Team: Hyderabadi Biryani
Hemanjeni Kundem - [hxk180032]
Nazneen Amtul - [axn180041]

Semantic Evaluation - Named Entities

CS 6320 - NLP Final Project

PROBLEM DESCRIPTION

We are given a TAC KBP dataset using which we try to build a model to predict relations and direction of relations between pre-tagged entities. After the training phase , a similar test set is provided to the built model, and we predict the relations and directions between its entities.

Accuracy , recall , macro precision and F-scores under settings relation-only and relation-direction are calculated and reported below. And also, given a test sentence, we load the same model and make relation, direction prediction between the entities. The model we trained for the purpose is probabilistic.

PROPOSED SOLUTION

Our solution begins with preprocessing the given input text. Once done, we move to extracting the semantic features that help in predicting the relation and direction between the given entities. These extracted features are then fed into a Multinomial Naive Bayes Classifier along with the output direction labels and relation labels from the training data. Once the model has completed the learning, it is saved to a disk and retrieved when required for future predictions.

Our solution concentrated on the following semantic features - *Tokenization, Lemmatization, Part of Speech Tagging, Named Entity Recognition, Shortest Path - Dependency Parsing and WordNet Features such as - Hypernyms, Holonyms, Hyponyms and Meronyms*,

FULL IMPLEMENTATION DETAILS

Programming Tools

- **Python 3.7** - Language used
- **Stanford Core NLP** - This library was used to extract tokens, lemmatization, POS tagging
- **SkLearn Naive Bayes** - We used multinomial Naive Bayes from this package to make predictions

- **Sklearn.feature_extraction DictVectorizer** - For converting features to model consumable data
- **SpaCY** - Extract dependencies and return shortest dependency paths
- **NLTK Corpus-Wordnet** - We extracted hypernyms, hyponyms, meronyms and holonyms for tokens using wordnet library
- **Sklearn metrics** - We used this package to collect metrics for the test run. Calculated accuracy ,macro precision, recall and Fscores
- **_pickle** - To store the model trained on disc

Architectural Diagram

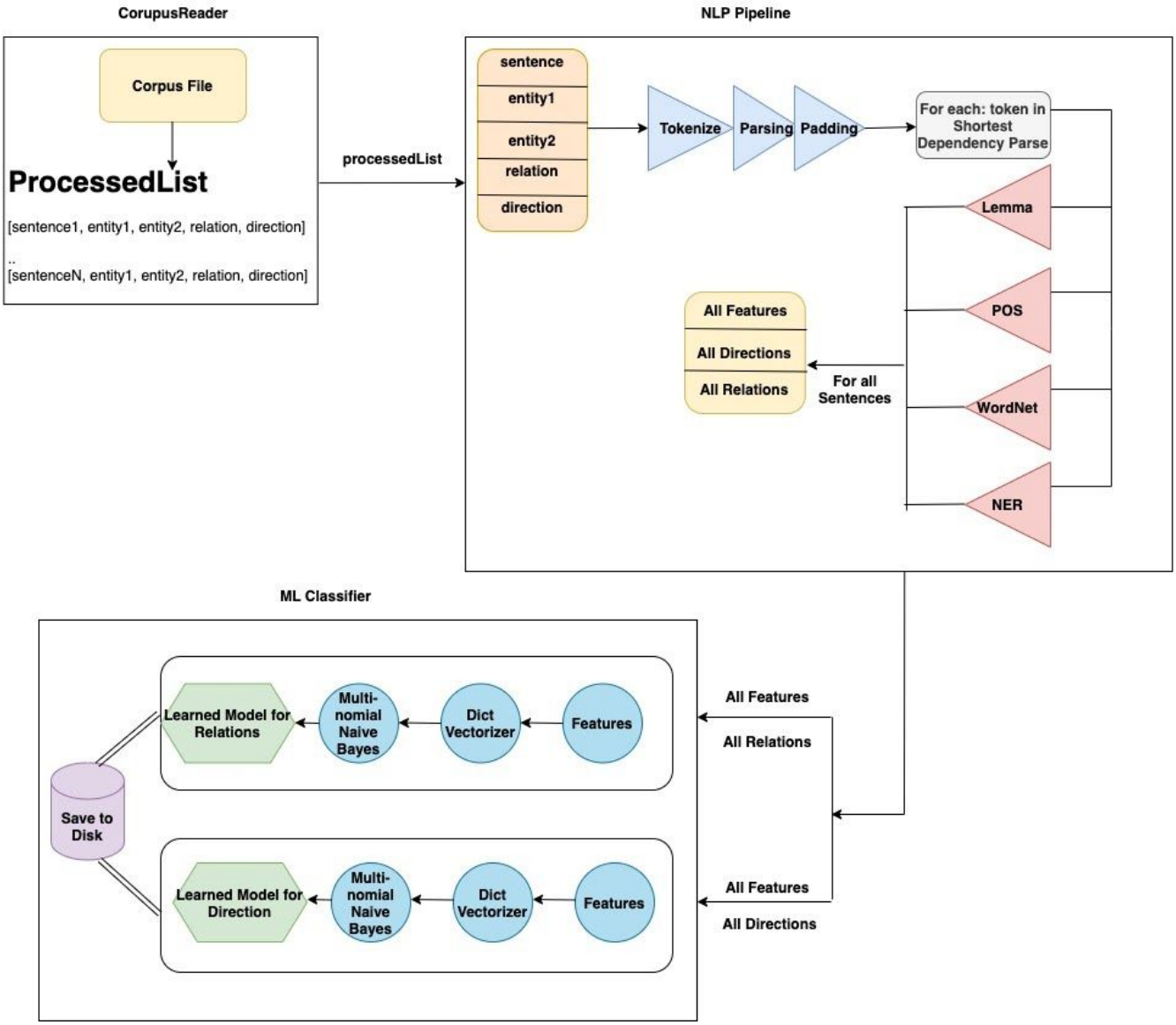

Figure 1: Training Architecture

Training Architecture

Corpus Reader

Reads the input file (training or testing) and converts the raw text into a form suitable for our program to consume. Each paragraph in the file is processed into an array of [sentence, entity1, entity2, relation, direction].

Thus, after preprocessing we have an array of processed information for each sentence

NLP Pipeline

The pipeline consumes each entry which is an array of [sentence, entity1, entity2, relation, direction] and extracts semantic features such as - *Tokenization, Lemmatization, Part of Speech Tagging, Named Entity Recognition, Shortest Path - Dependency Parsing and WordNet Features such as - Hypernyms, Holonyms, Hyponyms and Meronyms*, as required.

All the extracted features for a given sentence are stored in an array. At the end of the pipeline, we have a giant array of all features, all directions and relations for the given corpus.

ML Classifier

We are using two classifiers which use Multinomial Naive Bayes. The first one trains on features and their relations, the second one trains on features and their directions. The two learned models are trained on the training data set and saved to disk.

Testing Architecture

For each given input sentence, the NLP Pipeline is run which extracts the following semantic features - *Tokenization, Lemmatization, Part of Speech Tagging, Named Entity Recognition, Shortest Path - Dependency Parsing and WordNet Features such as - Hypernyms, Holonyms, Hyponyms and Meronyms*, as required

Once done, the two trained models - for relation and direction are extracted from the disk and predictions for the input are made using these classifiers.

The accuracy and predictions for the model are described in the results section and the architecture is explained in the following page.

In both the phases - we have extracted the shortest dependency path as part of parsing, and extracted the semantic features for only these tokens. Doing this, significantly improved the accuracy of our model.

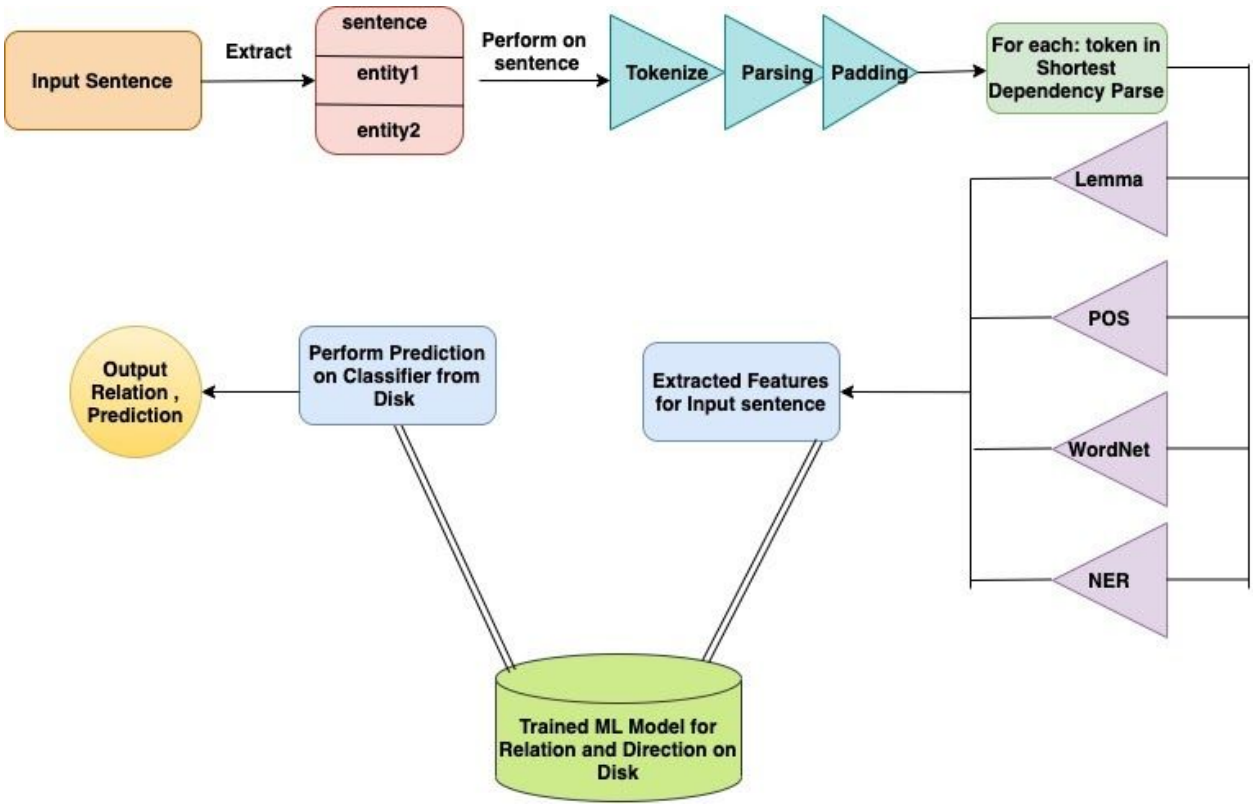

Figure 2: Testing Architecture

Results and Error Analysis

######### METRICS under SETTING #1: Relation #########

METRICS: Overall - Accuracy:

0.52741994847258

METRICS: Overall - Precision, Recall, FScore (Macro)

(0.7481055315470418, 0.47956382938558306, 0.5199799951028874, None)

METRICS: Overall - Precision, Recall, FScore (None)

(([0.92631579, 0.68965517, 0.78592375, 0.78640777, 0.24766355,

0.83428571, 0.89156627, 0.72818792, 0.71604938, 0.875]), ([0.45833333, 0.25974026, 0.81707317, 0.31034483, 0.70044053,

0.62660944, 0.28682171, 0.69551282, 0.59589041, 0.04487179]), ([0.61324042, 0.37735849, 0.80119581, 0.44505495, 0.36593786,

0.71568627, 0.4340176 , 0.71147541, 0.65046729, 0.08536585]), ([192, 231, 328, 261, 454, 233, 258, 312, 292, 156]))

METRICS: Unique Relation Labels:

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

METRICS: Per Label - Precision, Recall, FScore (Macro)

(0.7481055315470418, 0.47956382938558306, 0.5199799951028874, None)

METRICS: Per Label - Precision, Recall, FScore (None)

(([0.92631579, 0.68965517, 0.78592375, 0.78640777, 0.24766355,

0.83428571, 0.89156627, 0.72818792, 0.71604938, 0.875]), ([0.45833333, 0.25974026, 0.81707317, 0.31034483, 0.70044053,

0.62660944, 0.28682171, 0.69551282, 0.59589041, 0.04487179]), ([0.61324042, 0.37735849, 0.80119581, 0.44505495, 0.36593786,

0.71568627, 0.4340176 , 0.71147541, 0.65046729, 0.08536585]), ([192, 231, 328, 261, 454, 233, 258, 312, 292, 156]))

######### METRICS under SETTING #2: Relation and Direction #########

METRICS: Accuracy:

0.3803867021015853

METRICS: Overall - Precision, Recall, FScore (Macro)

(0.491847200285561, 0.32213567384024866, 0.34234180163711625, None)

METRICS: Overall - Precision, Recall, FScore (None)

(([0.64975727, 0.55111929, 0.56489791, 0.54923418, 0.22430611,

0.25510268, 0.49191012, 0.5268752 , 0.50227558, 0.47711746,

0.49623727, 0.61333333]), ([0.33129833, 0.26982289, 0.48085039, 0.37286417, 0.48115403,

0.50565627, 0.26518866, 0.39402153, 0.46852855, 0.24499804,

0.04739757, 0.00384766]), ([0.43884057, 0.36227789, 0.51949667, 0.444182 , 0.30597275,

0.33911996, 0.34460229, 0.45086526, 0.48481551, 0.32375109,

0.0865303 , 0.00764734]), ([120393, 229747, 377945, 398440, 499815, 408750, 369733, 383710,

407973, 318125, 141906, 35866]))

METRICS: Unique Direction Labels:

[0, 1, 2]

METRICS: Per Label - Precision, Recall, FScore (None)

(([0, 0.64975727, 0.55111929]), ([0, 0.33129833, 0.26982289]), ([0, 0.43884057, 0.36227789]), ([0, 120393, 229747]))

METRICS: Per Label - Precision, Recall, FScore (Macro)

(0.40029218794766647, 0.20037374114540155, 0.26703948669791555, None)

Total Time Taken (Training + Testing):

1728 seconds/28 minutes

Problems Encountered and Resolutions

Problem Encountered #1: Feeding Input Features to ML Classifier

Once we obtained the input features for all the sentences in the training data, we were having difficulty in providing these features to our Multinomial Naive Bayes Classifier. Most of the examples we had come across in our research involved only features that are one-dimensional. But our feature set contained data that was a multidimensional array with rows representing sentence tokens and columns representing the semantic feature and its value

Resolution: Converting the multidimensional array to a DictVectorizer helped.

Problem Encountered #2: Obtaining NER for compound entity words

There were several examples in the training and testing set where the given entity spanned multiple words. Ex: San Jose. These represent a single concept but when this token was sent to NER Libraries to generate the NER Tag, it wasn't able to successfully do that.

Resolution: Concatenation of the compound entity words into a single word and using that to extract the NER tags

Problem Encountered #3: Consistent length for features across sentences

As the training data varies in length, the feature set of each sentence was of variable length too. The ML Classifier doesn't function when the feature set of input is not consistent.

Resolution: Padding - when a feature does not exist for a sentence, we have padded a special symbol to indicate its absence

Pending Issues

Pending Issue #1: Multi word entities

NER extraction for multi word is resulting in '0'. Handling of this situation in a better way is envisioned.

Pending Issue #2: Accuracy improvement

Our model accuracy is at 52%. Could explore options which could help increase accuracy

Potential Improvements

Improvement#1: Improving Accuracy

As explained in the results and the error analysis section, our accuracy under both the settings can be greatly improved. One of the ways is to include more meaningful semantic features in the NLP Pipeline such as - FrameNet, PropBank or NomLex.

Improvement #2: Strengthen ML Classifier

We have currently used Multinomial Naive Bayes as the default classifier. Using Gaussian Naive Bayes instead of it could also be a potential remedy for improving the training of model