Diving for Treasure in a Sea of Scientific Literature: Extracting Scientific Information from Free Text Articles

Aarthi Koripelly
University of Chicago

Zhi Hong (Advisor)
University of Chicago

Kyle Chard (Advisor)
University of Chicago

ABSTRACT

It has become impossible for researchers to keep up with the more than 2.5 million publications published every year. We explore scalable approaches for automatically extracting relations from scientific papers (e.g., melting point of a polymer). We implement a dependency parser-based relation extraction model to understand relationships without the need for a Named Entity tagger, integrate several word embeddings models and custom tokenization to boost learning performance for scientific text.

ACM Reference Format:
Aarthi Koripelly, Zhi Hong (Advisor), and Kyle Chard (Advisor). 2020. Diving for Treasure in a Sea of Scientific Literature: Extracting Scientific Information from Free Text Articles. In *Proceedings of Supercomputing '20: The International Conference for High Performance Computing, Networking, Storage, and Analysis (SC '20).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

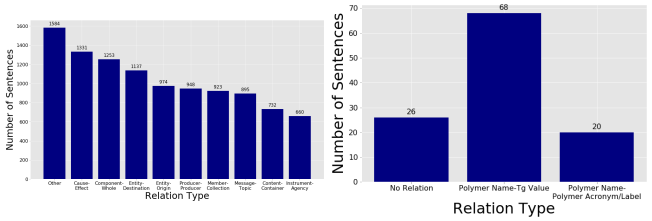
1 INTRODUCTION

The exponential growth of scientific publication [1] is overburdening researchers, who now have to read hundreds of publications just to understand the current state-of-the-art technology. Even with herculean efforts, it is still likely that they will miss important papers or key information included in papers. Crowdsourcing is often suggested as a scalable method for extracting information from publications; however, it is infeasible as most people do not have the necessary expertise to extract information from scientific papers [3]. Scalable and automated methods are required to process papers and to extract important facts, including molecular compounds and the relationships between different compounds such as (aluminum, melting point, 660.3 °C) from the the text

"The **melting point** of **Aluminum** is **660.3 °C**."

2 DATASETS

We focus on two different datasets, the standard SemEval Task 8 relation extraction dataset [2] and a polymer science dataset composed of papers from the Macromolecules journal. Fig. 1 shows the distribution of relations in these datasets. The noun dataset has 8000 training sentences and 2717 testing sentences, each labelled with entities and relation type. The polymer dataset has 300,000 sentences, of which, we manually labelled 114 sentences, each with entities and the relation type. There are three different relationships shown in Fig. 1.



(a) SemEval2010 Task 8 Dataset

(b) Polymer Science Dataset

Figure 1: Dataset Distributions

3 APPROACH

Traditionally, relation extraction consists of five stages: tokenization, part-of-speech tagging, named entity recognition (NER), phrase parsing, and information extraction. We focus here on developing a relation extraction pipeline using a dependency parser rather than using costly NER. A dependency parser analyzes the grammatical structure of a sentence, establishing relationships between "head" words and words which modify those heads. We used a dependency parser as they are useful for extracting relationships between words using only their parts of speech. We used a dependency parser provided by spaCy and customized it through tokenization and word embeddings.

3.1 Default Pipeline

We first explored the accuracy of a dependency parser pipeline using the spaCy's default tokenizer and word embeddings model ('en_core_web_sm'). We then attempted to apply the same pipeline to the polymer dataset; however, it performed poorly due to the difficulty identifying entities in scientific text (e.g., hyphenated and non-dictionary words). To address this limitation we developed a custom tokenizer which combines words with hyphens, degrees signs, and other symbols necessary for understanding polymer notation. We again used the default word embeddings model ('en_core_web_sm'). Finally, we compared these methods against the state-of-the-art ChemDataExtractor [5] The precision, recall, and f1 score of these pipelines is shown in Table 1.

Table 1: Comparison of baseline models

Model	Precision	Recall	F-1
Default pipeline on noun dataset	0.495	0.614	0.548
Default pipeline on poly dataset	0.062	0.156	0.089
Pipeline/custom tokenizer on poly	0.584	0.596	0.599
ChemDataExtractor on poly dataset	0.651	0.587	0.617

SC '20, Nov 17–19, 2020, Atlanta, GA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

3.2 Custom Word Embeddings

To further improve performance we trained custom word embeddings using Skip-gram and Continuous Bag of Words (CBOW) models from the Gensim [4] library. CBOW determines the semantic and syntactic information of a word based on the context in which the word appears. Skip-gram uses a context window around the center word for which it creates a word embedding vector. An example of both is shown in Figure 2.

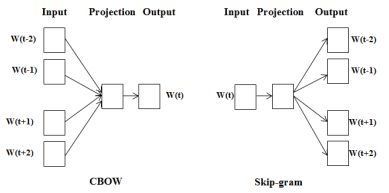


Figure 2: CBOW and Skip-gram Algorithm Overview

4 EVALUATION

We evaluate our approach in terms of accuracy and scalability.

4.1 Accuracy

We used k-fold cross validation to evaluate our model. We set $k = 5$ and took a mean of the evaluation scores to determine the model's performance. We evaluated our CBOW and Skip-gram models using default hyperparameters. The results are shown in Fig. 3. We changed various hyperparameters while training our word embeddings and found that the default CBOW word embeddings gave us the highest F1 score of 0.671.

4.2 Scaling

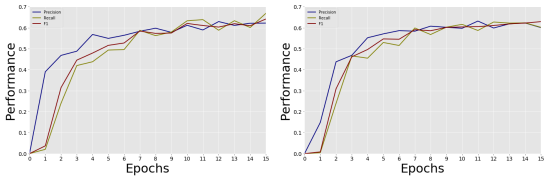
We explored scaling our pipeline on a single node and across many nodes. We used the full Macromolecules dataset of 300,000 sentences and implemented a Python-based program using the Parsl parallel programming library. We executed our pipeline on a campus cluster. Fig. 4 shows our results when increasing the number of cores and number of nodes (with 32 cores per node). For single-node scaling, we were able to reach peak throughput of 103 sentences per second using 32 cores. When scaling across 16 nodes we were able to reach throughput of 590 sentences per second.

4.3 Hyperparameter Tuning

Finally, we evaluated the affect of various hyperparameters on performacne. Our highest F1 score for the CBOW model was 0.671. Our highest F1 score for the Skip-gram model was 0.654. The addition of word embeddings greatly improved the performance of our model (F1 of 0.599). Fig. 5 shows different parameters and the performance of the model given those parameters for both CBOW and Skip-gram.

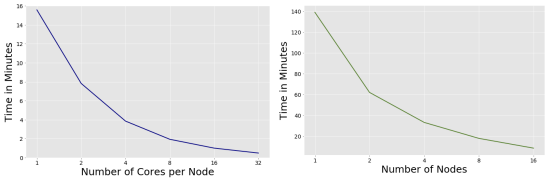
5 SUMMARY

We have presented a depdency parser-based approach for extracting relations from scientific papers. Our approach removes the need for the costly NER phase and can obtain reasonable accuracy with minimal run-time. Our model's best result for identifying polymer



(a) CBOW **(b) Skip-gram**

Figure 3: Performance of Default Hyperparameters on CPU



(a) Scaling Cores on 3000 Sentences **(b) Scaling Nodes on 300,000 Sentences**

Figure 4: Scaling using Parallel Computing

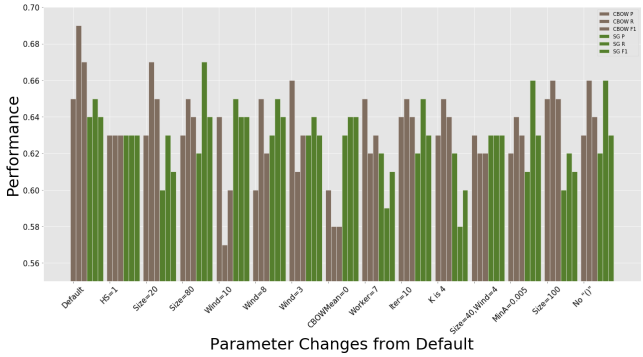


Figure 5: Hyperparameter Tuning

names and relations reaches an F1 score of 0.671- outperforming the 0.617 achieved by ChemDataExtractor. Our model scales well and is able to process almost 600 sentences per second on 16 nodes.

REFERENCES

[1] Santo Fortunato, Carl T Bergstrom, Katy Börner, James A Evans, Dirk Helbing, Staša Milojević, Alexander M Petersen, Filippo Radicchi, Roberta Sinatra, Brian Uzzi, et al. 2018. Science of science. *Science* 359, 6379 (2018).

[2] Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid O Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2019. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. *arXiv preprint arXiv:1911.10422* (2019).

[3] Brigitte Mathiak and Katarina Boland. 2015. Challenges in matching dataset citation strings to datasets in social science. *D-Lib Magazine* 21, 1/2 (2015), 23–28.

[4] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. http://is.muni.cz/publication/884893/en.

[5] Matthew C Swain and Jacqueline M Cole. 2016. ChemDataExtractor: a toolkit for automated extraction of chemical information from the scientific literature. *Journal of chemical information and modeling* 56, 10 (2016), 1894–1904.