INTRODUCTION

Text mining refers to the application of text-mining techniques to the unstructured text to extract useful information. The volume of published text is expanding at an increasing rate and there is an exponential growth in the scientific publications recently. Researchers have a hard time in figuring out the main concepts of the publications, find material for their reference and the relatedness between papers. In order to gain domain specific knowledge, they have to read huge amount of papers in a short period which is quite impossible. Extracting information manually from the literature is extremely time-consuming and the explosion of information in recent years has made this task almost impractical. As more and more text becomes available a great interest can be observed in extracting useful information hidden in them. Emerging trends of NLP technologies can address the above problem as we are in need of intelligent processing of scientific papers such that they can do information extraction, identifying concepts and recognizing the semantic relation between them.

SemEval 2018 Task 7

Task 7 of the SemEval competition addresses the above mentioned problem: *Semantic relation extraction and classification is to improve the access to scientific literature through NLP technologies.* They have proposed mainly two types of subtasks which are further divided into subtasks.

1) Identifying pairs of entities that are instances of any of the six semantic relations (extraction task)
2) Classifying instances into one of the six semantic relation types (classification task)

Subtask 1: Relationship classification.

This is further divided into two tasks: classification on clean data and classification on noisy data.

Relation classification on clean data is performed on data where entities are manually annotated where entities represent domain concepts specific to NLP, while high-level scientific terms such as “experiment”, “hypothesis”. Relation classification on noisy data is identical to the previous task, but the entities that are annotated contain noise.

Subtask 2: Relationship extraction and classification.

This task combines the extraction task and the classification task. Relation instances needs to be classified into one of the following six relation categories: USAGE, RESULT, MODEL, PART_WHOLE, TOPIC and COMPARISON. Relationship is considered between two entities X and Y. All relations except COMPARISON are asymmetrical. Directionality of the relation (i.e. whether x is before Y or after Y in the context) has to be taken into account when doing the classification.

For the project code drop 1, we partially implemented subtask 1.1 of the given task, i.e. classification on clean data. For the project code drop 2, we fully implemented subtask 1.1 with an enhanced set of features and worked on the directionality of the relation during classification. And we used word2vec algorithm for feature extraction in the subtask 2.

Table 1 Summary on the relation categories of the SemEval task 7.

USAGE	RESULT	MODEL_FEATURE	PART_WHOLE	TOPIC	COMPARE
X is used for Y X is a method used to perform a task Y X is a tool used to process data Y	X gives as a result Y (where Y is typically a measure of evaluation) X yields Y (where Y is an improvement or decrease)	X is a feature/an observed characteristic of Y X is a model of Y X is a tag(set) used to represent Y	X is a part, a component of Y X is found in Y Yisbuilt from/composed of X	X deals with topic Y X (author, paper) puts forward Y (an idea, an approach)	X is compared to Y (e.g. two systems, two feature sets or two results)

METHODOLOGY

In this section, we describe the proposed NLP and machine learning based on relation extraction and classification method which classifies the relations into one the given six relations. The main steps of the proposed approach can be summarized as shown in Fig. 1. When an abstract with relations is given as the input into the model, first all the sentences are segmented. Then, the model classifies each relation into one of the six relations through feature engineering and a learned model.

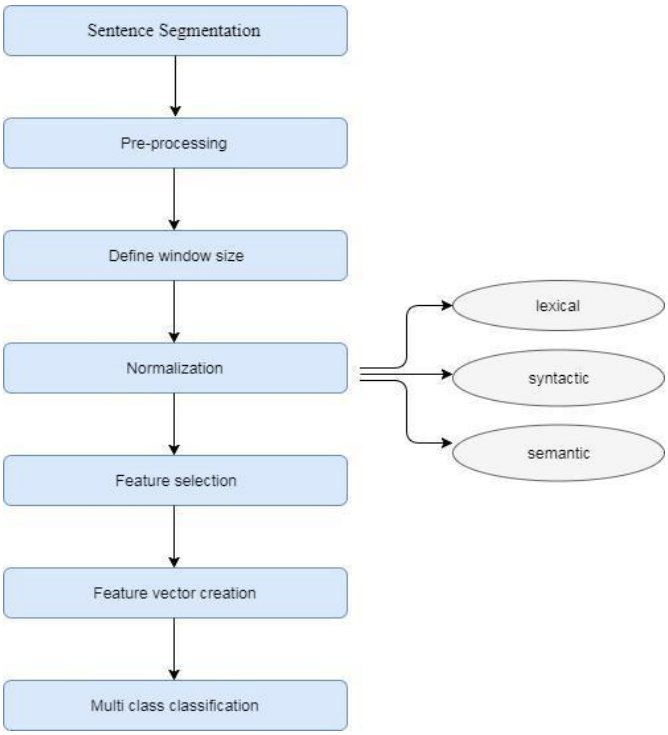

Fig. 1 General pipeline of the proposed approach

A. Sentence Segmentation

Abstracts are identified from the dataset and the sentences with the relations are extracted separately.

B. Pre-processing Steps

Some of the existing NLP techniques and tools are used for preprocessing. Preprocessing is performed as follows.

- All letters are changed to lowercase.
- Special characters and punctuations are removed.
- Stop words are removed
- Lemmatization / stemming is performed

Stop words are common words of the language that do not contribute to the semantics of the documents and do not contain any significance but has a high frequency. They are usually filtered out during search queries to prevent returning vast amount of unnecessary information. Lemmatization is the process of grouping together the different forms of a word and return the base or dictionary form of a word so that they can be analyzed as a single word.

C. Define window size

Prior to features selection, for each relationship a window size is defined. It determines the number of words to choose on either side of the entities

D. Normalization

All sentences in the text corpus are preprocessed in order to normalize the corpus as well as to simplify the feature selection. Lexical normalization includes n-grams and TF-IDF (term frequency-inverse document frequency) calculations. N-grams are used to develop bigram or trigram model in the relation. And TF-IDF is used to evaluate how important a word is to the abstract in the dataset. POS tagging and parsing are used in syntactic normalization. Word2vec is used in semantic normalization during feature extraction.

E. Feature selection

The crucial part of the pipeline is feature selection and feature vector generation. During the feature selection, a representative set of features is computed for each relation. Features used in building the proposed model is listed the section below.

F. Feature vector generation

In the final step of the proposed model, a feature vector is generated using the selected feature for each relation as shown in fig 2.

Fig. 2 Generated feature vector

G. Multi class classification

The generated feature vector is used to train a classifier which classifies the relation into the given six categories. Following classifiers are used: SVM, Decision tree.

Evaluation metrics are used to evaluate the performance of the classifier as requested by the SemEval task. Following are the evaluation metrics used:

1. Subtask 1
    a. Precision, Recall, F1- measure
    b. Macro- average and Micro-average
2. Subtask 2
    a. Extraction : Precision, Recall, F1-measure
    b. Classification : Precision, Recall, F1-measure, Macro- average and Micro-average

DATASET

Our system uses two datasets provided by the SemEval task 7. Dataset include abstracts of papers from the ACL Anthology Corpus with pre-annotated entities that represent concepts. The dataset provided initially is divided into two subsets: training set and test set. Dataset is in XML format containing abstracts from the articles with annotated entities and a text file containing type of relations for some entities. Training set includes 30 abstracts containing 479 entities and 93 annotated types of relations between entities. And test set includes 10 abstracts containing 131 entities and 25 annotated types of relations between entities.

Due to the changes of our SemEval task a new dataset was provided which has the same format but only training data set was given. And it includes 350 abstracts, 5256 entities and 1228 annotated relations.

TASKS

Subtask 1

This task mainly focuses on the categorization of the relations. Features selection is the crucial step in this task and following are the features are selected.

- F1 - Word distance between tags after lowercasing and stop words removal
- F2 - POS tag of the last word in the entity sequence
- F3 - Rule based extraction for boosting prediction of the less represented classes
- F4 - number of words before Entity1 (E1) - with / without stop words
- F6 - number of words after Entity 2 (E2) - with / without stop words
- F8 - word before Entity1 (E1)
- F9 - word after Entity 2 (E2)

- F10 - POS of the word after Entity 2 (E2) - with / without stop words
- F12 - POS of the word before Entity1 (E1) - with / without stop words
- F14 - POS of the word after Entity1 (E1) - with / without stop words
- F16 - POS of the word before Entity 2 (E2) - with / without stop words
- F18 - Bigram of first word after Entity 2 (E2) - with / without stop words
- F20 - Bigram of first word before Entity1 (E1) - with / without stop words
- F22 - Highest bigram value of words in between entities - with / without stop words
- F24 - number of unique POS types in between the entities - with / without stop words
- F26 - number of unique POS types Entity1 (E1) - with / without stop words
- F28 - number of unique POS types Entity 2 (E2) - with / without stop words
- F30 - POS type of the word with highest tf-idf score in between the entities
- F31 - POS type of the word with highest tf-idf score in before Entity1 (E1)
- F32 - POS type of the word with highest tf-idf score in after Entity 2 (E2)

Different combinations of features show different performance with each classifier. Therefore, the different combinations of features are run through each classifier.

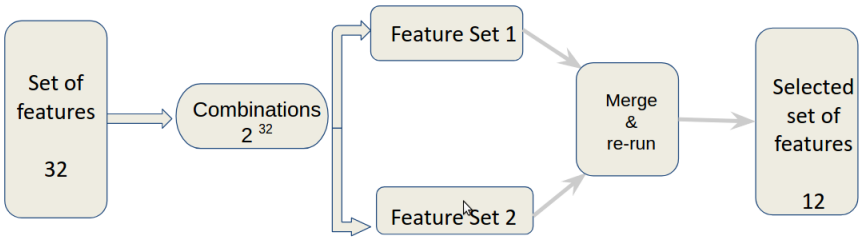

Fig. 3 Feature combination selector

The enhanced feature set includes 32 features in total which gives 2^{32} different combinations of features. Feature set is divided into two and different combinations of features are run with different classifiers and best feature combinations that results in high performance are selected. Best features from each set are merged and re-run and 12 features that gave the best results are selected. Following are best features selected:

- F1 - number of words before Entity1 (E1) - with stop words
- F2 - number of words after Entity 2 (E2) - with stop words

- F3 - POS of the word before Entity1 (E1) - with stop words
- F4 - POS of the word after Entity1 (E1) - without stop words
- F5 - POS of the word after Entity1 (E1) - with stop words
- F6 - POS of the word before Entity 2 (E2) - without stop words
- F7 - POS of the word before Entity 2 (E2) - with stop words
- F8 - number of unique POS types in between the entities - without stop words
- F9 - Bigram of first word before Entity1 (E1) - without stop words
- F10 - Bigram of first word before Entity1 (E1) - with stop words
- F11- Bigram of first word after Entity 2 (E2) - with stop words
- F12 - POS type of the word with highest tf-idf score in before Entity1 (E1)

Subtask 2

This task basically contains 2 steps.

- Relation extraction
- Relation Classification

For the relation extraction task, the cosine similarity of two entities is measured. Cosine similarity value represents how two entities are linguistically or contextually similar. Cosine similarity values are obtained through word2vec. Both CBOW and Skip-gram method values are computed. For the subtask a baseline is not given. As preprocessing steps, all the data are converted to text format. All letters are changed to lowercase and special character and punctuation removal are removed.

Data set which was used is same as the subtask 1.1 which contains set of annotated abstracts for entities and a separate file which contains NLP domain specific relations. We calculate W2V values under the assumption that relationships only exist within annotated entities. So the big image is as follows,

For : each abstract
    For : each entity E1
        For : each E2 from set of possible entities (E2_1, E2_2, …… , E2_n)
            Calculate W2V Cosine similarity between E1 and E2
        End
    End
End

Table 2 shows the cosine similarity values between possible E1 and possible E2 combinations when trained under different models.

Table 2 Cosine similarities of the entities under different trained models.

E1	E2	W2V Model1	W2V Mode	W2V Model	W2V Model1
cclinc:korean:to:english:translation:syste	cclinc:common:coalition:language:system:at:lincoln:labor	0.8339	0.786871	0.318163	0.707578
cclinc:korean:to:english:translation:syste	core:modules	0.868635	0.803579	0.356427	0.524025
cclinc:korean:to:english:translation:syste	language:understanding:and:generation:modules	0.752977	0.724371	0.491328	0.539767
cclinc:korean:to:english:translation:syste	language:neutral:meaning:representation	0.754852	0.742909	0.545376	0.542642
cclinc:korean:to:english:translation:syste	semantic:frame	0.919428	0.742599	0.189378	0.291822

According to the results it can be seen that E1 and E2 has the highest similarity between them. It is noticed that the cosine similarity values depend on the trained model and the selected parameters. Therefore, it is important to select which parameters to use. It is also noted that using cosine similarity it is difficult to find out semantic relationships only for particular domain (NLP domain) as it identifies all levels of semantic relationships.
In above, it is possible to see that 4 w2v networks identify 4 different E1 - E2 pairs as "strongest connections/relatedness/similarity. So, the major problem for us is to narrow down which parameters to use. Rest of the implementation of this task will be covered as future work.

FRAMEWORK - CURRENT METHODOLOGY

In this section, we describe the proposed solution that makes use of some basic Natural Language Processing as well as Machine Learning techniques. With the SemEval 2018 Task 7 in mind, framework was written from scratch in Python 3 programming language with use of NLTK and Scikit-Learn libraries. The diagram of the framework is shown on the *Figure 4*.

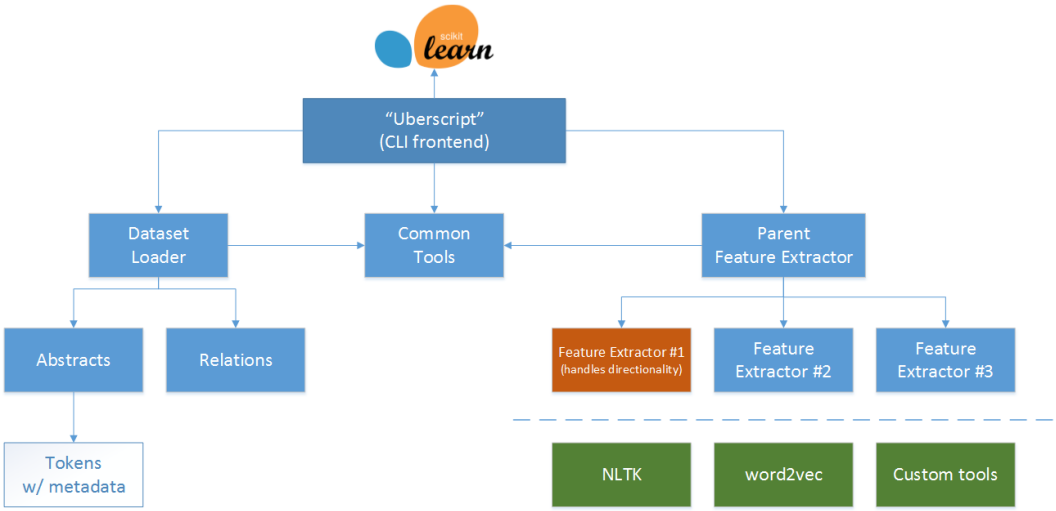

Fig 4. Diagram of the framework.

The framework consists of following main parts:

- Dataset loader allows reading the dataset in both XML file for abstracts and text-file with relations between entities. Each abstract is represented as an instance of a class Abstract, then the content is tokenized and stored as an array of instances of class Object which allows to store metadata like for example entity ID in the particular token. Relations are represented in form of class Relation.

- Common tools, which allows to map class labels to numerical values and vice versa, wrap available classifiers or make use of available metrics (Accuracy, F-Measure).

- Parent feature extractor, which is used for the feature extraction but not directly as it depends on the three Feature Extractors implemented by each team member which can be enabled or disabled in the evaluation scripts.

- Evaluation scripts including main script that stands for partially functional CLI front-end, used for assessing performance of the proposed solution with both the old dataset as well as the new one.

As for now, four evaluation scripts are implemented which allows to assess performance of the new dataset with k-Fold Cross Validation done with k = 5 and split 60/40 for training and testing subset respectively.

Following features are used from the Feature Extractor 1 in order to achieve reasonable as of time of writing performance for the *directionality classification* part for the Subtask 1.1:

- Word distance between entities, calculated as a distance between entities (integer)

- Part-of-Speech tags of both entities, consisting of two features which are mapped PoS tags to the numerical values from the last tokens in particular entities

- Taking into account type of relationship i.e. asymmetric and symmetric (comparison class)

FUTURE WORK

Subtask 1.1

- More work on the feature set to improve performance
- Handling imbalanced classes

Subtask 1.2

- Relation classification on noisy data
- Entities in title, done by concatenating it to the abstract
- Entities inside entities, for this one dataset loader would need to be slightly modified (XML parsing part)

Subtask 2

- Improve feature extraction capability by increasing the size of the dataset and incorporating more constraints to narrow down the relationships.
- Define cutoff frequency to decide whether two entities are semantically related
- Categorize identified relationships into 6 categories.
- Combine with external corpora to reduce overtraining
- Implement other relatedness measurements
- Calculate cosine similarity values using GenSim (Refer Appendix B)

Tool / framework

- Work on making our tool not specific to a single domain

APPENDIX A

CPAN Word2Vec implementation was used which is based on google Word2Vec implementation. This program is free software; We can use it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation.

APPENDIX B

Gensim is a free python library to analyze unstructured text for semantic structure. It produces word vectors with deep learning via word2vec's "skip-gram and CBOW models", using either hierarchical softmax or negative sampling. It takes a sequence of sentences as the input and each sentence should be a list of words. It provides pre-written iterators to apply preprocessing to the text. By changing the parameters model can be trained and cosine similarity between words can be calculated.

CONTRIBUTION

Our team consists of three members and contribution of each individual is listed below.

1. Chathuri Wickramasinghe
    a. Code drop 1
        i. Study of the research problem, related papers
        ii. Literature survey
        iii. Implementation of some features
        iv. Code drop1 presentation preparation
        v. Assistance in writing the files needed for the code drop 1 submission (README FILE - Introduction)
    b. Code drop 2
        i. Study the problem of feature extraction
        ii. Improvements to 1.1 subtask proposed methods for classification task(features_f3.py class)
        iii. Implementation of new methods for 1.1 subtask (including IF-IDF value calculator)
        iv. Implementation of a method for finding best feature combination
        v. Study of Word2Vec tool and other available tools for feature extraction
        vi. Use of CPAN word2vec for subtask 2
        vii. Code drop 2 presentation preparation
        viii. Updates to REQUIREMENT, INSTALL, ORIGIN file, and README (subtask 2 updated part about word2vec and APPENDIX A part)

2. Przemyslaw Lukasz Skryjomski
    a. Study and research of the problem
    b. Framework design
    c. Framework implementation including dataset parser, classes used, feature extractors, common tools
    d. Feature extractor with three features used in the proposed solution as they gave the best results

e. Implementation of evaluation scripts as well as the presentation of the results
f. Directionality classification
g. Code drop 1 and 2 submission
h. Code drop 1 and 2 presentation
i. Wrote BASELINE, INSTALL, ORIGIN file
j. Wrote the framework part of the README file with Subtask 1.1 in mind
k. Maintaining the repository, provided the scripts

3. Samantha Mahendran
    a. Code Drop 1
        i. Study of the research problem
        ii. Review on the related works
        iii. Design of the NLP pipeline
        iv. Selection of the set of features
        v. Implementation of some features
        vi. Code drop1 presentation preparation
        vii. Assistance in writing the files needed for the code drop 1 submission (README FILE - Methodology, Data set)

    b. Code Drop 2
        i. Study and enhancement of the feature set
        ii. Implementation of proposed features (Feature_F2)
        iii. Study and review on related works (Feature extraction)
        iv. Study of word2vec tools (CPAN Word2Vec, Gensim word2vec)
        v. Partial implementation of the tool -GENSIM
        vi. Code drop 2 presentation preparation
        vii. Assistance in code drop 2 submission (README- Methodology, subtasks, future work, BASELINE, ORIGIN)