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

1 ABSTRACT

- With over 2.5 million publications published every year [7], it is increasingly difficult for researchers to locate important data in publications.
- We explore scalable NLP methods for automatically extracting scientific facts and the relationship between facts
  - E.g., ***melting point*** of ***Aluminum*** is ***660.3 C***
- Approach
  - Use spaCy's dependency parser without costly Named Entity Recognition (NER) component to understand relationships between words
  - Integrate several word embeddings models and custom tokenization to boost learning performance
  - Parallelize pipeline across cores and nodes on a cluster
  - Evaluate the model's run-time and performance, by tuning of hyperparameters for the word embedding algorithms

2 DATA PREPROCCESSING

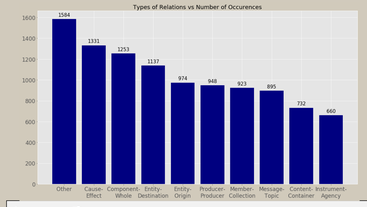
Noun Data

- Dataset provided by SemEval Task 8
- 8000 training and 2717 testing sentences, each labelled with entities, and the relation type.

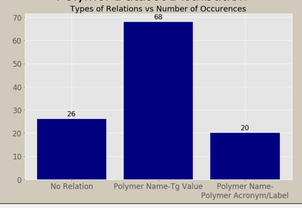
Polymer Data

- Dataset was extracted from the *Macromolecules* journal
- 114 sentences, each labelled with entities, and the relation type.

Noun Dataset Distribution



Polymer Dataset Distribution

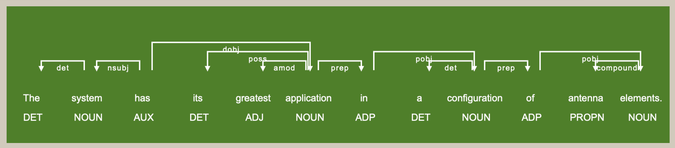


Noun Data

- Input:
  - *"The system as described above has its greatest application in an arrayed <e1>configuration</e1> of antenna <e2>elements</e2>."*
    Component-Whole(e2,e1)
- After preprocessing for spaCy dependency parser training form: o
  - *('The system as described above has its greatest application in an arrayed configuration of antenna elements .', {'heads': [1, 5, 3, 1, 3, 5, 8, 8, 5, 8, 12, 12, 9, 12, 15, 13, 5], 'deps': ['-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', '-', 'Whole', 'pobj', '-', 'Component', 'pobj']})*

Note: Polymer Data was preprocessed in the same manner.

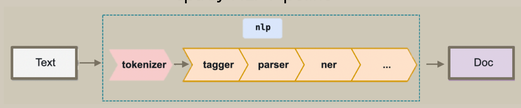
3 DEPENDENCY PARSER



- A dependency parser analyzes the grammatical structure of a sentence, establishing relationships between "head" words and words which modify those heads.
- Useful for extracting relationships between words using solely their parts of speech, rather than relying on named entity recognition.

4 DESIGN/SETUP

spaCy NLP Pipeline



Only the parser was used in the NLP pipeline. All other pipeline components were disabled for all three models.

Model 1: Noun (Baseline)

Pipeline Modifications: Default Tokenizer, Default Word Embeddings using ('en_core_web_sm')

Model 2: Polymer (Built from Baseline)

Pipeline Modifications: Custom Tokenizer, Default Word Embeddings using ('en_core_web_sm')

Custom Polymer Data Tokenization

Example of Tokenization Necessary for Scientific Data:

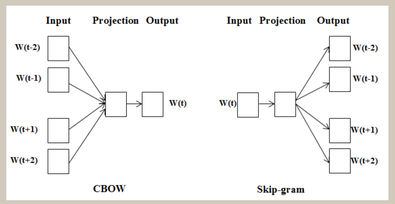
Default:

(The, glass, -, to, -, rubber, transition, of, poly, (, BC, -, co, -, BS,), copolymers, was, investigated, by, DSC, on, melt, quenched, samples, .)

Custom:

(The, glass-to-rubber, transition, of, poly, (, BC-co-BS,), copolymers, was, investigated, by, DSC, on, melt, quenched, samples, .)

Model 3: Polymer with Custom Word Embeddings



Our word embeddings were trained using both CBOW and Skip-gram. Gensim [3] provides a tool, **word2vec**, to train custom word embeddings. **word2vec** is highly customizable and the results of using custom word embeddings is shown in the Results section.

Comparison of Model 1 and Model 2 Performance

Models	Precision	Recall	F1-Score
Model 1: Default pipeline on noun dataset	0.495	0.614	0.548
Default pipeline on poly dataset	0.062	0.156	0.089
Model 2: Pipeline/custom tokenizer on poly	0.584	0.596	0.599
ChemDataExtractor on poly dataset	0.651	0.587	0.617

K-Fold Cross Validation for Polymer Dataset

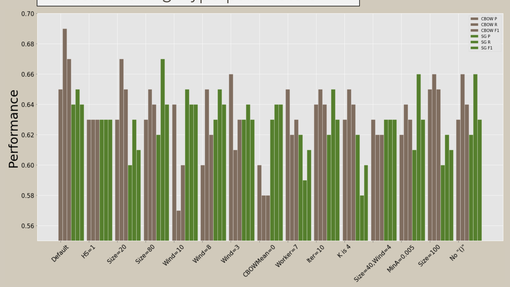
The polymer dataset model was training using k-fold cross-validation as we only had a dataset with 114 sentences. We used a k-value of 5 and took a mean of the evaluation scores to determine the model's performance.

5-Fold Cross-Validation (K=5)

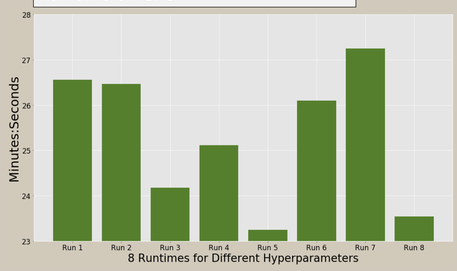
	Dataset				
Estimation 1	TEST	TRAIN	TRAIN	TRAIN	TRAIN
Estimation 2	TRAIN	TEST	TRAIN	TRAIN	TRAIN
Estimation 3	TRAIN	TRAIN	TEST	TRAIN	TRAIN
Estimation 4	TRAIN	TRAIN	TRAIN	TEST	TRAIN
Estimation 5	TRAIN	TRAIN	TRAIN	TRAIN	TEST

5 PERFORMANCE

Model 3: Tuning Hyperparameters



Run-time on CPU



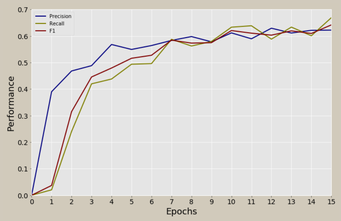
Model 3 Hyperparameter Results

- Highest F1 score for CBOW was 0.671 and Skip-gram was 0.654.
- CBOW outperformed Skip-gram for most hyperparameters

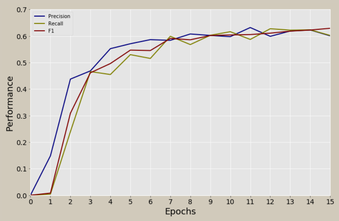
Model 3 Run Time

- spaCY 's code is not optimized for GPU acceleration
- Run time is is scalable based on parameters being used (Default word2vec & Gensim)

Performance Over 15 Epochs for CBOW



Performance Over 15 Epochs for Skip-gram



Model 3 Performance

- CBOW's performance increases at a slower rate than Skip-gram's performance.
- Skip-gram levels out from epoch 8, but CBOW is still increasing in performance at that time and levels out at epoch 12.

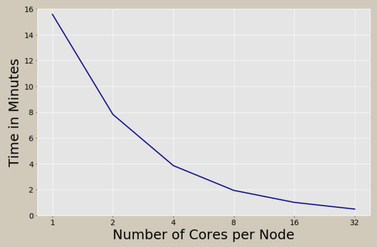
Note: Bottom two graphs show performance for CBOW and Skip-gram with default word2vec hyperparameters.

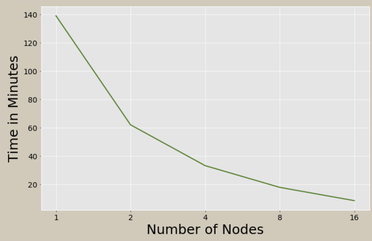
6 SCALING

Parsl

Per Core Scaling

- For scaling we used the larger polymer dataset
- We tested single-node scaling on 1,2,4,8,16, and 32 cores on a test set of 3000 sentences.
- For single-node scaling, our model took 29 seconds on 32 cores.





Per Node Scaling

- We scaled the nodes in the same method while using 32 cores during each run.
- With 16 nodes and 32 cores, our model took 8 minutes and 28 seconds to process a dataset of 300,000 sentences.

7 CONCLUSION

- Traditional methods for extracting scientific facts generally rely on manual efforts. Generally, natural language processors for relation extraction consist of five stages: tokenization, part-of-speech tagging, named entity recognition, phrase parsing, and information extraction. We accomplish this without named entity recognition.
- Our model's best result for identifying polymer names and relations reaches an F1 score of 0.671 - outperforming the 0.617 achieved by ChemDataExtractor [5], the state-of-the-art domain-specific toolkit available in the field of chemistry.
- CBOW generally performed better than Skip-gram.
- We hope that structured data extraction from publications will be beneficial for many applications, such as discovering new molecular pathways and understanding molecular relationships between polymers.

8 ACKNOWLEDGEMENTS

I thank Globus Labs and the Center for Data and Computing at the University of Chicago for making this research possible.

9 REFERENCES

[1] Santo Fortunato, Carl T. Bergstrom, Katy Borner, James A. Evans, Dirk Helbing, Stasa Milojevic, Alexander M. Petersen, Filippo Radicchi, Roberta Sinatra, Brian Uzzi, Alessandro Vespignani, Ludo Waltman, Dashun Wang, and Albert-Laszlo Barabasi. Science of science. Science, 359(6379), 2018.
[2] Zhi Hong, Roselyne Tchoua, Kyle Chard, and Ian Foster. SciNER: Extracting Named Entities From Scientific Literature, 2020.
[3] Brigitte Mathiak and Katarina Boland. Challenges in matching dataset citation strings to datasets in social science. D-Lib Magazine, 21(1/2):23–28, 2015.
[4] Radim Rehu˚rek and Petr Sojka. Software Framework for Topic Modelling with Large
Corpora. In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, pages 45–50, Valletta, Malta, May 2010. ELRA. http://is.muni.cz/ publication/884893/en.
[5] Matthew C Swain and Jacqueline M Cole. ChemDataExtractor: A toolkit for automated extraction of chemical information from the scientific literature. Journal of Chemical Information and Modeling, 56(10):1894–1904, 2016.
[6] Roselyne B Tchoua, Aswathy Ajith, Zhi Hong, Logan T Ward, Kyle Chard, Alexander Belikov, Debra J Audus, Shrayesh Patel, Juan J de Pablo, and Ian T Foster. Creating training data for scientific named entity recognition with minimal human effort. In International Conference on Computational Science, pages 398–411. Springer, 2019.
[7] Mark Ware and Michael Mabe. The STM report: An overview of scientific and scholarly journal publishing. Technical report, International Association of Scientific, Technical and Medical Publishers, 2015.