativeName	14
I-AlternativeName	44

Table 1: Distribution of BIO entity labels in the annotated dataset

are scored using as the **average of the NER and RE Macro averaged F1-scores.**

3 System Overview

3.1 Named Entity Recognition

To address the NER problem, we designed a scalable and resilient pipeline based on Deberta-v3-large (He et al., 2021), an encoder-only transformer with high performance that can be easily well adapted towards challenging NER part of our problem. The model was selected precisely because it can support the difficulties of our dataset — software entities that tend to be fine-grained, vague, and sparsely located throughout the text. more than 80% of the dataset tokens are non-entities, which makes detection even harder. Deberta's disentangled attention and powerful contextual representations enable the model to detect subtle attention patterns between token in such cases. We add a token classification head on top of Deberta for the NER task.

3.1.1 Training Configuration and Implementation Strategy

For Named Entity Recognition (NER) we provided a comprehensive training procedure using a large-scale pre-trained transformer and prepared for and selected optimization and regularization methods. In this section we describe the complete training configuration such as model architecture, hyper-parameters, data, and compute.

Model Architecture: We utilize the Deberta-v3-large (He et al., 2021), which offers strong contextual encoding through disentangled attention and enhanced mask decoding. A token classification head is placed atop the encoder to support the NER task, where each token is labeled based on the BIO tagging scheme for named entity spans. The same backbone is later extended for joint NER and RE learning in downstream settings.

Hyperparameter Settings: The model is fine-tuned using the following configuration:

- **Maximum sequence length:** 512 tokens
- **Learning rate:** 2.5e-5

Named Entity Recognition	Precision	Recall	F1-Score	Support
Abbreviation	0.6667	0.5000	0.5714	12
AlternativeName	0.5833	0.8235	0.6829	17
Application	0.6560	0.6198	0.6374	363
Citation	0.7245	0.7594	0.7415	187
Developer	0.3261	0.7500	0.4545	20
Extension	0.5000	0.1667	0.2500	6
OperatingSystem	0.5000	0.5000	0.5000	2
PlugIn	0.2449	0.6000	0.3478	20
ProgrammingEnvironment	0.8261	0.7917	0.8085	24
Release	1.0000	1.0000	1.0000	10
SoftwareCoreference	1.0000	1.0000	1.0000	3
URL	0.7746	0.7857	0.7801	70
Version	0.6250	0.7292	0.6731	96
Micro Avg	0.6438	0.6904	0.6663	830
Macro Avg	0.6482	0.6943	**0.6498**	830
Weighted Avg	0.6675	0.6904	0.6731	830
Relation Extraction				
Developer_of	0.2344	0.7500	0.3571	20
Citation_of	0.5321	0.7968	0.6381	187
Version_of	0.3901	0.7396	0.5108	96
PlugIn_of	0.1013	0.6154	0.1739	13
URL_of	0.4701	0.7857	0.5882	70
License_of	0.0000	0.0000	0.0000	0
AlternativeName_of	0.6522	0.8824	0.7500	17
Release_of	0.5263	1.0000	0.6897	10
Abbreviation_of	0.5000	0.5000	0.5000	12
Extension_of	0.0000	0.0000	0.0000	6
Specification_of	0.0000	0.0000	0.0000	0
Micro Avg	0.4240	0.7633	0.5452	431
Macro Avg	0.3785	0.6744	**0.4675**	431
Weighted Avg	0.4599	0.7633	0.5675	431

Table 2: Test Phase - Performance metrics for Named Entity Recognition (top) and Relation Extraction (bottom).

- **Learning rate scheduler:** Linear with warmup
- **Warmup ratio:** 10%
- **Weight decay:** 0.01
- **Batch size:** 8 (with gradient accumulation of 16 steps to simulate a batch size of 128)
- **Epochs:** 30
- **Evaluation strategy:** Epoch-based with best-model checkpointing
- **Mixed precision (AMP):** Enabled to accelerate training

These hyperparameters were chosen based on empirical tuning, as well as experience with prior work on transformer-based NER models. Our use of both warmup scheduling and weight decay regularization avoids overfitting, while gradient accumulation enables stable training under memory limitations.

Negative Sampling: Due to the highly imbalanced class distribution (over 80 % were non-entity tokens) we implement negative sampling during training with a downsampling ratio of 0.3. Thus, this approach allows the model to avoid the over-representation of non-entity classes and enhanced the sensitivity of the model in regard to minority classes (i.e. software-related entities).

3.1.2 Synthetic Data Generation for NER Using LLMs

To augment the training data for the Named Entity Recognition (NER) task, we use LLMs to generate synthetic text that maintains annotated entities while introducing significant variation. The general strategy is to pair two samples from the training set and combine their content into a single passage with all named entities from the original texts intact. This is achieved by combining the related entity labels into one consolidated mapping and token definition to maintain explicitly within a carefully designed prompt.The prompt instructs the LLM to paraphrase and merge the two texts both syntactically and semantically, promoting variability in sentence structure and wording without compromising entity coherence. By maintaining same tokens within the generated text,

the method guarantees perfect label recovery, enabling us to label the generated text by mapping each token back to its respective entity type, or as non-entity where there is no match. We generate with samples using LLMs (such Gemma-2-9b-it (Gemma Team et al., 2024), Mistral-7b-instruct-v0.1 (Jiang et al., 2023), Qwen2.5-7b-instruct(Yang et al., 2024)(Team, 2024)). For examples check **Appendix A**

This process not only contributes linguistic diversity to the data set, but also creates more varied context by combining information from multiple samples. The pre-structured prompts form the centerpiece of this operation, which guide the LLM to maintaining both entity accuracy and task cohesiveness. Hence, the resulting data enriches the quality and generalizability of the NER model

3.2 Relation Extraction

For the Relation Extraction we used a contextaware strategy by concatenating the entire input text along with the recognized entities and associated entity types. This allowed the transformer based architectures such as Deberta (He et al., 2021) and ModernBERT (Warner et al., 2024) to utilize both sentence-level and entity-specific upon training.In our early experiments, we finetuned models on a multiclass classification configuration including 12 different relationship types. Trained using this configuration, models with both Deberta (He et al., 2021) and Modern BERT (Warner et al., 2024) with a macro-averaged F1 value of about 15% over the relation-level test set.

3.2.1 Model Architecture and Implementation Strategy

We adapted a transformer-based model with Deberta-v3-large (He et al., 2021) as the backbone encoder. The model has all hidden states from all layers and is instantiated with all dropout components (both hidden and attention) set to 0.1. The model includes a mean pooling layer which aggregates the token embeddings weighted by the attention mask with a linear head that projects to output classes. Given this setup, we were able to train the model efficiently using dual T4 GPU that are publicly available on Kaggle

Hyperparameter Settings: The following configuration summarizes the key hyperparameters used throughout our experiments:

- **Maximum sequence length:** 384 tokens
- **Batch size:** 64 (no accumulation needed)
- **Learning rate:** 4e-5 (encoder), 6e-5 (decoder)
- **Learning rate scheduler:** Linear decay
- **Warmup steps:** None
- **Epochs:** 6
- **Optimizer:** AdamW (β_1=0.9, β_2=0.999, ϵ=1e−6)
- **Weight decay:** 0.01
- **Gradient clipping:** 5000 (max norm)
- **Evaluation frequency:** Every 20 steps

Team	F1 NER	Precision	Recall
TU Graz Data Team	0.68	0.66	0.75
psr123 (Our Team)	0.65	0.65	0.69
Ekbana	0.64	0.67	0.65

Table 3: Comparison of system-level NER (Macro Average) metrics across different teams in Test Phase

3.2.2 Data Augmentation for RE

To address the relation extraction (RE) task, we design a strategy that pairs each sentence with two entity and their corresponding types, formatted as 'entity_type [SEP] entity_text'. For every document, we extract annotated entity pairs and label them with their respective relation types, such as `Developer_of` or `URL_of`. To augment the dataset and introduce harder negative samples, we generate additional entity pairs that do not appear in the original annotations and label them as `no_relation`. For example, if the annotated data contains a pair like `B-Developer [SEP] Software` and `B-Application [SEP] Remote` labeled as `Developer_of`, we add unannotated pairs like `B-Developer [SEP] Software` and `B-Application [SEP] ProctorU` with the `no_relation` Figure2 label. This augmentation process results in a more balanced and challenging training set, enabling the model to better differentiate true relations from coincidental entity co-occurrences. Each training sample ultimately takes the form of the sentence followed by the two entity spans, separated by `[SEP]` tokens: `[document text] [SEP] entity_1 [SEP] entity_2`. This design allows the transformer-based model to leverage full sentence context alongside focused entity information, thereby improving its ability to capture complex relationships in software-related text.

Reproducibility: Our entire pipeline which of NER and RE are implemented using the open

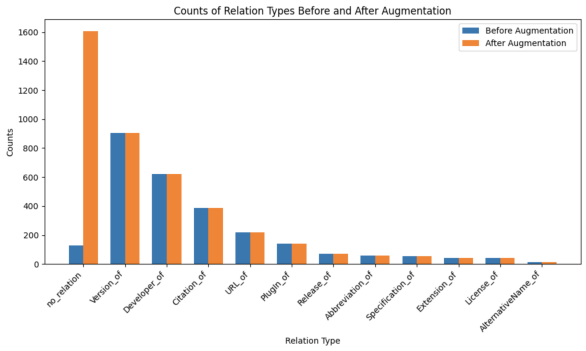

Figure 2: Label distribution after augmentation

source libraries. The design of pipeline is highly modularized, reproducible and easy-to-implement system. Complete configurations details are available in the code release.

Hardware Configuration: To maximize cost-efficiency and experimentation flexibility, our training and experiments used Kaggle's free GPU available environments. In particular, we used the newly released **T4 x2 dual-GPU** environment, which greatly improved the efficiency of training and allowed us to conduct more extensive ablation studies.

4 Performance Analysis

Experimentation Phase. In the experimentation stage, we investigated several modeling methods for the Named Entity Recognition (NER) task, mainly utilizing the Deberta-V3-Large (He et al., 2021) model with a token classification method. We started with a vanilla Deberta-V3-Large model and used a 4-fold cross-validation configuration. In this setup, we observed that the model trained on **Fold 1** achieved an **F1 score (macro average) of 60%** Table 4 on the test set. The other fold-trained models yielded similar validation performance, but none exceeded an F1 score of 60% on the test set.

To enhance generalization, we supplemented our training data with **synthetically generated datasets** produced using different large language models (LLMs), namely Gemma-2-9B-it (Gemma Team et al., 2024), Mistral-7B-Instruct-v0.1 (Jiang et al., 2023), and Qwen2.5-7B-Instruct(Yang et al., 2024)(Team, 2024). Among these, only the synthetic data generated by Mistral-7B-Instruct-v0.1 (Jiang et al., 2023) led to a significant improvement of **5%** in the F1 score. The datasets generated by Qwen2.5-7B-Instruct and Gemma-2-9B-it (Gemma Team et al., 2024) offered a modest **2% gain** Table 4 in a full-training scenario but did not surpass the performance of the Mistral-7B-Instruct-v0.1 (Jiang et al., 2023) based augmentation.

Our best-performing system was the final model: a Deberta-V3-Large (He et al., 2021) trained on the complete original training dataset, **augmented with synthetic data** from Mistral-7B-Instruct-v0.1 (Jiang et al., 2023). This configuration achieved a **macro-averaged F1 score of 65%** Table 4 on the official test set.

We also experimented with XLM-RoBERTa(Conneau et al., 2019) using the same setup, training it on the full dataset along with the additional synthetic data. However, it achieved only a **macro-averaged F1 score of 28% Table 4** on the NER task. This comparatively lower performance further reinforced our choice of and confidence in the Deberta-V3-Large (He et al., 2021) model for this task.

4.1 Performance of NER Test Phase

- **Best Performers:** The model did extremely well in "Release" and "SoftwareCoreference" with 100% Table 2 spot perfect F1-scores. What this indicates is that the model was consistently dependable in identifying these entities.The "ProgrammingEnvironment" entity also demonstrated good performance, which had an F1-score of 80.85% Table 2, and thus ensured that the model was capable of identifying this type as well.

- **Lower Performing Categories:** Other categories were tougher for the model. "Extension" had the worst F1-score of 25%Table 2. Entities "Developer" and "PlugIn" also had lower scores.

- **Overall Performance:** The macro average F1-score over all the NER classes was 65% Table 2, which shows consistent overall performance. Although the model performed well in most entity types, there is still some improvement to be made, particularly in dealing with rare or contextually ambiguous entities.

On the **Relation Extraction** A closer examination showed that more common relation classes were overfitted to and relations between unrelated tokens. To counter this, we employed a data augmentation method with negative examples, i.e., entity pairs having no relation among them, and

Task	Model / Setup	Precision	Recall	F1
NER	Deberta-V3-Large	0.5734	0.6612	0.5993
	Deberta-V3-Large (Full Fit + Mistral-7B)	0.6482	0.6943	**0.6498**
	Deberta-V3-Large (Full Fit + Gemma2-9B)	0.5875	0.6808	0.6199
	Deberta-V3-Large (Full Fit + Qwen2.5)	0.6657	0.6531	0.6215
	XLM-RoBERTa (Full Fit + Gemma2-9B)	0.2775	0.3104	0.2871
RE	Deberta-V3-Large	0.1025	0.4117	0.1543
	Modern BERT-Large	0.0878	0.4228	0.1379
	Deberta-V3-Large (Augmented Data)	0.3785	0.6744	**0.4675**
	Modern BERT-Large (Augmented Data)	0.3473	0.6702	0.4384

Table 4: Performance of different models and training setups on NER and RE tasks (Macro Averaged Scores)

thereby balancing the dataset and robust training. This improvement produced an increase in performance of substantial size, raising our test F1 measure to **47%, a gain of 32%**. Although we could not enter our RE results during the initial test period due to some constraints, we extend a sincere thanks to the workshop organizers for providing an open submission phase, which enabled us to enter our RE model. Our final submission uses a full-fit model trained on the entire relation extraction dataset along with our data augmentation, which gave us our best result of **47%** Table 2 macro-averaged F1.

4.2 Performance of RE Test Phase

- **Best Performers:** "AlternativeName_of" achieved a strong F1-score of 0.7500, demonstrating the model's ability to effectively identify this relation type."Citation_of" and "URL_of" also showed good performance, with F1-scores of 0.6381 and 0.5882, respectively.
- **Overall Performance:** The macro average F1-score for RE is 0.4675, indicating a relatively low overall performance in relation extraction.This result points to challenges in dealing with unbalanced or complex relation types, with a significant opportunity for improvement in handling these cases effectively.

5 Acknowledgments

We would like to thank the organisers of the Software Mention Detection (SOMD) shared task and the Scholarly Document Processing (SDP) workshop for running this competition. We also thank the anonymous reviewers for their insightful and constructive comments, which helped raise the standard of this manuscript considerably.

6 Conclusion

Our approach exhibits decent performance in addressing the task of RE and NER within the software space. In the case of NER, our model has a macro F1-score of **65%** Table 4, and entity categories such as Release and SoftwareCoreference have perfect recall and precision. Still some categories, particularly those that had sparse or confusing examples—such as PlugIn and Extension—were still hard to predict to the model. RE results were not good enough, as macro F1-scores of about 15% because of potential overfitting in over dominant relation classes and incorrect classification of no-relation entity pairs. This was solved by data augmentation process of adding hard negative samples, which boosted RE performance with the macro F1 improved to **47%**, reflecting on the strength of balancing the data and enhancing generalisation. We also investigated synthetic data augmentation with LLMs. Of these, only Mistral-7B demonstrated a definite performance boost (**5%** F1), Our best-performing model Deberta-V3-Large trained on the entire dataset with augmented data produced optimal overall performance, our system produces **combined macro F1** of **56%** Table 4. Overall, our findings highlight the potential of combining pretrained models with synthetic data to tackle complex information retrieval problems in scholarly text. Our hope is that this work will help to further develop better software mention detection systems.

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Thomas Mesnard Gemma Team, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, and et al. 2024. Gemma.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing.

Moritz Hennen, Florian Babl, and Michaela Geierhos. 2024. ITER: Iterative transformer-based entity recognition and relation extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11209–11223, Miami, Florida, USA. Association for Computational Linguistics.

Pere-Lluís Huguet Cabot and Roberto Navigli. 2021. REBEL: Relation extraction by end-to-end language generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2370–2381, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

Qi Li and Heng Ji. 2014. Incremental joint extraction of entity mentions and relations. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 402–412, Baltimore, Maryland. Association for Computational Linguistics.

David Schindler, Felix Bensmann, Stefan Dietze, and Frank Krüger. 2021. Somesci- a 5 star open data gold standard knowledge graph of software mentions in scientific articles. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, page 4574–4583, New York, NY, USA. Association for Computing Machinery.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. 2024. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 35–45, Copenhagen, Denmark. Association for Computational Linguistics.

Appendices

A Synthetic Data Generation for NER

Example of Synthetically Generated Data

Task: Merge and rephrase the following two texts at both syntactic and semantic levels while preserving specific tokens.

Text 1: *Significance was accepted at $P \leq 0.05$, statistical power was 90%, and the analyses were performed in Statistical Package for Social Sciences version 20.0 (SPSS, Chicago, Illinois, USA).*

Text 2: *Microsoft Excel 2007 was used to compile the data and STATA Version 12.0 was used for statistical analyses [28].*

Required Tokens (to be preserved as-is): {`Microsoft`, `Excel`, `2007`, `12.0`, `[28]`, `Statistical`, `Package`, `for`, `Social`, `Sciences`, `20.0`, `SPSS`}

Generated Output Text:
In order to analyze the data, statistical significance was accepted at $P \leq 0.05$ and the statistical power was determined to be 90%. The analyses were performed using **Statistical Package** for **Social Sciences** version **20.0**, commonly referred to as **SPSS**. Data compilation was carried out using **Microsoft Excel 2007**, while further statistical analysis was conducted using STATA version **12.0** (**[28]**).

Example 1: Synthetic NER sample

Example of Synthetically Generated Data

Task: Merge and rephrase the following two texts at both syntactic and semantic levels while preserving specific tokens.

Text 1: *In contrast to the original tool, ggsashimi internally generates an R script which uses the ggplot 2 library [5] for the graphical rendering.*

Text 2: *The pre-processed data files were imported into Agilent Mass Profiler Professional software (version 12.1) for further statistical analysis.*

Required Tokens (to be preserved as-is): {`ggsashimi`, `R`, `ggplot`, `2`, `[5]`, `Agilent`, `Mass`, `Profiler`, `Professional`, `version`, `12.1`}

Generated Output Text:
While **ggsashimi** internally generates an **R** script using the **ggplot 2** library **[5]** for graphical rendering, in contrast to the original tool, the pre-processed data files were imported into **Agilent Mass Profiler Professional** software (version **12.1**) for further statistical analysis.

Example 2: Synthetic NER sample