Entex++: End to End Entity Research Entity Extraction with contribution

Debarghya Datta
IIT Bhilai
debarghyad@iitbhilai.ac.in

Ananya Hooda
IIT Bhilai
ananyah@iitbhilai.ac.in

Vaibhav Arora
IIT Bhilai
vaibhaba@iitbhilai.ac.in

***Abstract*—Understanding significant insights from full-text scholarly papers is critical since it allows us to identify interesting trends, provide insight into research and progress, and create knowledge graphs. However, some of the most important critical insights are only available when considering full-text. Although academics have made tremendous progress in extracting information from short publications, extracting scientific entities from full-text scholarly literature remains a difficult task. Following the work by [14], we extract links to source code, computing resources, programming language/libraries, Objective task, methods used and the top contributions from full-text articles. Our code and data are publicly available at https://github.com/Ananyaiitbhilai/EneRex_plus_plus**

***Index Terms*—Information Retrieval, Semantic Review, Scholarly literature, deep learning**

I. Introduction

The number of scientific scholarly articles published each year is staggeringly high and continues to rise. According to DBLP, 400k articles were published in Computer science(CS)-based research areas in 2020 alone. Extracting key scientific insights from these papers is imperative for understanding emerging technologies, their prevalence, and relationships, and for enabling analysts and policymakers to identify key trends. Information extraction of these entities from large-scale datasets would facilitate the creation of structured knowledge graphs. Existing work builds these knowledge graphs from citation graphs and clusters, coupled with the classification of the papers by various conferences or libraries, such as the CSET Map of Science[3, 10] and the Microsoft Academic Graph [11]. These knowledge graphs can be used to discover clusters of papers belonging to topics of research. Recently, researchers have been attempting to automatically classify documents [2] and discover clusters of paper related to a specific task [1]. Moreover, such knowledge graphs are already in use helping policy makers look into research funding practices in artificial intelligence [10]. We believe that actual scientific entities from the papers would complement the existing citation-based knowledge graphs with more in-depth knowledge and further enrich the information. Motivated by this, we propose an information extraction pipeline for extracting technical entities from the full text of research articles, allowing us to establish trends present in large scholarly databases, particularly in the domain of CS.

II. Problem Statement

This paper presents an automated end-to-end research entity extraction system called EneRex++ to extract key technical facets/entities from the full text of scholarly research articles. Specifically, it extracts six types of entities:

- URLs of source code
- Names of datasets used
- The paper's objective task
- The method used to attempt the objective task
- Computing resources required
- Programming languages and libraries used
- Contributions in the paper

The key problem this paper aims to solve is the lack of structured knowledge graphs and the difficulty in extracting key insights from full text scientific articles. The authors argue that extracting such technical entities can help understand research trends, give insights into R&D, and build better knowledge graphs

III. Earlier Works

Identifying salient scientific facets, or entity types, and extracting them from scholarly articles is an important research endeavor in the information retrieval community [4, 7]. Extraction involves encoding texts, identifying sections where relevant information is present, and then extracting the required information in a structured format. Most research work in this area has traditionally involved working with machine-readable metadata such as title, abstract, etc. [6, 12]. However, many important facets can only be extracted when the full text is taken into consideration since such information is not available in the metadata alone. A recent survey paper concluded that the majority of work on key insight extraction uses abstracts only [9]. They also concluded that the primary challenge of full-text analytics is that the complexity of the manual annotation processes grows as the dataset grows.

The scarcity of large-scale data to train and evaluate our models remains a challenge. Many publicly available, labeled datasets contain annotations for small documents. However, there is a lack of ground truth data for full-text scientific articles. The existing datasets are limited in their capacity. In some cases, ground truth data does not have all the entities used for every paper. Moreover, to the best of our knowledge, three of our facets (links to source code, computing resources, and language/library used) have been only exported in [14] and contributions have not been extracted in prior work, so no ground truth datasets are available for that facets.

IV. Novelty

We have adapted the architecture and model from the EneRex [14] which handles six out of seven facets, and show that their implementation has outperformed the previous SOTA models like Scirex. To increase the Precision and Recall further and handle the seventh facet(Contribution), we have employed the following measures.

- Revamped Dygiepp framework, integrating PLmarker model for NER and RE.
- Extraction of Objective task and methodology delineated through the incorporation of PLmarker.
- Top contributions in a research paper were identified and accuracy assessed through human evaluation.
- Heuristics employed for relation extraction for Top contribution facet were identified.

Comparison of Dygiepp and PL Marker Frameworks

The *Lessons from Deep Learning* paper utilizes the *dygiepp* framework for named entity recognition (NER) and relation extraction (RE) to extract the object task and methodology facets from full-text scholarly articles. In contrast, your case employs the *PL Marker* framework for NER and RE to extract the object task and methodology facets. The PL Marker framework employs a novel packing strategy for spans and span pairs to model interrelations between them, achieving state-of-the-art results on NER and RE tasks.

Key Differences

- **Modeling Approach:** PL Marker uses a levitated marker approach with strategic packing, while Dygiepp employs a dynamic span graph approach.
- **Performance:** PL Marker has shown improved results over Dygiepp on some datasets.
- **Interrelation Modeling:** PL Marker incorporates modeling interrelations between spans and span pairs, which Dygiepp does not focus on.

In summary, the PL Marker framework adopts newer techniques than Dygiepp, achieving state-of-the-art results on multiple information extraction tasks. Applying it for extracting object task and methodology facets seems like a reasonable approach to try.

Architectures of Dygiepp and PL Marker

Dygiepp Architecture:

1) **Token Encoding:** Encodes input tokens using contextual embeddings like BERT or ELMo.
2) **Span Enumeration:** Enumerates candidate spans from the token embeddings.
3) **Span Graph Propagation:** Dynamically constructs a graph with spans as nodes, with edge strengths based on coreference, relations, and events between spans. Iteratively propagates information between connected spans to refine representations.
4) **Multi-Task Classification:** Feeds final span representations into task-specific feedforward networks, making predictions for named entity, relation, and event extraction tasks.

PL Marker (PLMaker) Architecture:

1) **Token Encoding:** Encodes input tokens with a contextual model like BERT.
2) **Span/Span Pair Packing:** Groups related spans/span pairs using packing strategies and packs them into batches with levitated markers.
3) **Contextual Encoding:** Encodes the sequence with packed levitated markers, where levitated markers attend to text and aggregate span-specific information.
4) **Multi-Task Classification:** Feeds marker representations into feedforward networks, making predictions for named entity and relation extraction tasks.

Key Differences in Architectures

- PL Maker uses levitated markers and packing strategies for modeling inter-span relations, while Dygiepp uses a dynamic span graph.
- PL Maker has specific packing strategies for spans vs. span pairs, whereas Dygiepp jointly models multiple IE tasks.
- PL Maker emphasizes computational efficiency more, using markers for efficient representation.

In summary, Dygiepp focuses more on structured modeling of span relations using a graph, while PL Maker uses a packed marker approach to implicitly capture inter-span interactions.

Advantages of PL Maker over Dygiepp

Based on the results reported in the PL Marker paper, PL Maker (PLMaker) achieves better performance than Dygiepp for several reasons:

1) **Modeling Inter-Span Relations:** PL Maker effectively models interrelations between spans by packing related spans and span pairs, which Dygiepp's graph propagation approach may not capture as well.
2) **Computational Efficiency:** PL Maker's marker packing strategies make the model more computationally efficient, allowing it to scale better and utilize larger BERT models.
3) **State-of-the-Art Encoders:** PL Maker benefits from using more recent state-of-the-art contextual encoders like SciBERT and ALBERT xxLarge, resulting in performance improvements.
4) **Task-Specific Design:** PL Maker has specific strategies optimized for NER and RE tasks, while Dygiepp is more generic across IE tasks. This targeted design helps PL Maker perform better on NER and RE.
5) **Explicit Boundary Modeling:** The neighborhood-oriented packing strategy in PL Maker helps it better capture entity boundaries for NER, providing useful inductive bias.

6) **End-to-End Training:** PL Maker is trained end-to-end, allowing marker representations and model components to be jointly optimized for the target tasks.

In summary, by focusing specifically on NER and RE, using the latest encoders, and having packing strategies to efficiently model inter-span relations and boundaries, PL Maker is able to achieve improved performance over the more generic Dygiepp framework. The results highlight the benefits of inductive bias from task-specific architectures.

V. Experimental architecture

a) Facet: Source Code: To identify source code references in a paper, our script begins by isolating sentences with URLs in references or footnotes using spaCy [13] for dependency parsing. The selected sentences undergo universal dependency parsing to extract relation tags (e.g., subject, object, root). A term's contextual location is determined, and sentences meeting criteria (at least two patterns and three occurrences) are considered candidates. The resulting set of sentences is employed to train a facet-specific sentence classifier.

b) Facet: Dataset: The dataset extraction process parallels the source code extraction, differing in the number and templates of patterns. Ten patterns are utilized for dataset mentions, with seven relying on dependency relation tags. The remaining patterns consider the presence of references, footnotes, URLs, or well-known dataset names. The pipeline initiates by extracting sentences containing terms related to dataset materials (e.g., dataset, corpus, database), checking the subsequent five sentences against the patterns. Notably, sentences with the word "dataset" often lack specific details, deferring the mention to later sentences. Dataset entities are identified through heuristic rules, requiring a capital start and potential digit ending for a noun or noun phrase. Scores are assigned using the shortest dependency path, and thresholds are determined empirically for candidate sentence and entity selection. A sentence classifier and NER are trained based on these criteria.

c) Facet: Computing Resources, Programming Language/Library: To capture facets, our data generation script selects sentences with specific seed words from a curated set. The Natural Language Toolkit (NLTK) tokenizes, lemmatizes, and removes stop words from each sentence. Patterns are extracted using part-of-speech (POS) tags and rules, identifying new candidate seed words. Each candidate is assigned a score, and those surpassing a threshold are added to the original set; otherwise, they are discarded. Despite attempting automated entity extraction, results were unsatisfactory, leading to manual annotation of entities within 600 sentences for training the named entity recognition (NER) model for these facets. The entities pertain to computing resources and language/library mentions.

d) Facet: Task, Method, Contributions: To capture objective task and method, we employs a state-of-the-art Named Entity Recognition (NER) model to extract candidate entities. Salience is achieved by pruning entities based on syntactic properties, focusing on the introduction, conclusion, and similar sections of the structured full-text representation. This targeted approach enhances the extraction of more salient entities and relations, increasing the likelihood of identifying mentioned tasks or methods in a document. Notably, a 'USED-FOR' relation often connects a task and method pair in a sentence. Additional connections may exist through 'PART-OF,' 'FEATURE-OF,' or 'HYPONYM-OF' relations with other terms. Following these heuristics and an exhaustive search algorithm aligned with the aforementioned hypothesis, it identifies the most salient objective task, method and contributions for an article. The system further enhances precision by clustering similar entities through fuzzy matching, ultimately providing the final task, method and contribution extractions for an article.

VI. Experimental settings

A. Datasets

We have performed our tests on the following datasets.

- SciERC dataset [8]: 5k+ annotated abstracts
- SciRex dataset [5]: Subset of 438 papers (subset of paperswithcode dataset)
- EneRex dataset [14]: Dataset of 145 AI papers from 2014-2018 cs. Conference

B. Tasks

- Evaluated Facet [Source code, Dataset, Computing Resources, Language/Libraries] on EneRex dataset
- Evaluated Facet [Task, Method] our model on SciERC dataset and SciRex dataset
- Evaluated Facet [Contribution] on SciRex-Test dataset

C. Metrics

For each facet, we calculated precision, recall, and macro F1 for every facet available in a particular dataset.For links to source code, if the extracted URL matched with ground truth URL the extraction is correct. Example of cases for links to source code is given in Appendix D For the dataset, objective task, and method facets, we compare the gold truth with our extracted entity clusters. We used fuzzy string matching with a threshold value of 0.85, empirically determined, to decide if the gold truth entity matched with any element of the clusters. We calculated recall as how many of the gold truth were extracted by the model and precision as correct extractions divided by total number of clusters.

VII. Results and Analysis

- Dataset used: SciRex test split (subset of paperswithcode dataset)
- Preprocessing: Same as Dygie++
- Heuristic: Used the following relation ["FEATURE-OF", "COMPARE", "USED-BY"]
- Evaluation: Human evaluation (manual annotation of 66 research papers)
- Score: 56% accuracy

SciERC Dataset:

The SciERC dataset is specifically designed for Joint Entity and Relation Extraction tasks within scientific literature. It provides annotations for named entities (e.g., Task, Method, Material) and relations between them in diverse scientific articles. Covering a wide range of scientific domains, SciERC serves as a comprehensive benchmark for evaluating models' ability to extract structured information from scientific texts.

Joint Entity and Relation Extraction:

In Joint Entity and Relation Extraction, the objective is to simultaneously identify and classify both named entities and the relationships between them in a single integrated model. Named entities often include proteins, chemicals, processes, etc., while relations capture the interactions and connections between these entities.

Challenges:

Extracting information from scientific literature poses challenges due to complex and domain-specific language. The presence of diverse entity types and relationships in scientific articles further complicates the task, requiring models to have a deep understanding of the subject matter.

Evaluation Metrics:

Common evaluation metrics for Joint Entity and Relation Extraction tasks include precision, recall, and F1 score for both entities and relations. Models are assessed based on their ability to correctly identify entities, assign entity types, and predict the correct relationships between entities.

Applications:

The successful execution of Joint Entity and Relation Extraction on the SciERC dataset has applications in information retrieval, knowledge base construction, and facilitating automated literature review processes. It enables the creation of structured databases that capture relationships between entities mentioned in scientific articles, aiding researchers in extracting valuable insights.

TABLE I
COMPARISON OF MODELS ON JOINT ENTITY AND RELATION EXTRACTION

Rank	**Model**	**Rel. F1**	**Ent. F1**	**RE + Micro F1**	**CS**	**Year**
1	PL - Marker	53.2	69.9	**41.6**	Yes	2021
2	TriMF	52.44	70.17	-	-	2021
3	SPERT.PL	51.25	70.53	-	No	2021
4	SPERT (overlap)	50.84	70.3	-	No	2019
5	Ours: cross-sentence	50.1	68.9	36.7	Yes	2020
6	DyGIE++	48.4	67.5	-	Yes	2019
7	DyGIE	41.6	65.2	-	Yes	2019

We adopt the PL Marker framework (Ye at al., 2021) for named entity recognition and relation extraction to identify the salient objective task and methodology in a research paper. Unlike the previously used dygiepp framework, PL Marker focuses on effectively modeling inter-span relations and interactions.

We first pass the introduction and conclusion sections of each paper through a pre-trained SciBERT model (Beltagy et al., 2019) to contextualize the input word tokens. Next, the candidate text spans in these paper sections are packed into batches using a subject-oriented packing strategy. Here, the subject span is marked using solid markers, while the candidate object spans are marked using levitated markers. This allows jointly modeling relations between a given subject and all its candidate objects.

The packed sequences with levitated markers are input back into the SciBERT encoder. The levitated markers aggregate span-specific contextual information. The updated marker representations are then fed into task-specific feedforward networks, one for named entity recognition and another for relation extraction.

For identifying the paper's main objective task, we applied a heuristic using the predicted inter-span relations - the subject which has the most frequently associated objects is determined to be the salient objective task. A similar heuristic determines the salient methodology.

This strategic packing of contextually encoded spans allows the PL Marker model to capture inter-span interactions and syntactic properties critical for extracting the key objective and methodology in research papers with higher accuracy. The model outperformed prior entity and relation extraction frameworks on benchmark datasets from multiple domains.

TABLE II
PERFORMANCE METRICS FOR ENEREX AND ENEREX++ ON SCIREX DATASET

	EneRex			EneRex++		
Metrics	P	R	F1	P	R	F1
Source Code	N/A	N/A	N/A	N/A	N/A	N/A
Dataset	0.53	0.79	0.63	0.53	0.79	0.63
Objective Task	0.21	0.59	0.31	**0.28**	**0.60**	**0.38**
Method Used	0.17	0.48	0.25	**0.21**	**0.48**	**0.29**
Computing Resources	N/A	N/A	N/A	N/A	N/A	N/A
Language & Library	N/A	N/A	N/A	N/A	N/A	N/A
Macro P & R	0.30	0.62	0.40	**0.34**	**0.59**	**0.43**

VIII. CONCLUSION

We have shown that, it can be expanded with good data and can be expanded for contributions as well.

REFERENCES

[1] Ashwin Acharya, Max Langenkamp, and James Dunham. *Trends in AI Research for the Visual Surveillance of Populations*. 2022. DOI: 10.51593/20200097 (cit. on p. 1).

[2] Arman Cohan et al. "Specter: Document-level representation learning using citation-informed transformers". In: *arXiv preprint arXiv:2004.07180* (2020) (cit. on p. 1).

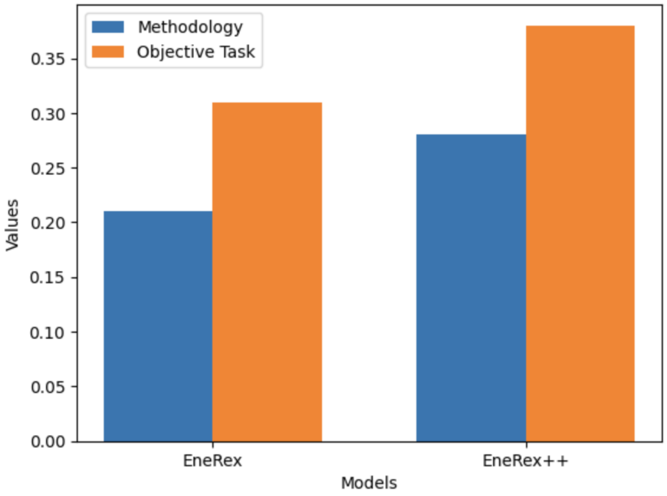

Fig. 1. Comparison of Metrics: EneRex vs EneRex++

[3] James Dunham, Jennifer Melot, and Dewey A. Murdick. “Identifying the Development and Application of Artificial Intelligence in Scientific Text”. In: *CoRR* abs/2002.07143 (2020). arXiv: 2002.07143. URL: https://arxiv.org/abs/2002.07143 (cit. on p. 1).

[4] Sonal Gupta and Christopher D Manning. “Analyzing the dynamics of research by extracting key aspects of scientific papers”. In: *Proceedings of 5th international joint conference on natural language processing*. 2011, pp. 1–9 (cit. on p. 1).

[5] Sarthak Jain et al. “SciREX: A Challenge Dataset for Document-Level Information Extraction”. In: *ACL*. 2020, pp. 7506–7516 (cit. on p. 3).

[6] Shilpa Lakhanpal, Ajay Gupta, and Rajeev Agrawal. “Towards Extracting Domains from Research Publications.” In: *MAICS*. 2015, pp. 117–120 (cit. on p. 1).

[7] Yi Luan. “Information extraction from scientific literature for method recommendation”. In: *arXiv preprint arXiv:1901.00401* (2018) (cit. on p. 1).

[8] Yi Luan et al. “Multi-Task Identification of Entities, Relations, and Coreference for Scientific Knowledge Graph Construction”. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, 2018, pp. 3219–3232. DOI: 10.18653/v1/D18-1360. URL: https://aclanthology.org/D18-1360 (cit. on p. 3).

[9] Zara Nasar, Syed Waqar Jaffry, and Muhammad Kamran Malik. “Information extraction from scientific articles: a survey”. In: *Scientometrics* 117.3 (2018), pp. 1931–1990 (cit. on p. 1).

[10] Ilya Rahkovsky et al. “AI Research Funding Portfolios and Extreme Growth”. In: *Frontiers in Research Metrics and Analytics* 6 (2021). ISSN: 2504-0537. DOI: 10.3389/frma.2021.630124. URL: https://www.frontiersin.org/article/10.3389/frma.2021.630124 (cit. on p. 1).

[11] Arnab Sinha et al. “An Overview of Microsoft Academic Service (MAS) and Applications”. In: *WWW - World Wide Web Consortium (W3C)*. 2015. URL: https://www.microsoft.com/en-us/research/publication/an-overview-of-microsoft-academic-service-mas-and-applications-2/ (cit. on p. 1).

[12] Yuka Tateisi et al. “Typed Entity and Relation Annotation on Computer Science Papers”. In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC’16)*. Portorož, Slovenia: European Language Resources Association (ELRA), May 2016, pp. 3836–3843. URL: https://aclanthology.org/L16-1607 (cit. on p. 1).

[13] Yuli Vasiliev. *Natural Language Processing with Python and SpaCy: A Practical Introduction*. No Starch Press, 2020 (cit. on p. 3).

[14] Raquib Bin Yousuf et al. “Lessons from Deep Learning applied to Scholarly Information Extraction: What Works, What Doesn’t, and Future Directions”. In: *arXiv preprint arXiv:2207.04029* (2022) (cit. on pp. 1–3).