WORD BASED TEXT PARSER AND ROUTE EXTRACTOR FOR CHEMISTRY ABSTRACTS

A PREPRINT

Yannan Yuan
University of Pittsburgh
School of Computing and Information
Pittsburgh, PA 15213
yay75@pitt.edu

Langkun Chen
University of Pittsburgh
The Kenneth P. Dietrich School of Arts and Sciences
Pittsburgh, PA 15213
lac214@pitt.edu

Runbo Jiang
University of Pittsburgh
Swanson School of Engineering
Pittsburgh, PA 15213
ruj11@pitt.edu

December 8, 2019

ABSTRACT

To help scientists in chemistry field to get information from papers more conveniently, we trained a word-based model using data mining, natural language processing and machine learning techniques to parse the text and extract the synthesis information for a synthesis-related chemical paper automatically. We created a pipeline for our model, which includes big text data retrieval, paragraph classification, feature engineering, word label prediction and information extraction so that our model got a great performance on word prediction and extraction.

1 Introduction

Data mining and machine learning methods take more important roles on nature science like chemistry. One aspect is learning from text of synthesis-related chemistry abstracts. Due to the demand of helping scientists spend less time on reading papers, a data mining tool that can 'read' the paper and get the information is needed. Inspired by Kim's work [1], in our work, we build a word-based model trained from large amounts of synthesis abstracts retrieved from publishers for text processing first. Text parser and extractor, which are the main components of our model, can then parse the text and extract the important synthesis information automatically.

2 Methods

Data preprocessing, feature engineering, and machine learning models used in this work will be covered in this section.

2.1 Data Preprocessing

Data we used in this study are words in abstracts of chemistry and the preprocessing consists of two parts, data retrieval and paragraph classification. For the data retrieval, an ArticleDownloader which can automatically retrieve text based on publishers' API was created to get the paper's abstract from Elsevier. Since the abstracts are open-source, it is safe for us to retrieve.

For all of the abstracts, we only need those that are related to chemical synthesis. So a simple Support Vector Machine [2] paragraph classifier was trained on part of abstracts data to predict whether this paragraph is related to chemical

synthesis. Then the well-trained classifier was applied to rest of our data to get more synthesis abstracts for further training.

2.2 Feature Engineering

Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work. In this work, feature engineering includes the processes of word embedding, applying heuristic rule, and manual labeling. First, each paragraph was tokenized to break them up into individual words and these tokens are then used as the input. Then, we manually labeled all words by our defined categories.

Features were created based on words and word embedding was used as our main features. A word embedding is a low-dimension vector representation of words in which similar words are mapped to points close together (by Euclidean distance), whereas dissimilar words will be further away. A common approach in word embedding is the Word2Vec algorithm [3], due to its re-trainable property.

Some heuristic features, such as the part-of-speech and chemical entity were involved. ChemDataExtractor [4] and NLTK [5] were used for creating these rules. ChemDataExtractor is a language model specifically trained on chemical text was used to identify part-of-speech and chemical entities in our text. NLTK, a general natural language processing toolkit was used to identify stop words (Common expression in English which is usually not useful for training, like 'a', 'the', etc). Eventually, these heuristic features were transformed into binary features.

2.3 Models

2.3.1 Training Models

Support Vector Machine, Random Forest and LightGBM were used as training models in our work.

Support Vector Machine [2] Classifier is an approach based on "maximum margin classifier", which use a line separator so that different labels will have maximum margin. To handle the non-linear classification problem, support vector machine uses a kernel trick, by using a kernel function to easily calculate the high dimension representation of data so that non-linear samples can be well separated on high dimension space.

Random Forest [6] is an ensemble algorithm. It use a bagging method: each time it re-samples a subset from data and uses a decision tree to fit them, and combine all the decision trees. For a classification problem, The final result is based on the voting of the results of several decision trees. By involving large amounts of model, it can reduce overfitting.

LightGBM [7] is also an ensemble model. It uses a boosting method called gradient boost decision tree. Besides, it also involves gradient one-side sampling and leaf-wise split approach rather than depth wise, which makes it much faster than other GBDT algorithm like XGBoost.

2.3.2 Hyperparameter Tuning with Hyperopt

Hyperopt [8] is a hyperparameters tuning tool. Unlike GridSearch, Hyperopt keeps track of past evaluation results and uses Bayesian optimization to search the best hyperparameters in parameters space. The model with best performance would be tuned by Hyperopt.

2.3.3 SHAP Explainer

Model interpretability is a priority in today's data science community. SHAP [9] was used in our work to explain the output of our model. SHAP uses Shapley value to measure the importance of variables. Compared with feature importance, SHAP has two advantages: one is it can show the feature importance for every sample, the other is it can identify the direction of variable influence.

3 Results and Discussion

3.1 Data Preprocessing

In data retrival, we retrieved more than 10000 chemical abstracts and chose part of them for paragraph classification. For our paragraph classifier, We created 152 features for each paragraph, 100 features are word count features which record the count of the most frequent words in whole data, and 52 features are based on heuristic rules, like the length of paragraph and specific key words.

We used Support Vector Machine classifier to train on 792 paragraphs, it gave us 0.99 and 0.97 AUC scores on training and test set respectively, which is an excellent result.

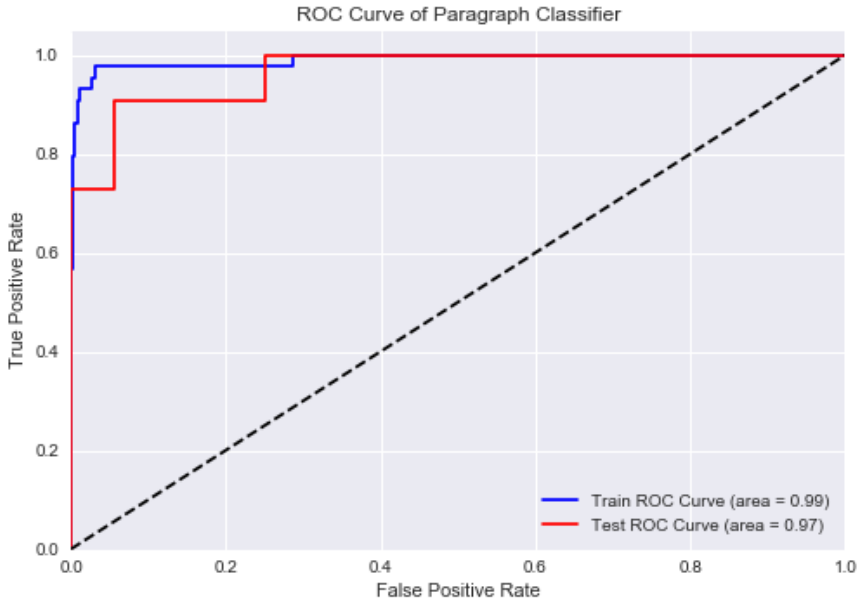


Figure 1: ROC Curve of Paragraph Classifier.

3.2 Feature Engineering

Based on our knowledge, we labelled the words by our pre-defined 13 categories manually, the categories are shown in Table 1.

Label	Label category
0	Null
1	Reactant
2	Product
3	Genetic material
4	Condition
5	Apparatus
6	Operation
7	Synthesis category
8	Condition unit
9	Amount unit
10	Number
11	Condition description
12	Amount description

Table 1: Categories used in manual labeling

T-SNE was used to visualize our Word2Vec vectors, as shown in Figure 2. similar words such as Sulfur compounds are clustered in vector space.

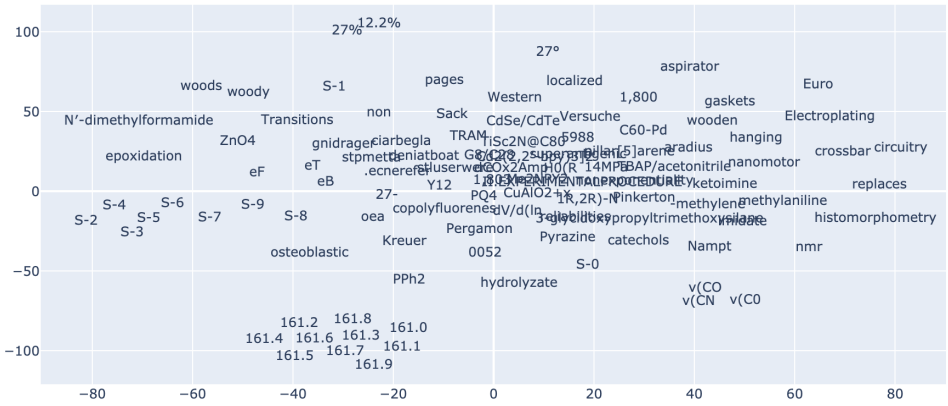


Figure 2: Word2Vec TSNE Result.

Heuristic rules were also used as our features, which are shown in Table 2. These rules were created based on general characteristics of grammar and text, or the result of other language models.

No.	Heuristic Rules
1	Part of speech
2	Chemical entity
3	Whether including number
4	Whether including character
5	whether including stop words
6	length of word

Table 2: Heuristic Rules Features

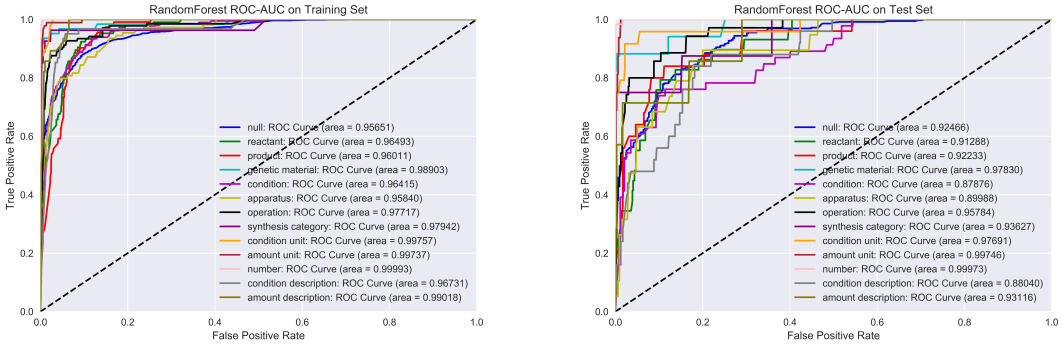
3.3 Model Evaluation

We trained three models for label prediction and the scores of each model are shown in Figure 3.

Models	Accuracy		Micro-AUC	
	Train	Test	Train	Test
SVM	0.671	0.671	0.977	0.958
RF	**0.766**	**0.740**	**0.986**	**0.969**
RF Hyperopt	**0.744**	**0.725**	**0.983**	**0.968**
LightGBM	0.806	0.750	0.989	0.967

Figure 3: Performance of Models.

Based on Accuracy and AUC score, Random Forest is our best model, we then used Hyperopt for hyper-tuning, the result became less overfitting, with accuracy equals to 0.744 and 0.725, as well as AUC score 0.983 and 0.968 on training and test set respectively. The ROC-AUC curve was plot in Figure 4:

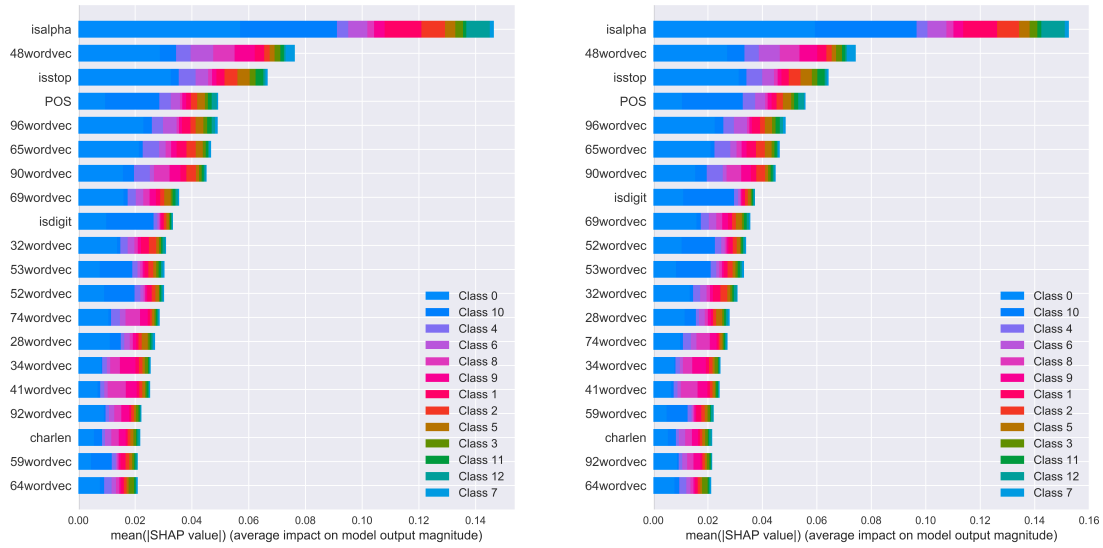


(a) Training Set (b) Test Set

Figure 4: ROC-AUC Plot for Random Forest

3.4 SHAP

SHAP summary plot for training and test set are shown in Figure 5. In general, heuristic features take more effect for label prediction than word embedding in our model, especially the 'isalpha' variable which is very decisive for our model.



(a) SHAP Summary Plot for Training Set (b) SHAP Summary Plot for Test Set

Figure 5: SHAP Summary Plot

3.5 Extractor

By using our model on a new text or sentence, our model can predict the word label correctly and the extractor can highlight the most important information for a chemical synthesis like reactant, condition and apparatus, which can help researchers to get the information of a paper clearly and conveniently.

Word	Zn(NO3)2•6H2O	0.297g	1	mmol	were	dissolved	in	70	mL
Label	reactant	null	number	amount unit	null	operation	null	number	amount unit
Word	of	deionized	water	and	sealed	in	the	steel	autoclave
Label	null	null	condition	null	null	null	null	null	apparatus
Word	and	heated	into	120	˚C	After	the	reaction	the
Label	null	operation	null	number	condition unit	null	null	null	null
Word	autoclave	was	cooled	to	room	temperature			
Label	apparatus	null	operation	null	condition	condition			

Figure 6: Extracted Results

4 Future work

This study has been mainly focused on the use of several common models for parsing text and extract route of chemistry abstracts. However, there are still some ideas could be tested in the future:

1. Build context relevant models. The models we used in this study only focused on isolated words. For a better performance on text parser, a context relevant model like recurrent neural network can be taken into consideration.

2. Find better rule features in feature engineering. The more specific and professional the rules, the better the result.

3. Labelling strategy should be considered more. It is better to ask chemistry exports to label the word and others to train the model, so that it can prevent inherent bias.

References

[1] Edward Kim, Kevin Huang, Alex Tomala, Sara Matthews, Emma Strubell, Adam Saunders, Andrew McCallum, and Elsa Olivetti. Machine-learned and codified synthesis parameters of oxide materials. *Scientific data*, 4:170127, 2017.

[2] Bernhard Schölkopf, Alexander J Smola, Francis Bach, et al. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[3] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[4] 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.

[5] Edward Loper and Steven Bird. Nltk: the natural language toolkit. *arXiv preprint cs/0205028*, 2002.

[6] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[7] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pages 3146–3154, 2017.

[8] James Bergstra, Daniel Yamins, and David Daniel Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. 2013.

[9] Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. Explainable ai for trees: From local explanations to global understanding. *arXiv preprint arXiv:1905.04610*, 2019.