Extraction of Competing Models using Distant Supervision and Graph Ranking

Swayatta Daw, Vikram Pudi
Data Sciences and Analytics Center
IIIT Hyderabad, India
swayatta.daw@research.iiit.ac.in, vikram@iiit.ac.in

Abstract
We introduce the task of detection of competing model entities from scientific documents. We define competing models as those models that solve a particular task that is investigated in the target research document. The task is challenging due to the fact that contextual information is required from the entire target document to predict the model entities. Hence, traditional sequence labelling approaches fail in such settings. Furthermore, model entities themselves are long-tailed in nature, i.e, their prevalence in scientific literature is limited, along with a scarcity of labelled data for training supervised learning techniques. To address the above bottlenecks, we combine an Unsupervised Graph Ranking algorithm with a SciBERT-CRF based sequence labeller to predict the entities. We introduce a strong baseline using the above mentioned pipeline. Also, to address the label scarcity of long-tailed model entities, we use distant supervision leveraging an external Knowledge Base (KB) to generate synthetic training data. We address the problem of overfitting in small sized datasets for supervised NER baselines using a simple entity replacement technique. We introduce this model as part of a starting point for an end-to-end automated framework to extract relevant model names and link them with their respective cited papers from research documents. We believe this task will serve as an important starting point to map the research landscape of computer science in a scalable manner, needing minimal human intervention. The code and dataset is available in the given link :
https://github.com/Swayatta/Competing-Models.

Keywords
NER, Graph Ranking, Distant Supervision, CEUR-WS

1. Introduction

The number of scientific publications in the computer science domain has increased exponentially in the recent past. Hence, it has become increasingly cumbersome for researchers to keep track of the advancement of the research landscape. Often, research papers introduce new models that perform strongly in comparison with the baseline or advance the state-of-the-art. In order to effectively benchmark models and compare their performances, it is important to be able to map the research landscape for similar or related tasks. Papers with Code (Pwc[1]) is a community driven corpus that serves to automatically list models that solve particular subtasks , with links to the scientific research paper that introduced the model. Our aim is to build a similar but automated end-to-end pipeline which detects model names from scientific papers and benchmarks them against other similar models that solve the same task.

In this paper, we introduce the task of extracting competing model names from a research paper. We establish an end-to-end pipeline that extracts all the competing model names from a research paper and links them to their respective citation. While browsing related work for a given task, a researcher has to manually visit every research paper that uses a competing model that is used for the same task. This process is time-consuming if a survey of a research landscape is to be done on a large scale. Our motivation is to automate this process by automatically extracting model names that solve a similar task and linking them to their corresponding cited paper. If executed on a large scale, this pipeline would be able to effectively map the computer science research landscape in an automatic and scalable manner with minimal human intervention.

We introduce a strong baseline for this task by combining an unsupervised document level graph ranking algorithm and a supervised BERT-based sequence tagger to obtain entity model names. Essentially, we treat the relevant keyphrases extracted by the graph ranker as a superset of candidates for the sequence labeller.

We introduce two datasets for this task. For training the supervised sequence tagger, we create weakly supervised distant labels using an external Knowledge Base and unlabelled corpora. We also release a manually annotated dataset for the evaluation purpose of the sequence tagger. For evaluating the entire framework of competing model name extraction, we release another dataset with full paper document level annotation. Furthermore,

Proceedings of the AAAI-22 Workshop on Scientific Document Understanding at the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-22)

[1] https://paperswithcode.com/

we use a simple entity citation linking technique to link the extracted model names with their respective citation in the research document. We believe this task will be a significant step forward towards mapping the research landscape of computer science.

Our contributions can be summarised as follows:

- We introduce a novel approach of treating ranked keyphrases as a superset of sequence labellers for solving this task. To the best of our knowledge, this approach has not been used before in prior research work. We believe this approach can be extended to other similar tasks that require document level contextual information for NER.
- We create an annotated dataset of annotated full papers for evaluation of the pipeline. Previous datasets for sequence labelling in the scientific literature focused only on annotating abstracts of scientific papers [1, 2]. We believe the approach of incorporating full length document information is crucial to capture the entire document context, hence we introduce a full paper annotated dataset for final evaluation.
- We introduce strong baselines while relying only on distantly supervised weak labels to train our sequence labeller. We evaluate the trained model on our annotated evaluation dataset.

2. Related Work

Unsupervised Ranking Algorithms for Keyphrase Extraction: EmbedRank[3] extracts candidate phrases based on POS sequences and uses sentence embeddings (Doc2Vec or Sent2vec) to represent both the candidate phrases and the document in the same high-dimensional vector space and ranks them using cosine similarity with respect to the document embedding. [4] propose WikiRank, an unsupervised automatic keyphrase extraction method that links semantic meaning to text. In graph-based ranking algorithms, candidate phrases are treated as nodes and related candidate phrases are connected by edges. TextRank [5] considered related candidates as co-occurring phrases within a given window. SingleRank [6] added weights to the edges between related candidates.SGRank [7] and PositionRank [8] incorporated statistical and positional heuristics into a graph-based algorithm to obtain ranked keyphrases. MultipartiteRank [9] is an advanced version of TextRank that incorporates positional knowledge in edge weights, leading to state-of-the-art performances over benchmark datasets.

Sequence labelling for Named Entity Recognition: Long tailed entities are named entities which rarely occur in text documents. For these types of entities, the task of Named Entity Recognition (NER) is non-trivial. Recent approaches have aimed at solving the problem of NER using supervised training using deep learning models. However, supervised learning techniques require a large amount of token-level labelled data for NER tasks. Annotating a large number of tokens can be time-consuming, expensive and laborious. For real-life applications, the lack of labelled data has become a bottleneck on adopting deep learning models to NER tasks.

Most scientific named entities can be classified as long-tailed entities because of the rarity and domain-specificity of their occurrence. Recent work on NER in scientific documents has been concentrated around detecting biomedical named entities [10] or scientific entities like tasks, methods and datasets [1, 2, 11]. Some papers like [12] focus on the detection of a single specific entity-type (like dataset names) from scientific documents. Although previous work has focused on identifying methods [1, 2] as named entities, but what constitutes a method can have a significant variance when it comes to human annotated data. The authors [1] report the Kappa score of 76.9% for inter-annotator agreement in the SciERC dataset, which is widely used as a benchmark for scientific entity extraction.

NER has traditionally been treated as a sequence labelling problem, using CRF [13] and HMM [14]. Recent approaches have used deep learning based models [15] to address this task, which require a large amount of labelled data to train. The high cost of labelling remains the main challenge to train such models on rare long tailed entity types, where availability of labelled data is scarce. In order to address the label scarcity problem, several methods like Active Learning [16], Distant Supervision [17, 18, 19], Reinforcement Learning-based Distant Supervision[20, 21] have been proposed. [12] focused on detecting dataset mentions from scientific text and used data augmentation to overcome the label scarcity problem.

3. Motivation

Papers with Code (PwC[2]) is a community driven corpus that serves to automatically list models that solve particular subtasks, with links to the scientific research paper that introduced the model. Our aim is to build a similar but automated end-to-end pipeline that detects model names from scientific papers and benchmarks them against other similar models that solve the same task. We believe the task introduced in this paper (extraction of competing model names from scientific documents) to be a significant step forward towards the whole pipeline.

[2]https://github.com/paperswithcode/paperswithcode-data

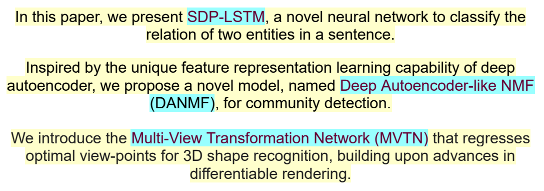
Type	Sentence	Paper Title
Competing	Other transition-based models extend **TransE** to additionally use projection vectors or matrices to translate head and tail embeddings into the relation vector space, such as: **TransH** (Wang et al., 2014), **TransR** (Lin et al., 2015b), **TransD** (Ji et al., 2015), **STransE** (Nguyen et al., 2016b) and **TranSparse** (Ji et al., 2016).	A Novel Embedding Model for Knowledge Base Completion Based on Convolutional Neural Network
Competing	In Table 2, we compare **SCIBERT** results with reported **BIOBERT** results on the subset of datasets included in (Lee et al., 2019).	SCIBERT: A Pretrained Language Model for Scientific Text
Non-competing	**TransE** [4] is a translation based model inspired by *Word2Vec* [16]	On Evaluating Embedding Models for Knowledge Base Completion
Non-competing	(Xie et al. 2016) use *convolutional neural networks (CNN)* to encode word sequences in entity descriptions.	KG-BERT: BERT for Knowledge Graph Completion
Non-competing	To find the hyper-parameters, we used *HyperOpt* (Bergstra et al., 2015), which uses Bayesian optimization.	Tabular Data: Deep Learning is Not All You Need

Table 1
Few examples of competing and non-competing models. The competing models are highlighted in bold, whereas the non-competing models are highlighted in underlined italic.

4. Task Definition

We define **competing** models as model names that attempt to solve the same task as investigated by the target research paper. For example, if a research paper investigates the task of producing knowledge base embeddings, TransR [22] will be a competing model name as it has been introduced by prior research work to solve the same task. If a research paper investigates the task of Question Answering, some competing model names can be T5 model [23] or XL-Net [24], because these are models that have been used to solve this task in prior research work. A non-competing model name would be a model that has not been used directly to solve the same task. We provide a few examples to illustrate the difference between a competing and a non-competing model in Table 1. For the first two examples, the models highlighted in bold are competing models because they directly solve the task investigated in the input research paper. For the third example, TransE is a competing model, but Word2Vec is not. The reason for this is that TransE produces Knowledge Base embeddings directly that aid in Knowledge Base completion (which is the target task in the research paper). But, Word2Vec is a language model that TransE is inspired by, as denoted in the sentence. Hence, it only contributes indirectly to the research task. So, it is a non-competing model. Similarly, HyperOpt, in the last example, is non-competing, as it is an algorithm the authors used for hyperparameter search and is not a model that contributes directly in solving the task investigated in the input research paper.

Our task in this paper is to detect competing model names given an input research document. Also, after extracting the model names, we link the extracted entities with their respective cited papers.


In this paper, we present SDP-LSTM, a novel neural network to classify the relation of two entities in a sentence.

Inspired by the unique feature representation learning capability of deep autoencoder, we propose a novel model, named Deep Autoencoder-like NMF (DANMF), for community detection.

We introduce the Multi-View Transformation Network (MVTN) that regresses optimal view-points for 3D shape recognition, building upon advances in differentiable rendering.

Figure 1: Example sentences with annotated model name entities

5. Annotation Process

We create two datasets for training and evaluation. We annotate sentences from scientific papers as per token-level BIO tagging scheme to evaluate our sequence labeller, which only uses contextual information from an input sentence for sequence tagging. To evaluate the whole pipeline, we provide document-level annotations with full length research papers as input and competing model names as the annotated output. We use two different datasets for a more comprehensive evaluation, as our pipeline uses two stages. The first stage involves extracting candidate keyphrases utilising the entire document level information for keyphrase ranking. The second stage is our sequence labeller that uses sentence level information to find model named entities. We describe the annotation process for the dataset creation for sequence labelling first. Considering our end goal

	Train	Test	Total
# sentences	7800	1000	8800
# tokens	232600	22873	255473
# entities	19012	3647	22659
# unique entities	14748	1249	15672
avg # tokens per sentence	29.82	22.873	29.03
avg # entities per sentence	2.44	3.65	2.57

Table 2
Overall statistics of train and evaluation dataset for sequence labeller evaluation

# total papers	75
# total sentences	34656
# avg sentences per paper	462.08
# entities	622
# unique entities	473
# avg entities per paper	8.29

Table 3
Overall statistics of the document-level annotated dataset for evaluation of the entire pipeline

of automating a high precision framework of extracting related model names and to minimise ambiguity, we consider only named models as model entities for this task . Few examples are - *NMN+LSTM+FT, SpERT (with overlap), B-BOT + Attention and CL loss, SA-FastRCNN, DS-CNNs (Random Walk), Sparse Transformer 59M (strided).* We consider model entities that have a unique name or that are formed by combination of other model names, eg - NMN+LSTM+FT. A few example sentences with model entities are displayed in Figure 1. We define and annotate the test corpus using the standard BIO tagging scheme. Each model entity type was defined to have maximum span length. For Acronyms, we consider the full length entity name instead of the short form acronym if it occurs in text - eg. *DeCLUTR: Deep Contrastive Learning for Unsupervised Textual Representations.* On average, there are 2.5 tokens per entity. We refer Google Scholar and Semantic Scholar to confirm entity types. We randomly selected a subset of abstracts from the arxiv dataset containing 1.7M+ paper data and metadata and randomly select sentences from them to annotate. Also, we randomly sample the DBLP citation dataset containing 1,511,035 papers and obtain the full length versions from the available papers using DOI matching and obtained a random sample of sentences from the full text. We use two different sets of corpus because we want our model to be evaluated on multiple domains within computer science and different publication venues. All the statistics related to our annotated corpus and train set are provided in Table 2

For evaluating the whole pipeline, we annotated full length research papers. We read through the introduction and find out the task the paper solves. Then we browse the entire paper and find all mentions of model names that solve a similar task. The process has a low level of ambiguity because a majority of the model mentions occur in the related work section, citation contexts or experimental results section. It is a standard practice among authors to cite the relevant research paper if they mention any model names from prior research work. Hence, we only consider models that the authors cite to be candidates for competing models. We make sure the labelled entities are model names by referring to Google Scholar and Semantic Scholar. If there is any ambiguity regarding whether a labelled entity is a model name or not, we discard the full paper. To infer if a model is a competing model or not, we find the task or the problem the paper solves. This is usually mentioned clearly in the introduction and the related work section. We label the model entities (that the authors mention as solving a similar problem or task as the original paper) as competing models. To further verify that the claim by the authors is indeed true, we visit the cited research paper and ensure that the model is solving a similar task. Furthermore, we only consider papers where the "competing" relation among models is clear and discard any paper where there is ambiguity regarding this relation. Hence, we ensure ambiguity to be significantly low regarding our annotations. The statistical details about the annotations are provided in Table 3. As we ensure a negligible level of ambiguity, we use only one human annotator (one of the authors in this paper) for our annotation process. We believe the need for multiple annotators for an inter-annotator agreement is insignificant for our task, as a low level of ambiguity is ensured by considering only named models and clearly defined tasks with competing model names.

6. Method

Our entire pipeline has two components. Firstly, we extract all citation sentences from the input research paper. We combine all the citation sentences to create a mini-document. We use a graph ranking algorithm to extract all the candidate keyphrases from this mini-document. This graph ranking algorithm utilises document level information to rank keyphrases. Secondly, we use a sequence labeller for extracting named entities from the positively labelled citation sentences. Lastly, we merge the results of the graph ranker and the sequence labeller to output final competing model entities. In the subsection Sequence Tagging , we provide details about the training process and the model for our sequence tagger. In subsection Graph-Ranking Algorithm, we provide details about the unsupervised graph ranking algorithm for keyphrase extraction.

6.1. Graph-Ranking Algorithm

We use Multipartite Rank [9] as it had proved to be the state-of-the-art among all keyphrase ranking algorithms, performing particularly well on longer scholarly documents. We briefly describe how we use this algorithm for unsupervised keyphrase extraction.

Let C be the set of all citation sentences in a document d. C forms an order set of citation sentences, which is collectively treated as a document. We build a graph representation of C. A set of candidate keyphrases K is extracted from C. The candidate keyphrases K are grouped into topics based on the stem forms of the words they share using hierarchical agglomerative clustering with average linkage. The candidate keyphrases are used to build a multipartite graph, where the nodes are keyphrase candidates that are only connected if they belong to a different topic. The edges between each node is weighted as the inverse of the distance between the two keyphrases K_i, K_j in C. Weight wij is calculated as the sum of the inverse distances between Ki and Kj:

$$w_{ij} = \sum_{p_i \in P(K_i)} \sum_{p_j \in P(K_j)} \frac{1}{p_i - p_j}$$

where $P(K_i)$ is a set of word offset positions of K_i. The first occurring candidates of each topic are promoted more as they capture higher relevance. Weights of the first occurring candidates of each topic is modified according:

$$w_{ij} = w_{ij} + \alpha . e^{\frac{1}{p_i}} \sum_{K_k \in T(K_j) \setminus K_j} w_{ki}$$

where α is a hyperparameter that controls the strength of the weight adjustment, $T(K_j)$ is the set of candidates belonging to the same topic as K_j , p_i is the offset position of the first occurrence of candidate K_i. After the graph is built, a ranking algorithm is then used to order each keyphrase candidate K_i. We adopt the popular TextRank Algorithm [5] for the ranking mechanism. A final set of top ranked keyphrases $\tilde{K}$ is obtained.

6.2. Sequence Tagging

For training our sequence tagger, we only rely on distant labels created using an external Knowledge Base and an unlabelled research text corpus. We also demonstrate that for long-tailed entity types, there is a need to ensure fairer distribution among entity occurrence in order to prevent overfitting, which occurs in the form of the model memorising certain popular entity names. The details about the training set creation is provided in section Training Set Creation with Entity Replacement. The details about the model and the results on the evaluation set is provided in section Distantly Supervised NER Model. The training process overview for the sequence labeller is shown in Figure Training pipeline for the Sequence Labeller.

6.2.1. Training Set Creation with Entity Replacement

We utilise the publicly available Papers with Code (PwC) corpus as a Knowledge Base. We crawl PwC and obtain all the model names occurring in the metadata for each task and subtask. We obtain a total of 14,748 model names. For the unlabelled corpora, we use a total of 227,000 abstracts from arxiv and obtain all sentences (7800) containing a model name mention. We find that the occurrence of some model names is much more frequent in literature (e.g - CNN). Due to the small dataset size and the large imbalance in few entity mentions, the model is prone to overfitting. To mitigate this, we use a simple entity replacement technique, where we find all model entity mentions, and randomly replace them with other names to ensure a fairer distribution. The distribution pre-replacement is shown in Figure 4. We use all 14,748 model entities at least once and limit an entity occurrence to at most 2 in the train dataset, after replacement.

6.2.2. Distantly Supervised NER Model

We treat NER as a sequence labelling problem. Given a sequence of N tokens $X = [x_1, ..., x_N]$, we aim to find an entity which is a span of tokens $s = [x_i, ..., x_j](0 \le i \le j \le N)$ associated with the entity type model name. We formulate this as a sequence labelling task of assigning a sequence of labels $Y = [y_1, ..., y_N]$. The aim of our sequence labeller is to classify each token as a certain entity type as per the BIO tagging scheme.

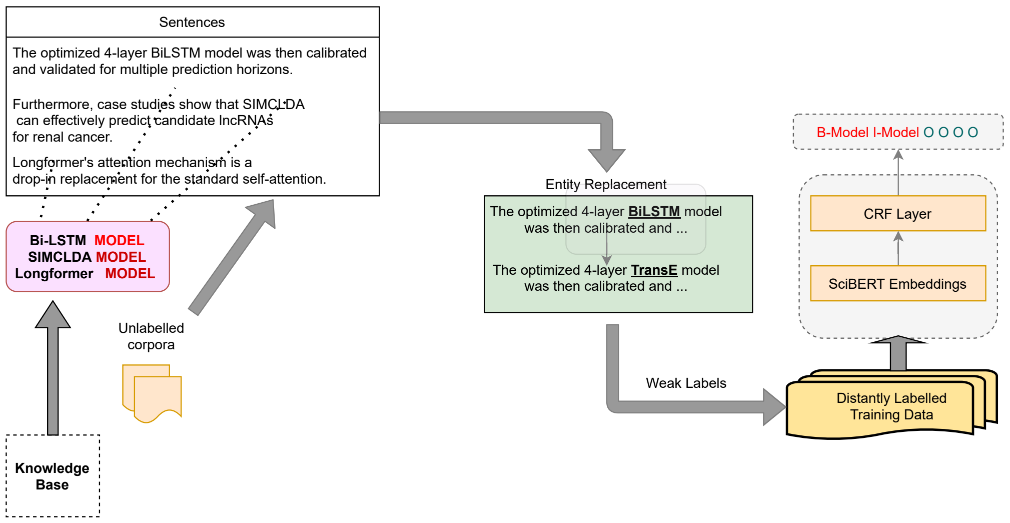
We consider K train sentences denoted as $\{(X_k, Y_k)\}_{k=1}^{K}$ with distant token level annotations. We aim to learn a function $f(X, \theta)$, which can correctly predict the entity labels for a train sentence X_k. We minimise the loss:

$$\theta^* = \arg\min_{\theta} \frac{1}{K} \sum_{k=1}^{K} l(Y_k, f(X_k, \theta))$$

over $\{(X_k, Y_k)\}_{k=1}^{K}$ where θ is the parameter and l is the cross-entropy loss.

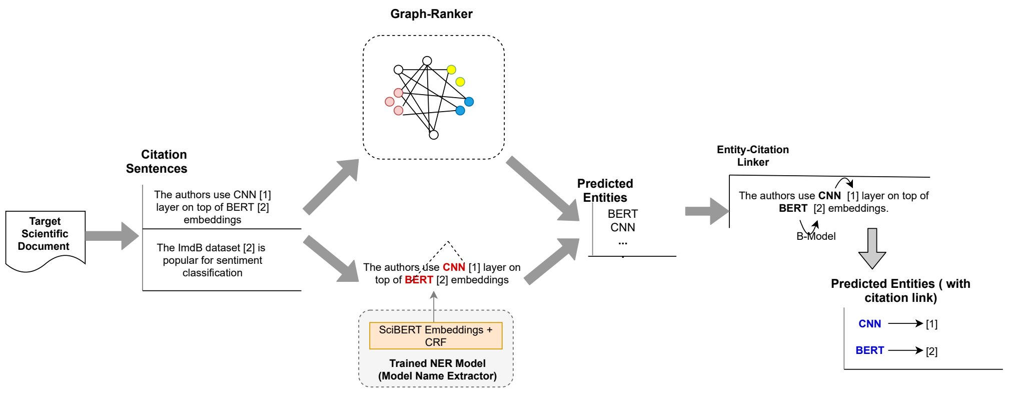
We experiment with multiple baselines which are standard for the sequence labelling process.

- A **BiLSTM + CRF** model where the bidirectional contextual representations are captured by the BiLSTM model, and the resultant representations are passed to the Conditional Random Field (CRF) that produces sequence labels as output.



Sentences
The optimized 4-layer BiLSTM model was then calibrated and validated for multiple prediction horizons.
Furthermore, case studies show that SIMCLDA can effectively predict candidate lncRNAs for renal cancer.
Longformer's attention mechanism is a drop-in replacement for the standard self-attention.
Bi-LSTM MODEL
SIMCLDA MODEL
Longformer MODEL
Unlabelled corpora
Knowledge Base
Entity Replacement
The optimized 4-layer BiLSTM model was then calibrated and ...
The optimized 4-layer TransE model was then calibrated and ...
Weak Labels
Distantly Labelled Training Data
B-Model I-Model O O O O
CRF Layer
SciBERT Embeddings

Figure 2: Training pipeline for the Sequence Labeller



Graph-Ranker
Citation Sentences
Target Scientific Document
The authors use CNN [1] layer on top of BERT [2] embeddings
The ImdB dataset [2] is popular for sentiment classification
The authors use CNN [1] layer on top of BERT [2] embeddings
SciBERT Embeddings + CRF
Trained NER Model (Model Name Extractor)
Predicted Entities
BERT
CNN
...
Entity-Citation Linker
The authors use CNN [1] layer on top of BERT [2] embeddings.
B-Model
Predicted Entities (with citation link)
CNN → [1]
BERT → [2]

Figure 3: Inference Pipeline of the end-to-end framework

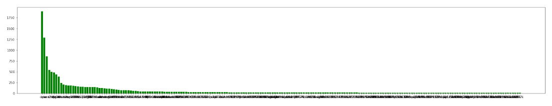


Figure 4: Distribution of entity occurrence frequency in the training dataset pre-replacement

- A **BERT + CRF** model where the contextualised embeddings are captured by a pre-trained BERT base uncased model and passed onto the CRF layer to produce token labels.
- A **SciBERT + CRF** model where the domain specific contextualised embeddings are captured by a pre-trained SciBERT [25] model. SciBERT is BERT-based language model train on large unlabelled scientific corpora using MLM objective. The output embeddings are passed to the linear CRF layer which predicts token labels from contextual representations.

We evaluate our baselines using our evaluation dataset and the results are displayed in Table 4. We demonstrate that entity replacement provides a significant boost in performance for each of these models. The reason is that the model does not memorise entity names for the replaced dataset and uses the context to predict the entity types. The results also prove that standard NER approaches can provide decent results on the evaluation dataset while relying only on weakly labelled training data.

	P	R	F1
BiLSTM + CRF (w/o replacement)	0.205	0.519	0.294
BERT + CRF (w/o replacement)	0.389	0.310	0.345
SciBERT+CRF (w/o replacement)	0.391	0.312	0.346
BERT+CRF (with replacement)	0.575	0.563	0.569
BiLSTM + CRF (with replacement)	0.628	0.631	0.629
SciBERT+CRF (with replacement)	**0.641**	**0.632**	**0.636**

Table 4
Result on Evaluation Dataset

	P	R	F1
TextRank	0.063	0.273	0.098
PositionRank	0.098	0.841	0.162
SingleRank	0.105	**0.863**	0.179
MultipartiteRank	0.123	0.834	0.214
SciBERT-CRF	0.290	0.764	0.420
TextRank + SciBERT-CRF	0.512	0.235	0.322
PositionRank + SciBERT-CRF	0.608	0.661	0.633
SingleRank + SciBERT-CRF	0.609	0.679	0.642
MultipartiteRank+SciBERT-CRF	**0.639**	0.672	**0.655**

Table 5
Result on evaluation on the document level annotated dataset

7. Combining Graph-Ranker and Sequence Tagger

We used the Unsupervised Keyphrase Extraction algorithm to capture only those keyphrases that are most relevant to the document. Although the Sequence Tagger performs well on detecting model name mentions using sentences as the contextual information, we need to capture document level relevance as well to extract competing models. The reason is that not all model name mentions are relevant to the task the given target research paper aims to solve. Hence, we predict only those entities which are common to both top-ranked keyphrases and the extracted model names from our distantly supervised sequence tagger. More formally,

$$Y'' = \tilde{Y} \bigcap \tilde{K}$$

where $\tilde{Y}$ is the set of predicted entities by the sequence tagger, $\tilde{K}$ is the set of top-ranked keyphrases and Y'' is the final set of predicted entities. The entire inference pipeline is illustrated in the Figure 3.

8. Results

We use the evaluation metric of micro-average Precision, Recall and F1-Score to evaluate the performance of the different baselines investigated. We use the full document-level annotated dataset for this evaluation.

We report the results in Table 5. We compare performances of 4 Unsupervised Graph-Rankers for keyphrase extraction: TextRank [5], SingleRank [26], PositionRank [8] and MultipartiteRank [9]. We observe that the recall is highest for SingleRank, as it extracts most of the relevant candidate keyphrases and ensures a high amount of entity coverage. For SciBERT-CRF model, we notice that even though the recall is high, the precision is significantly low. It is due to the fact that although it detects model entity mentions with a good accuracy while considering sentences as contextual information as reported in Table 4, not all models are competing. In order to discern which of the extracted candidate entities are competing models, document context is needed. Hence, we find that combining the two approaches leads to a significant boost in precision while maintaining a decent recall. The highest performance is yielded by the combination of Multipartite Rank with SciBERT-CRF, despite Multipartite Rank having a slightly lower recall than SingleRank. The reason can be attributed to the higher precision of Multipartite Rank among all unsupervised keyphrase extraction algorithms investigated. The higher precision in Multipartite Rank can be attributed to the fact that it aims to select the most relevant phrases by incorporating positional information among edge weights among the candidate keyphrases. Hence, its combination with the sequence labeller yields the highest F1-score among all combinations.

9. Entity Citation Linker

The entity citation linker is inspired from the prior work of [27]. The aim of this algorithm is to link the entities with their corresponding citation. The first step is to obtain all the possible entities and the citations. Then, a closeness score is calculated for each entity-citation pair, which is the string distance between the entity and the citation. Then, we take all the citations and keep only the closest citations per entity. Finally, we take all the entities and keep the closest entity per citation. As demonstrated by the authors, this technique is able to accurately map most entities with their corresponding citations. We use this technique to link all the extracted model entities with their respective citations.

10. Error Analysis

We conduct error analysis for the Unsupervised keyphrase extraction, model entity extraction using sequence labelling, the two-stage framework and entity citation linking. For the keyphrase extraction, the graph-ranker extracts most of the relevant model candidates. However, precision suffers significantly as most models are not keyphrases. Their are multiple keyphrases extracted by the algorithm that are not model names - few examples being domain names like 'Information Retrieval', 'Networking architecture', dataset names like 'SquaD 1.1' or other terms that are relevant to the research paper.

For the sequence labeller, we observe mainly two types of error. First, we notice precision error being introduced into the model because in the training set we consider maximum span of each entity and the occurrence of I-Model (token lying inside a named entity) is relatively high. However, in the evaluation test set of the sequence labeller, the occurrence of singular B-Model entities is massively more. This leads to the misclassification of O as an I by the model. Also, although the model is able to detect model entities reasonably given the sentence as the context, it is unable to discern competing models from unrelated ones. This leads to a significant precision decrease when evaluated on the document-level annotated evaluation set.

Finally, after evaluating the performance of the two-stage pipeline on the document-level annotated dataset, we find that the model often mistakes dataset names for model entity mentions. This can be attributed to the high relevance of datasets with respect to the research paper.

Lastly, for the entity citation linker, sometimes an entity that is associated with a citation marker occurs in the initial part of a sentence and its not the closest to the citation. This can lead to missed out or incorrect linking.

11. Implementation details

We implement the NER model in Pytorch. For tokenization, we use the pre-trained SciBERT tokenizer. The embedding layer is the output from the pre-trained SciBERT model. We include a dropout layer with a dropout probability of 0.5 to reduce overfitting. Learning rate is set to 1e-5 and we train all models for a total of 10 epochs. The output from the dropout layer is passed through a linear layer with input dimension same as the hidden dimension of SciBERT (768). For all Unsupervised Graph Ranker, we use the same hyperparameter settings as specified in their respective papers

12. Conclusion and Future work

We have introduced the task of extraction of competing models from a research paper. We use a novel approach of treating relevant keyphrases extracted using an Unsupervised Graph Ranking algorithm as the superset of a BERT-based sequence labeller. We also use distant supervision to train our sequence labeller. We test our sequence labeller and the entire pipeline on two annotated datasets. We also utilise a simple entitiy replacement technique to reduce overfitting in the sequence labeller. Finally, we use the entity-citation linking technique to link all the extracted model entities with their respective citation. We believe this work to be a significant step forward to map the research landscape of Computer Science in an automated and scalable manner.

References

[1] Y. Luan, L. He, M. Ostendorf, H. Hajishirzi, 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, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 3219–3232. URL: https://aclanthology.org/D18-1360. doi:10.18653/v1/D18-1360.

[2] S. Jain, M. van Zuylen, H. Hajishirzi, I. Beltagy, Scirex: A challenge dataset for document-level information extraction, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020. arXiv:2005.00512.

[3] K. Bennani-Smires, C. C. Musat, M. Jaggi, A. Hossmann, M. Baeriswyl, Embedrank: Unsupervised keyphrase extraction using sentence embeddings, ArXiv abs/1801.04470 (2018).

[4] Y. Yu, V. Ng, Wikirank: Improving keyphrase extraction based on background knowledge, ArXiv abs/1803.09000 (2018).

[5] R. Mihalcea, P. Tarau, Textrank: Bringing order into text, in: EMNLP, 2004.

[6] X. Wan, J. Xiao, Single document keyphrase extraction using neighborhood knowledge, in: Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2, AAAI'08, AAAI Press, 2008, p. 855–860.

[7] S. Danesh, T. Sumner, J. H. Martin, SGRank: Combining statistical and graphical methods to improve the state of the art in unsupervised keyphrase extraction, in: Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics, Association for Computational Linguistics, Denver, Colorado, 2015, pp. 117–126. URL:

https://aclanthology.org/S15-1013. doi:10.18653/v1/S15-1013.

[8] C. Florescu, C. Caragea, PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Vancouver, Canada, 2017, pp. 1105–1115. URL: https://aclanthology.org/P17-1102. doi:10.18653/v1/P17-1102.

[9] F. Boudin, Unsupervised keyphrase extraction with multipartite graphs, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 667–672. URL: https://aclanthology.org/N18-2105. doi:10.18653/v1/N18-2105.

[10] V. Kocaman, D. Talby, Biomedical named entity recognition at scale, CoRR abs/2011.06315 (2020). URL: https://arxiv.org/abs/2011.06315. arXiv:2011.06315.

[11] S. Mesbah, C. Lofi, M. V. Torre, A. Bozzon, G.-J. Houben, Tse-ner: An iterative approach for long-tail entity extraction in scientific publications, in: International Semantic Web Conference, Springer, 2018, pp. 127–143.

[12] Q. Liu, P. cheng Li, W. Lu, Q. Cheng, Long-tail dataset entity recognition based on data augmentation, in: EEKE@JCDL, 2020.

[13] J. Lafferty, A. McCallum, F. Pereira, Conditional random fields: Probabilistic models for segmenting and labeling sequence data, in: ICML, 2001.

[14] H. L. Chieu, H. Ng, Named entity recognition with a maximum entropy approach, in: CoNLL, 2003.

[15] J. Li, A. Sun, J. Han, C. Li, A survey on deep learning for named entity recognition, ArXiv abs/1812.09449 (2018).

[16] S. Goldberg, D. Z. Wang, C. Grant, A probabilistically integrated system for crowd-assisted text labeling and extraction, J. Data and Information Quality 8 (2017). URL: https://doi.org/10.1145/3012003. doi:10.1145/3012003.

[17] X. Wang, Y. Guan, Y. Zhang, Q. Li, J. Han, Pattern-enhanced named entity recognition with distant supervision, in: 2020 IEEE International Conference on Big Data (Big Data), 2020, pp. 818–827. doi:10.1109/BigData50022.2020.9378052.

[18] C. Liang, Y. Yu, H. Jiang, S. Er, R. Wang, T. Zhao, C. Zhang, BOND: bert-assisted open-domain named entity recognition with distant supervision, CoRR abs/2006.15509 (2020). URL: https://arxiv.org/abs/2006.15509. arXiv:2006.15509.

[19] M. A. Hedderich, L. Lange, D. Klakow, ANEA: distant supervision for low-resource named entity recognition, CoRR abs/2102.13129 (2021). URL: https://arxiv.org/abs/2102.13129. arXiv:2102.13129.

[20] F. Nooralahzadeh, J. T. Lønning, L. Øvrelid, Reinforcement-based denoising of distantly supervised NER with partial annotation, in: Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 225–233. URL: https://aclanthology.org/D19-6125. doi:10.18653/v1/D19-6125.

[21] Y. Yang, W. Chen, Z. Li, Z. He, M. Zhang, Distantly supervised NER with partial annotation learning and reinforcement learning, in: Proceedings of the 27th International Conference on Computational Linguistics, Association for Computational Linguistics, Santa Fe, New Mexico, USA, 2018, pp. 2159–2169. URL: https://aclanthology.org/C18-1183.

[22] Y. Lin, Z. Liu, M. Sun, Y. Liu, X. Zhu, Learning entity and relation embeddings for knowledge graph completion, in: AAAI, 2015.

[23] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, Journal of Machine Learning Research 21 (2020) 1–67. URL: http://jmlr.org/papers/v21/20-074.html.

[24] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, Q. V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, in: NeurIPS, 2019.

[25] I. Beltagy, K. Lo, A. Cohan, Scibert: A pretrained language model for scientific text, arXiv preprint arXiv:1903.10676 (2019). URL: https://www.aclweb.org/anthology/D19-1371/.

[26] X. Wan, J. Xiao, Single document keyphrase extraction using neighborhood knowledge, in: AAAI, 2008.

[27] S. Ganguly, V. Pudi, Competing algorithm detection from research papers, in: Proceedings of the 3rd IKDD Conference on Data Science, 2016, CODS ’16, Association for Computing Machinery, New York, NY, USA, 2016. doi:10.1145/2888451.2888473.