Opinion Dominion - Jaecee Naylor

Phase 2 Project Report

IE Task

The IE task I am working to complete involves extracting an opinion's target expression (OTE), polarity, and category as a tuple. Given an opinionated text about a target entity, the task is to identify all the opinion tuples containing an aspect category, OTE, and sentiment polarity. An aspect category is an entity, E, and attribute, A, pair that an opinion is expressed towards. This category is pulled from a predefined pool of labels. The OTE is the explicit mention used to reference the E#A pair, and where there is no explicit reference, OTE takes on a value of NULL. Lastly, a sentiment polarity is assigned to every instance of an E#A pair. Sentiment polarity can be negative, positive, or neutral. The input I receive is the data from the SemEval 2016 task 5 restaurant data set in English. My program parses this input to create Review, Sentence, and Opinion objects corresponding to information in the provided XML. The dataset splits each review into sentences and also provides the manually annotated data for the opinion(s) found in the sentence. These correspond to the label types of "target" for the OTE, "category" for the entity#attribute (E#A) pair, and "polarity" for the polarity of the opinion. My system reads each sentence on a review and generates its own predictions for these variables stored in an Opinion object which corresponds to the output labels. The output of my system is the associated scores when comparing the predicted values to the expected values on an opinion. My system also creates six files, a file for each opinion label that is expected and predicted. The file format for each file is:

SentenceID: <sentence_id>
<Label>: <label_value>
Example:
SentenceID: 1090587:1
Polarity: positive

Resources List

1. Dataset(s): https://alt.qcri.org/semeval2016/task5/index.php?id=data-and-tools
2. NLTK (used for stopwords, wordnet, word and sentence tokenizers, lemmatizer, sentiment analysis, and part of speech tagging): https://www.nltk.org/
3. Stanza (used to gather relations between words in a sentence): https://stanfordnlp.github.io/stanza/
4. Sklearn Crfsuite (used to train a CRF model for BIO tagging for OTE labels): https://sklearn-crfsuite.readthedocs.io/en/latest/
5. Sklearn (used to train SVMs for polarity and category labeling as well as perform cross-validation): https://scikit-learn.org/stable/
6. Numpy (used to build feature vectors and label vectors): https://numpy.org/

Technical Description

For phase two, I have built statistical models to determine each label on an opinion. For category (E#A) extraction, I train a support vector machine (SVM) with a linear kernel that becomes a one versus one schema. The system uses the 1000 most common unigrams (excluding stopwords) as features for the feature vectors. For each sentence in the training data, a feature vector is composed using the features mentioned above and the annotated category value is considered the correct label for the feature vector. The SVM is trained on the training feature vectors and corresponding labels. Next for each test sentence, a feature vector in the same way one is built for a training sentence, and the trained SVM predicts a corresponding category. The system will also train an SVM for each specific category and perform one versus all classification.

For polarity prediction a process the same process is followed as above. In addition to the top 1000 most common unique unigrams for features in the feature vector, there is an additional numerical feature tagged to the end that corresponds to the aspect category chosen. The feature vectors extracted from the training data and their corresponding correct polarity labels are used to train an SVM of a one versus one schema. Then, for each test sentence, a feature vector is built and the trained support vector machine is used to predict the correct label. The system will also train an SVM for each specific category and perform one vs all classification.

Lastly, for OTE extraction I designed another baseline model as well as a statistical method to determine the target using BIO labeling. The baseline system will use the training instances to create for each aspect category a list of OTEs associated with it. Then for a test sentence and an assigned category, an OTE is found by finding the first instance of a target in the assigned categories list in the test sentence. If no target is found, the OTE is assigned the value of NULL.

The second way the system will extract an OTE is by building a Conditional Random Fields (CRF) model to perform BIO labeling on sentences. On the training data, the system designs feature vectors for each opinion extracted. For each word, in a training sentence, a feature vector is built containing the word itself, its part of speech, its shape (capital letter, lowercase letter, digit, punctuation, other symbols), its type (uppercase, digit, symbol, combination, or other), whether or not its a stop word, the two proceeding words, the three upcoming words, and the two proceeding and upcoming parts of speech, word shapes, and word types. This word is assigned a BIO label indicating if it is the beginning, inside, or outside of an OTE. The CRF is trained on these training feature vectors and their corresponding labels. Then for each word in a test sentence, a feature vector is built as defined above and the trained CRF model determined the BIO tag. The BIO tags are used to create the OTE for the opinion tuple in question.

Phase 2 vs. Phase 1

For phase 2 I have built a completely different system to determine labels on opinions. In phase 1, I used stanza to see how words in a sentence were related to each other. Next, I chose an OTE based on it being a noun and being attached to an adjective. I used the adjectives associated with an OTE to determine the polarity of the entity found. To find the category, I labeled it based on if the OTE found was in a categories list of associated OTEs from the training examples. My phase one implementation did not utilize any machine learning or statistical models. My phase two IE model does a better job at matching the task description as well as utilizing machine learning models to find labels on an opinion.

Evaluation

The performance of the system is evaluated using accuracy, precision, and f1-measure scores for each of the corresponding output labels. For OTE scores, I chose to calculate these scores based on word overlap between the predicted OTE and the expected OTE. This has its downsides as the order does not matter and small words in common that are not important to the answer will boost the score falsely potentially.

For OTE recall, precision, and f1-measure scores are produced as follows:
Recall: The number of correct words generated by my system divided by the total number of words in the expected target answer.
Precision: The number of correct words generated by my system divided by the total number of words generated by my system for the target.
F-Measure = (2*Recall*Precision) ÷(Precision + Recall)

For the Polarity and Aspect Category the scores are calculated as follows:
Recall: The average of all the classes recall scores that are calculated by the number of events where the system correctly identified class *i* divided by the number of instances where the expected class matched class *i*
Precision: The average of all the classes precision scores that are calculated by the number of events where the system correctly identified class *i* divided by the number of instances where the system predicted class *i*
F-Measure = (2*Recall*Precision) ÷(Precision + Recall)

The dataset is used to evaluate the performance of my IE system by leaving out 25% of examples to be test examples, through cross-validation, and by creating models on the training data in the training file as a whole and putting the test examples in the testing file through the trained model to see that performance.

When performing cross-validation where k is equal to 5 the results and scores are shown below. The values are listed in order for the score that each fold received. I will also not be showing the 1 vs All SVM results for polarity and category labels or target via category list since these usually always performed slightly worse.

	Recall	**Precision**	**F1-Score**
Target (BIO labeling CRF)	0.3854, 0.3598, 0.3218, 0.3124, 0.4396	0.470, 0.4238, 0.39233, 0.4060, 0.5267	0.4235, 0.3892, 0.35361,0.3531, 0.4792
Aspect Category (1vs1 SVM)	0.4058, 0.40669, 0.3616, 0.3296, 0.3507	0.4304, 0.40282, 0.4074, 0.3892, 0.3494	0.4178, 0.4047, 0.383, 0.356, 0.35
Polarity (1vs1 SVM)	0.49126, 0.5059, 0.488, 0.5796, 0.6756	0.4788, 0.5117, 0.501, 0.5643, 0.655	0.48496, 0.5088, 0.4952, 0.571, 0.665

When leaving out 25% of the training examples to simulate the testing examples the results and scores are below.

	Recall	**Precision**	**F1-Score**
Target (via category list)	0.3520	0.47049	0.4027
Target (BIO labeling CRF)	0.3914	0.48737	0.4315
Aspect Category (1vs1 SVM)	0.3766	0.3618	0.369039
Aspect Category (1vsAll SVM)	0.3414	0.3309	0.3361
Polarity (1vs1 SVM)	0.46427	0.49282	0.47812
Polarity (1vsAll SVM)	0.4586	0.4975	0.4773

Results from creating a model on entire training data and putting all test examples through the model to gather predictions. I will also not be showing the 1 vs All SVM results for polarity and category labels or target via category list since these usually always performed slightly worse.

	Recall	**Precision**	**F1-Score**
Target **(BIO labeling CRF)**	0.3549	0.4188	0.3842
Aspect Category **(1vs1 SVM)**	0.3823	0.4195	0.40006
Polarity **(1vs1 SVM)**	0.5019	0.5268	0.51408

The results for the first phase model when ran on the unseen test data are below so we can perform an analysis between the systems.

	Recall	**Precision**	**F1-Score**
Target **(using word relationships)**	0.2848	0.3299	0.3057
Aspect Category **(via target list)**	0.1297	0.7397	0.2207
Polarity **(using adjective sentiment)**	0.3775	0.3816	0.3795

It can be seen from the results that the statistical model outperforms the original model on unseen data.