ÖZYEĞİN UNIVERSITY
FACULTY OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE

CS 402

2018 Sprin0067

SENIOR PROJECT REPORT

Aspect based Sentiment Analysis of Online Reviews

By
Kenan Al Fayoumi

Supervised By
Dr. Reyyan Yeniterzi

Declaration of Own Work Statement/ (Plagiarism Statement)

Hereby I confirm that all this work is original and my own. I have clearly referenced/listed all sources as appropriate and given the sources of all pictures, data etc. that are not my own. I have not made any use of the essay(s) or other work of any other student(s) either past or present, at this or any other educational institution. I also declare that this project has not previously been submitted for assessment in any other course, degree or qualification at this or any other educational institution.

Student Name and Surname:	Kenan Al Fayoumi
Signature:	
Place, Date:	Ozyegin University 17/5/2018

Abstract

Over the last decade, the web has been growing non-stop. The internet is now full of opinions and reviews about pretty much everything. In many domains, it's essential to understand what people are saying or complaining in their review. However, the huge amount of reviews usually makes it impossible or very time and resources consuming to read each and every one of them. Sentiment Analysis often offer a solution for such situations. Where given a sentence, a sentiment analysis system would report the polarity of that sentence. But sometimes it's not enough to say whether a sentence was positive or negative, sometimes we would like to understand what was/were the entity/entities being described in that sentence and what was the sentiment value for each of those. For example, in a given sentence: "The phone battery really lasts, but the screen gets scratched easily." we should identify the phone battery as the first described entity/aspect, and the phone screen being the second. We should also be able to decide on the polarity (negative or positive) for each of the aspects we identified.

To solve this problem, we use Aspect-Based Sentiment Analysis (ABSA). Given a text, an ABSA system can identify aspects discussed or mentioned in that text, and determine a sentiment value for each aspect. Many ABSA systems have been proposed. In our project we will be working on implementing an ABSA system that has following subtasks: Opinion Target Expressions (OTE) extraction, aspect category or entity identification, and aspect polarity estimation.

This report will describe provide a step-by-step in our approach of extracting aspect term (OTE) from reviews. For this subtask, we tried two approaches, first was a rule-based approach. Second is a Supervised-Learning approach, where we perform sequence labeling using a Conditional Random Fields (CRF) model used on textual features extracted from the given reviews. We will also talk about our supervised approaches for the remaining ABSA subtasks: aspect category detection, and aspect polarity (sentiment)

estimation. In both subtask we perform classification using normal machine learning (SVM) and deep learning (CNN and LSTM) models.

Contents

I. Introduction

In a normal sentiment analysis task, given a sentence, the output would be a prediction of the overall negative or positive polarity of that sentence. However, sometimes we need to understand what are the specific sentiments towards various aspects of an entity, for example: a review from a restaurant: “The pizza was good, but I hated the atmosphere.” contains positive opinion towards aspect pizza but strong negative sentiment towards aspect atmosphere. Classifying the overall sentiment as negative would neglect the fact that pizza was good.

This is where ABSA comes in. In 2010, a new framework named aspect-based sentiment analysis (ABSA) [1] was proposed to address this problem. Given a sentence, the focus will be on identifying each aspects or particular entities in that sentence and determining the polarity for those aspects.

In recent years, Aspect-Based Sentiment Analysis has gained a lot of interest academically. It also gained a role in many business applications. It can be a source of information and provide insights that can: determine marketing strategy, improve campaign success and improve product messaging. The emergence of ABSA was not accidental. In the past decade, most services like restaurants, hotels, stores, went online, they either have website, or they have a page about their enterprise in some other website. Along with that increased the number of customers online reviews.

SemEval (Semantic Evaluation) is an annual international workshop which provides NLP tasks where participants are given a problem or task and in order to evaluate their approach to solve a specific problem, they’re given data to test their system on and measure how good it performed.

In this project we will follow SemEval 2016 TASK 5 (Aspect-Based Sentiment Analysis). We will be focusing on Subtask_1. For this task, we are given reviews datasets for multiple domains and in several languages. For each domain training data, test data, and the golden data are provided. This SemEval task has several subtasks. Participants are free to choose the subtasks/languages/domains they wish to participate in. In part 1 of our project, we concentrated more on identifying the Opinion Term Expression (OTE) for the English Restaurants dataset.

In part 2, we focused on the two remaining subtasks: aspect category detection and polarity estimation.

ABSA has 3 main subtasks:

1. **Identifying aspect term:**

To solve the problem of identifying aspect terms, we approached this from two different paths. Our first was, a rule based approach. Where extract a word as an opinion target if it matches a rule. Most of the rules are based on dependency relations of a sentence, example of relations used: Direct-Object relation, clausal-complement relation, prepositional relation…etc. Those rules are based on Sentic.Net Aspect parser [2].

The second approach, was to transform our task into a supervised learning sequence labeling or classification task, where we use IOB-tagging to identify which words are opinion targets and which are not. For that matter, we used Conditional Random Fields. CRFs are often used for this type of tasks, because they fall into the sequence modelling family where they consider nearby neighbors in a sequence.

After implementing both approaches and testing both models on our provided testing data, we evaluated the performance of these methods. Our rule-based system was not complex enough to cover most cases in our testing data, so it failed to extract most opinion target expressions. Our other method however, came up with pretty decent results. Being frequently used and mostly successful in such tasks, it’s what we expected from CRFs used for sequence labeling.

2. **Aspect category detection:**

In order to detect categories of aspect mentioned in a sentence (price, atmosphere, service…etc.), we implemented two different One-vs-Rest classifiers.

Our first model uses SVMs, and its trained on term frequency–inverse document frequency (TF-IDF) matrix. We then used this classifier to predict the probabilities for each sentence to belong to a specific category. After that, a threshold for probability value is used to decide which categories should be assigned for that sentence. This model is an implementation of the baseline approach for Subtask1 Slot1 (Aspect category detection) SemEval provided in their paper [3].

Our second model uses CNN networks as binary classifiers (one network for each class). Similar to the previous model we use those to generate the class probabilities for each sentence and then use a threshold to assign labels to each sentence.

We also a single CNN as multi-label classifiers. As input to all deep learning models we use word embeddings.

3. Aspect polarity estimation:

For this task our first approach was based on SemEval's baseline approach for Subtask1 Slot3 (Polarity prediction). We are using a linear SVM classifier which is trained on unigram features from the training data. As features, we also use the categories of that sentence as integer features.

Similar to the previous subtask, our other approaches use deep learning models to perform classification. These models include deep neural networks, CNN, and LSTM.

Throughout this project we used Python as our programming language. Python offers a wide range of open-source libraries for all purposes, but most importantly, machine learning libraries which offer implementations for many ML algorithms and models.

In the next chapter we will provide some background information on methods, algorithms, and tools we used throughout our project. Problem statement and all details about all three tasks will be provided in the second chapter. The third chapter will be split into three sections. In section 1 we will talk about our two approaches for OTE extraction task. Sections 2 and 3 will discuss the implementations of our model for the aspect category detection and polarity estimation, respectively. Evaluations measures and constrains for all tasks will be provided in Chapter 4. Chapter 4 will also include evaluating the results we obtained testing our model on Test datasets provided by SemEval 2016 Task 5.

Background

Throughout this project we have utilized many techniques and tools. A list of these are found below:

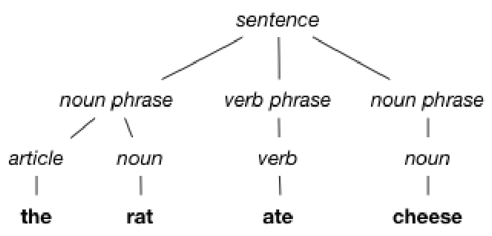
Concepts and Techniques:

Object-oriented programming:

Object oriented programming or OOP is a programming paradigm based on the concept of "objects". OOP was developed to increase the reusability and maintainability of source code.

Parse Tree:

Parse tree is an ordered, rooted tree representing the detailed structure of a sentence starting with phrases (such as Noun Phrases, Verb Phrases…etc.), and ending with Parts Of Speech.[1]



Recall, Precision and F1 Scores:

Recall or sensitivity is the fraction of relevant instances that have been retrieved over the total amount of relevant instances.
Precision is the fraction of relevant instances among the retrieved instances.
F1 score is a measure of a test's accuracy. The F1 score is the harmonic average of the precision and recall.
These metrics are used to evaluate how good a model is.

Machine Learning:

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed[7].

[1] http://www.cs.cornell.edu/courses/cs2112/2015fa/lectures/lecture.html?id=parsing

Machine learning can be split into many methods, a relevant method for our project is **Supervised machine learning**. In Supervised machine learning, the model takes as input data and desired output for that data, the model then produces an output for that input and compares it with the desired output and then adjusts itself, so it can learn (not memorize) how to correctly predict such input patterns in the next time.
An example of a Supervised machine learning model is:

- **Conditional Random Fields:**

  Conditional Random Fields or CRFs are a discriminative undirected probabilistic graphical framework for labeling and segmenting structured data, such as sequences, trees and lattices. CRFs are often used in structured prediction, POS Tagging, shallow parsing and named entity recognition.

- **Support Vector Machines:**

  Support Vector Machines or SVM, also known as maximum margin classifier, are supervised learning models with associated learning methods that analyze data used for classification, regression and outliers detection. SVM works on finding the maximum margin hyper-plane that separates classes with maximum distance between them. SVMs advantages is that it can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

- **Multi-Class Classification:**

  Multiclass classification means a classification task with more than two classes; e.g., classify a set of images of fruits which may be oranges, apples, or pears. Multiclass classification makes the assumption that each sample is assigned to one and only one label: a fruit can be either an apple or a pear but not both at the same time.

- **Multi-Label Classification:**

  Multi-label classification or Multi-output classification assigns to each sample a set of target labels. This can be thought as predicting properties of a data-point that are not mutually exclusive, such as topics that are relevant for a document. A text might be about any of religion, politics, finance or education at the same time or none of these.

- ### One-VS-Rest Strategy:

One-VS-Rest, also known as one-vs-all, this strategy consists in fitting one classifier per class. For each classifier, the class is fitted against all the other classes. An advantage of this approach is its interpretability. Since each class is represented by one and only one classifier, it is possible to gain knowledge about the class by inspecting its corresponding classifier. This is the most commonly used strategy and is a fair default choice.

- ### Cross-Validation:

Cross-validation is a technique to evaluate predictive models by partitioning the original sample into a training set to train the model, and a test set to evaluate it. Cross-validation is also used to learn the parameters of predictive models.

A method of cross-validation is:

  - #### K-Fold:

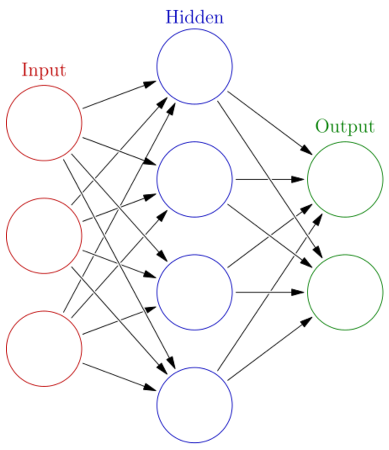
    K-Fold divides all the samples in k groups of samples of equal sizes (if possible). The predictive model is learned using k - 1 folds, and the fold left out is used for testing.

❖ Deep learning:

Deep Learning is a subset of machine learning that uses artificial neural networks which simulate the brain’s neural networks. Most of the times, these networks are deep (have many layers). In each layer different computation or transformations are done using weights that can be modified, which is how the model learns.

Below is a graph[2] of an artificial neural network which has 1 hidden layer and input and output layers:

[2] https://en.wikipedia.org/wiki/Artificial_neural_network



Below are examples of deep neural networks:

- **Convolutional Neural Networks:**

  CNNs or ConvNets are a deep learning model. The name Convolutional Neural Network comes from the convolutional layers inside the network. Convolutional layers extracts features from the input by sliding filters over input and generating maps which holds extracted features. CNNs are used widely in image processing and natural language processing tasks.

- **Recurrent Neural Networks:**

  RNNs are a type of deep learning models that handle sequential data (text, audio…etc.). RNNs doesn't only take the new input at the current timestep, but also the output/state of the network at the previous step. This creates a sort of memory in the network which makes them successful in handling tasks that involve data that is naturally sequential such as human language.

  - **Long Short-Term Memory:**

    LSTMs are an upgraded version or RNNs that have additional gates. These gates are responsible for remembering and forgetting values over time. This allows the network to overcome the problems RNNs had such as vanishing gradients and remembering long-term dependencies.

- **Overfitting:**

  Overfitting refers to a model that learns the training data too well (memorizing), it starts learning noise in the data, so it usually doesn't generalize on new unseen data.

- **Regularization:**

  Regularization is a method used to prevent machine learning models from overfitting.

- **L2 regularization:**

  L2 or ridge regression is a type of regularization method. L2 adds the squared magnitude of weights as a penalty term to the loss function. This forces the model to prefer smaller weights and build less complex models.

- **Dropout:**

  Dropout is a regularization technique where randomly selected neurons are dropped/switched off during training. This prevents the neurons from overfitting and forces them to learn distinctive features instead of learning the same feature.

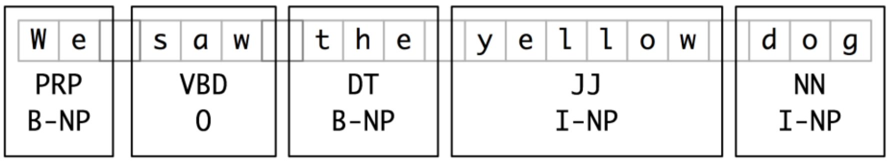
Word Embeddings:

Word embeddings are vector representations of words. This transformation from strings (words) to numbers is important because many machine learning algorithms require continues vectors of values and won't work well on plain strings. Embeddings are also a more expressive representation of words and the relation between words in a corpus vocabulary.

I-O-B Tagging:

The IOB format (short for inside, outside, beginning) is a common tagging format for tagging tokens in a chunking task in computational linguistics. The B- prefix before a tag indicates that the tag is the beginning of a chunk, and an I- prefix before a tag indicates that the tag is inside a chunk. The B- tag is used only when a tag is followed by a tag of the same type without O tokens between them. An O tag indicates that a token belongs to no chunk.

Below is the example[3] provided by NLTK of an IOB tagged sentence. The chunking method is Noun Phrases (NP). "WE" is the beginning of a noun phrase so it has a "B-NP" tag. "saw" is not inside a NP so it has an 'O' tag. "yellow" and "dog" are both inside a noun phrase, so the corresponding IOB tag is 'I-NP'.



[3] http://www.nltk.org/book/ch07.html

Term Frequency–Inverse Document Frequency:

Term frequency–inverse document frequency or TF-IDF is a numerical statistic that is intended to reflect how important a word is to a document in a or corpus. It is used as a weighting factor in searches of information retrieval, text mining, and user modeling. TF-IDF is calculated using the formula[4] below:

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

tf_{ij} = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

Tools and Libraries:

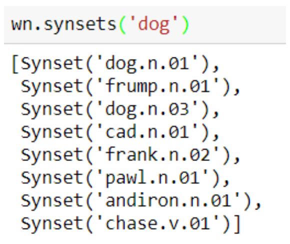
NLTK:

The Natural Language Toolkit[5] is a suite of libraries and programs for symbolic and statistical natural language processing for English written in the Python programming language.

WordNet:

WordNet[6] is an English language database which keeps sets of words that are linked together by their semantic relations. Those sets are called Synsets and they can be synonyms or antonyms.

In the example below: we search for Synsets for the word 'dog', and we get a list of Synsets or words that has different meaning.



```
wn.synsets('dog')
```

```
[Synset('dog.n.01'),
 Synset('frump.n.01'),
 Synset('dog.n.03'),
 Synset('cad.n.01'),
 Synset('frank.n.02'),
 Synset('pawl.n.01'),
 Synset('andiron.n.01'),
 Synset('chase.v.01')]
```

We can get an example for the first and third Synset. Here we can see that how the word 'dog' has two different usages:

[4] https://deeplearning4j.org/bagofwords-tf-idf
[5] http://www.nltk.org/
[6] https://wordnet.princeton.edu/

```
wn.synset('dog.n.01').examples()
```
```
['the dog barked all night']
```
```
wn.synset('dog.n.03').examples()
```
```
['you lucky dog']
```

We can also check similarity between words:

```
truck = wn.synset('truck.n.01')
car = wn.synset('car.n.01')
print_similarity(truck,car)
```
```
Similarity 0.9166666666666666
```

SentiWordNet:

Similar to WordNet, SentiWordNet[7] is a lexical database for English language that has Synsets. SentiWordNet assigns sentiment scores for each Synset: positivity, negativity and objectivity.

Here are example of using SentiWordNet to get sentiment scores for different words:

```
good = swn.senti_synset('good.a.01')
print_scores(good)
```
```
positivity: 0.75
negativity: 0.0
```
```
okay = swn.senti_synset('okay.n.01')
print_scores(okay)
```
```
Positivity: 0.125
Negativity: 0.125
```
```
dog = swn.senti_synset('dog.n.01')
print_scores(dog)
```
```
positivity: 0.0
negativity: 0.0
```

SenticNet:

SenticNet[8] is a publicly available semantic and affective resource for concept-level sentiment analysis. SenticNet 3.0 has a vocabulary size of 50,000 words. SenticNet can be used to get words semantically similar words. It can also be used to get polarity values.

```
senticnet.semantics('dog')
```
```
['pet', 'best_friend', 'cur', 'domestic_animal', 'domesticated_animal']
```
```
senticnet.polarity_intense('outstanding')
```
```
'0.868'
```
```
senticnet.polarity_intense('unreliable')
```
```
'-0.51'
```

Stanford CoreNLP:

Stanford CoreNLP[9] is a software that provides a set of human language technology tools. It can give the base forms of words, their parts of speech, mark up the structure of sentences in terms of phrases and syntactic dependencies, indicate which noun phrases refer to the same entities, indicate sentiment, extract particular or open-class relations between entity mentions, etc.

[7] http://sentiwordnet.isti.cnr.it/
[8] http://sentic.net/
[9] https://stanfordnlp.github.io/CoreNLP

CRFsuite:

CRFsuite[10] is an implementation of Conditional Random Fields (CRFs) for labeling sequential data.

Scikit-Learn:

Scikit-learn[11] or sklearn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms.

TensorFlow:

TensorFlow[12] is an open source software library for numerical computation using data flow graphs. TensorFlow is used widely in developing deep learning models.

Jupyter notebook:

Jupyter notebook (formerly known as IPython notebook) is an interactive computational environment, in which you can combine code execution, rich text, mathematics, plots and rich media. Notebooks are used for better visualization and code handling.

Knowledge and skills background:

- **CS102 Object-Oriented Programming:** Object-oriented programming skills.
- **CS222 Programming Studio:** Clean and refactored coding skills.
- **CS452 Data Science with Python:** Python skills and using ML libraries and tools.
- **CS300, CS400 Summer Training:** (Topic modelling on tweets) Python skills and using ML libraries and tools.
- **CS454 Intro to Machine Learning and Artificial Neural Networks:** Better understanding of machine learning algorithms and cross-validation.
- **CS466 Intro to Deep Learning:** Knowledge on deep learning algorithms and using TensorFlow.
- **CS449 Natural language processing:** More knowledge on text processing and classification, in addition to sequential machine learning models.

[10] http://www.chokkan.org/software/crfsuite/
[11] http://scikit-learn.org/stable/index.html
[12] https://www.tensorflow.org/

Engineering standards:

- Information Security Management *ISO 27001*
- Data format: XML

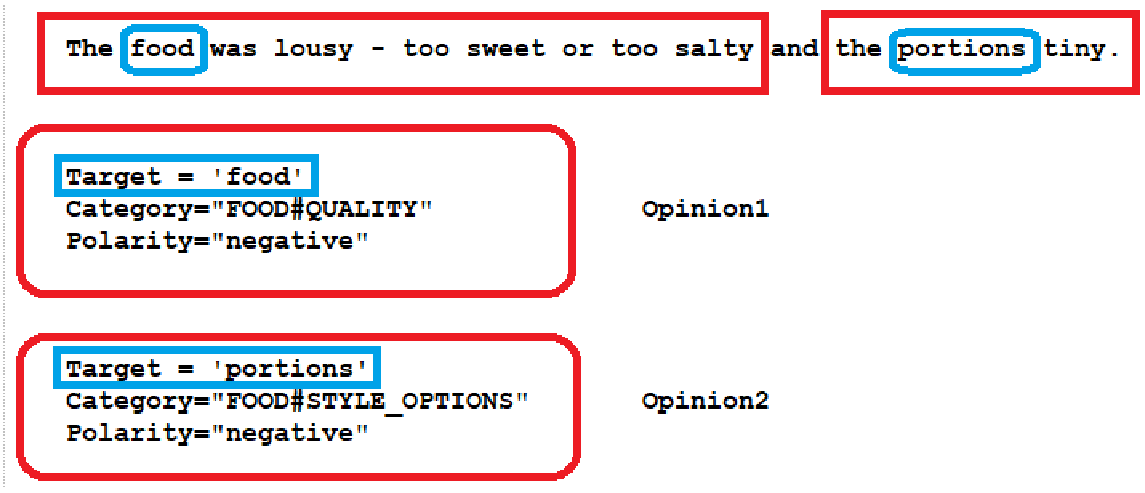
II. Problem Statement

As mentioned before, our project is based on SemEval 2016 Task 5[13]. Given training and testing datasets, the goal of this task is to correctly identify the aspects of entities and the polarity expressed for each aspect.

III.A Given Datasets:

In total there are 39 datasets available for this task. These include both training and testing datasets. The reviews were from 7 domains and 8 languages. Domains include: restaurants, laptops, mobile phones, digital cameras, hotels, and museums. Available languages are: English, Arabic, Chinese, Dutch, French, Russian, Spanish and Turkish. For data format, check appendix A.
Initially we've decided to work on English reviews from Restaurants. Later on, we will extend our system to work on different languages, and test it on other datasets.
An example of a review from Restaurants Reviews Subtask1 Training Dataset is given below:

```
Sentence:

    The food was lousy - too sweet or too salty and the portions tiny.
```



In this example, Category is the detected Entity-Attribute or the aspect that is being talked about in the sentence, Target is the term that corresponds to the aspect or category being mentioned in the text, Polarity is the negativity or positivity of that particular aspect. Together these three (Category, Target, Polarity) form an opinion. In the given example, we have 2 opinions: one is talking negatively about food quality, the other is describing the portions (food style), which is also negative.

[13] http://alt.qcri.org/semeval2016/task5/

The table below demonstrates the given training and testing datasets sizes. #Tuples refers to number of number of opinion tuples.

Dataset	**#Reviews (Training set)**	**#Sentences (Training set)**	**#Tuples (Training set)**	**#Reviews (Test set)**	**# Sentences (Test set)**	**#Tuples (Test set)**
ENG REST	350	2000	2507	90	676	859

III.B Subtask Description:

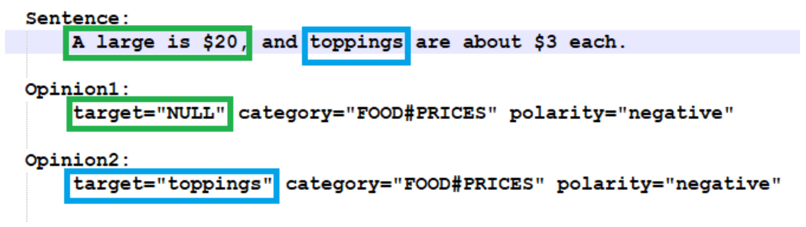
Task 5 has 3 different subtasks: Sentence-level ABSA, Text-level ABSA, and out-of-domain ABSA. Our focus will be on the first subtask:

- **Subtask 1:**

In this subtask, we are given reviews as separate sentences, and for each sentence we are asked to identify all opinion tuples. An opinion tuple contains the following:

1) SLOT 2: Aspect Term Extraction:

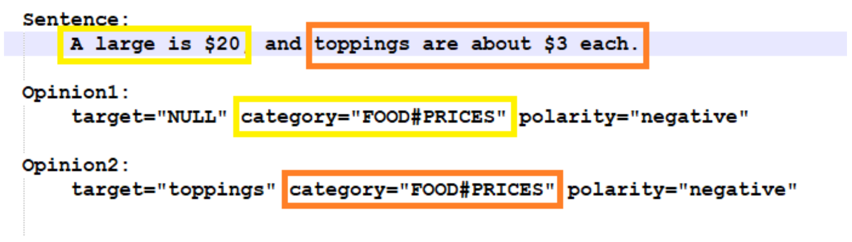
The task is to extract the linguistic expression used in the given text to refer to the reviewed entity E of each E#A pair. This slot will have the value ‘NULL’ if the target expression is not explicitly mentioned in the sentence. This example is a good case to demonstrate how a target can implicitly mentioned. The first opinion’s SLOT2 value is ‘NULL’, here we can see that the word “large” is not considered as a target, because the word large here refers to a large (Ice Cream, sandwich …etc.), and that second word is the actual target. That’s why the value should be ‘Null’.



```
Sentence:
    A large is $20, and toppings are about $3 each.

Opinion1:
    target="NULL" category="FOOD#PRICES" polarity="negative"

Opinion2:
    target="toppings" category="FOOD#PRICES" polarity="negative"
```

2) SLOT 1: Aspect Category (Entity-Attribute) Detection:

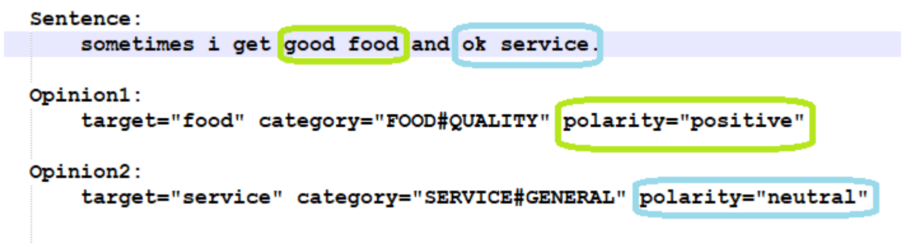
The task is to identify every entity E (HOTEL, RESTAURANT, FOOD…) and attribute A (GENERAL, DESIGN, PRICE, QUALITY…) pair towards which an opinion is expressed in the given text. a set E and A pairs is already defined for each domain, and the task is to determine the right one. In the example below there are two opinions addressing the same attribute but for different aspect terms. Here the first opinion is talking about the price of a large something, and the other is talking about the price of 'toppings'



```
Sentence:
    A large is $20  and toppings are about $3 each.

Opinion1:
    target="NULL" category="FOOD#PRICES" polarity="negative"

Opinion2:
    target="toppings" category="FOOD#PRICES" polarity="negative"
```

3) SLOT 3: Category Polarity Detection:

For each identified E#A, the task is to detect polarity of that given aspect. Possible values for this slot are: positive, negative, or neutral (mildly positive or mildly negative sentiment).



```
Sentence:
    sometimes i get good food and ok service.

Opinion1:
    target="food" category="FOOD#QUALITY" polarity="positive"

Opinion2:
    target="service" category="SERVICE#GENERAL" polarity="neutral"
```

Solution Approach

IV.A SLOT 2 (Opinion Target Extraction):

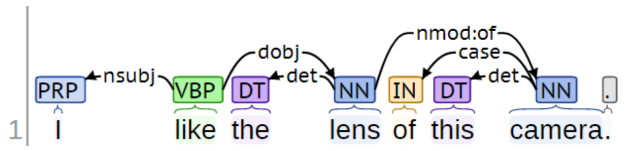
Preprocessing our data:

For both of our approaches, we must preprocess all textual data. Our preprocessing is all done using Stanford CoreNLP Parser. This strong tool can parse each sentence and returns its dependency tree, for which we are going to get information from, to extract aspect terms using our set of rules. Here are the steps of preprocessing:

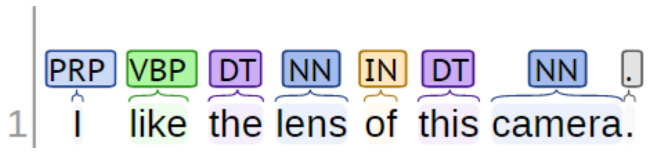
1 Segment each review to get the set of sentences

2 Using CoreNLP, parse and tokenize each sentence

3 Extract the dependency tree



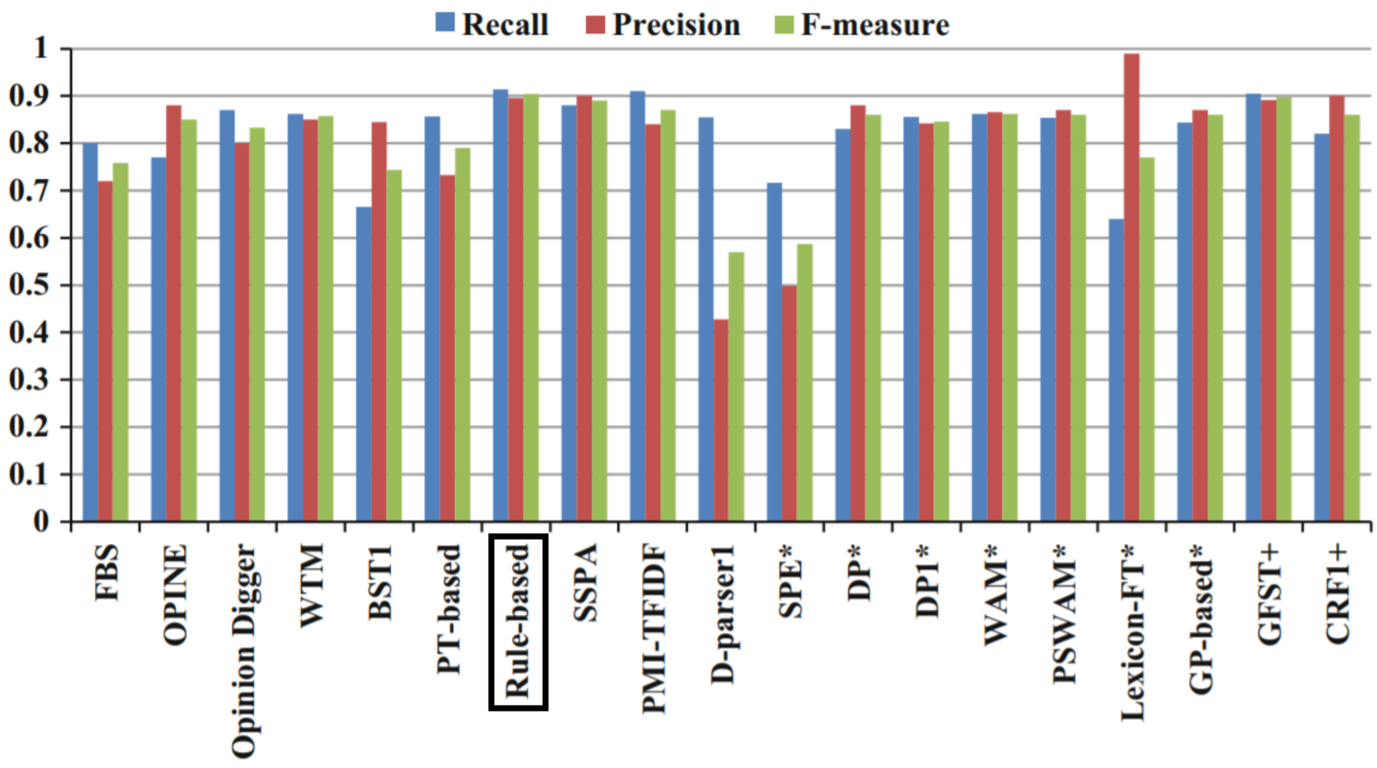
4 Extract POS-Tags for each token



1) Rule-Based Approach:

As mentioned earlier, this approach is based on an Aspect Parser proposed by SenticNet [2]. In their paper, SenticNet described 13 rules that exploit common-sense knowledge and assumptions on English language and uses sentence dependency trees to extract explicit and implicit aspect from a sentence. For our purpose, we are only interested in explicitly stated aspects, as for the implicit ones we can simply report the value 'NULL' for this SLOT.

Rule-Based approaches has already shown some very good performances in aspect extraction tasks. (Toqir et al., 2016) [4] has a paper about aspect extraction. In their paper, Toqir et al surveyed and compared different techniques for extracting aspects from online reviews. The reported scores of a rule-based approach by (Poria et al., 2014) [5] were very good (one of the best). Below is a comparison between different approaches for aspect extraction using customer review dataset[14]: (As reported in (Toqir et al., 2016))



The reason we chose to follow SenticNet's aspect parser, is its success in similar tasks. Below are evaluations scores of SenticNet's aspect parser tested on different datasets (As reported in their paper):

Performance on the SemEval 2014 dataset:

Domain	**Precision**	**Recall**
Laptop	82.15%	84.32%
Restaurants	85.21%	88.15%

[14] https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html

Performance on different reviews datasets provided by (Hu and Liu, 2004):

Dataset	Precision	Recall
Nikon Coolpix	82.15%	86.15%
Nokia-6610	93.25%	93.32%
Jukebox	89.25%	91.25%
Canon G3	90.15%	92.25%
DVD-player	92.25%	94.15%

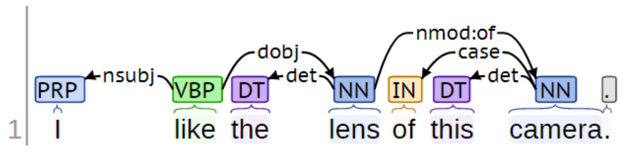
In order for us to use this rule-based system, an assumption has to be made: the reviews we are processing are grammatically correct. Because our system works on words dependencies between each other, and if a sentence is not grammatically correct, the dependency tree is likely to give inaccurate information about that sentence. There are two sets of rules:

1 Rules for sentences that have a subject verb.

2 Rules for sentences that doesn't have a subject verb.

Overall, we implemented 8 rules. For must rules, our system works in the following way:

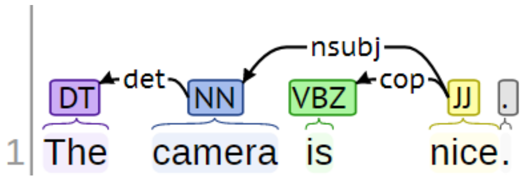
- Iterate over all tokens
- For each token, get its dependencies with other tokens. For example, here the word "lens" has two dependency relations with two different words: "the" and "like". In the case of 'dobj' relation (Direct Object dependency relation), the token 'lens' is the dependent of this relation, and "like" is the governor or head of this relation.



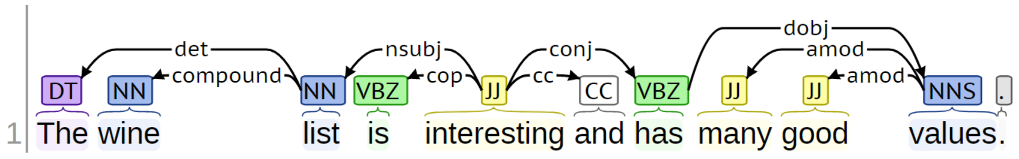
- Go over the rules, if a certain rule applies on some dependency relation (It has a direct object relation with a noun, for example), we extract the current token, or in some cases we extract the other side of relation.

Below are a couple of examples of rules we used to extract aspects:

- If a current token T is in a ‘nsubj’ (Noun Subject) relation with a token N, and N is in a ‘cop’ (Copula relation) with a copula verb (is, am, are, was …etc.), and the current token T is a noun, then we can extract T as an aspect. In this example, T is “camera” and T is in ‘nsubj’ with N “nice”, N is in ‘cop’ relation with a copula verb “is”, and T is a noun, we extract “camera” as an aspect.



- If we token T was extracted as an aspect, and T has a compound modifier (“compound”) M, and M is a noun, we extract ‘T-M’ as an aspect, and we remove T from the aspects list. In the example sentence below: “list” has a noun compound modifier “wine”, so we extract “wine list” as an aspect.



2) Supervised Machine Learning Approach:

In this approach, we used a supervised learning model/algorithm, which is Conditional Random Field. The reason we chose CRF is its success in similar tasks. Many teams in SemEval 2016 and SemEval 2015 used CRFs for this specific subtask (OTE Identification). As an input for CRF model, we have to transform our sentences and build features and use them as our input. Below is an example for the first item/token of an input sequence/sentence, and its features.

```
['Judging',
 'word.chunk=B-VP',
 'word.lower=judging',
 'word.orthographic=True',
 'word.lemma=judge',
 'first_4_letters=Judg',
 'first_3_letters=Jud',
 'first_2_letters=Ju',
 'last_2_letters=ng',
 'last_3_letters=ing',
 'last_4_letters=ging',
 'word.isupper=False',
 'word.polarity=mild',
 'word.obj=high',
 'postag=VBG',
 'judging',
 'judgment',
 'judgement',
```

As an output, we also have to transform the tokenized sentence into a sequence of IOB labels where each label is either:

1 **B:** beginning of target expression

2 **I:** inside the target expression

3 **O:** outside the target expression

```
 The wine list is interesting and has many good values.
 target="wine list"
 The:O  wine:B  list:I  is:O  interesting:O  and:O  has:O  many:O  good:O  values:O  .:O
 O        B       I      O        O            O      O      O      O        O        O
```

Using this format of labels, we transformed our task into a sequence labeling/classification task and made it possible for us CRF for extracting OTEs.

Team IIT-TUDA [6] are participants in SemEval 2016 Task5. In their they used CRFs for extracting OTEs. In their paper, they suggested to use several features for this task, we added many of those to our list features, we also added other features that can make an improvement to our model.

List of features for OTE identification:

For each token in a sentence, we build the current features:

- **Token:** each word in a sentence is used as a feature, we do not remove stop words or punctuation marks.
- **Lowercased token:**

```
Token = "Judging"  ==>   Feature = "judging"
```

- **POS tag for token:**

```
Token = "Judging"  ==>   Feature = "VBG"
```

- **Prefix and suffix:** for each token, all its prefixes and suffixes of length four and below are taken as features.

```
Token = "Judging"   ==>   Feature = ["ju", "jud", "judg", "ng", "ing", "ging"]
```

- **IsUpper**: as a binary feature, if token's characters are all in uppercase or not.

```
Token = "FOOD"   ==>   Feature = "True"
Token = "FOoD"   ==>   Feature = "False"
```

- **Orthographic:** a binary feature, if the token starts a capitalized letter.

```
Token = "Food"   ==>   Feature = "True"
```

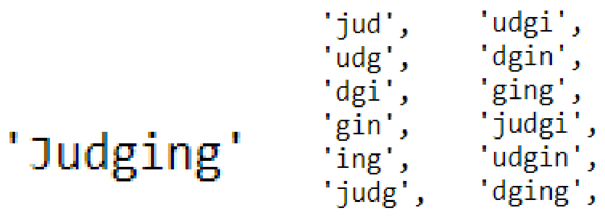
- **Word shape:**

```
Token = "gOoD32"   ==>   Feature = "xXxX##"
```

- **Lemmatized token:** we use CoreNLP to generate lemmas for each token.

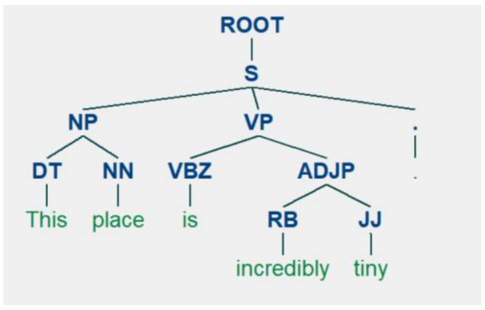
```
Token = "Judging"   ==>   Feature = "judge"
```

- **Polarity and Objectivity scores:** using SentiWordNet, we calculate polarity and objectivity score for each token, and use them as features. Possible values for these features: 'Neutral', 'Mild', 'High'.
- **Substrings N-grams:** all substrings of the current token of size 5 and below are used as features.



```
            'jud',   'udgi',
            'udg',   'dgin',
            'dgi',   'ging',
            'gin',   'judgi',
'Judging'   'ing',   'udgin',
            'judg',  'dging',
```

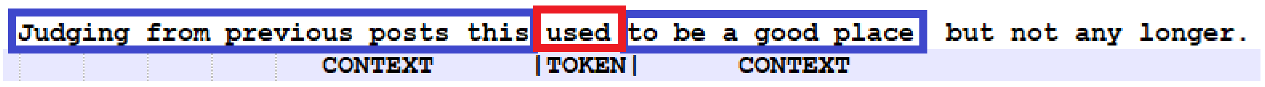
- **Chunk information:** using CoreNLP we can get a chunked parse tree, from which we can get chunk information for each token. We consider all types of chunks (Noun Phrases, Verb Phrases…etc.). In the example below: "This" chunk information is: 'B-NP', and "place" chunk information is: 'I-NP'.



- **Beginning of sentence feature:** binary feature, indicates if a token is the first word of a sentence.

- **End of sentence feature:** binary feature, indicates if a token is the last word of a sentence.

- **Context:** for a token we get the preceding 5 tokens and the following 5 tokens.


Judging from previous posts this used to be a good place but not any longer.
CONTEXT
|TOKEN|
CONTEXT

As features for each token in the context window we add:

1. Token itself
2. Lowercased token
3. POS tag of token
4. Polarity score
5. Objectivity score

For an example features matrix, refer to appendices B and C.

Training CRF:

After building the feature matrix and the output sequence labels, our remaining task is to train CRF model using the training dataset provided by SemEval, and test this model on the un-seen testing dataset.

We used CRFsuite for Python. CRFsuite makes it really easy to train a CRF model, once you have your features in the right form. For hyperparameters tuning, we perform 5-fold cross-validation on the provided training data.

- c1: The coefficient for L1 regularization= 0.0 (no L1 regularization)
- c2: The coefficient for L2 regularization = 0.05
- max_iterations = 50
- algorithm = lbfgs[15]

[15] Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) method

IV.B SLOT 1 (Aspect Category Detection):

One-vs-Rest approach with SVMs

Computing TF-IDF matrix and preprocessing:

To start we need to calculate TF-IDF matrix that will be used training input for the classifier. We used Sklearn's Tfidf-Vectorizer for that purpose. We pass the training data to the vectorizer. It first learns the vocabulary of the training data, then, transform it to term-document matrix. Using tfidf-vectorizer, we also transform all tokens to lowercase and filter-out English stop-words.

Setting up One-vs-Rest classifier and training:

Our next step is to implement our system classifier. The restaurant reviews database has 12 aspect classes/categories (FOOD#QUALITY, FOOD#STYLE_OPTIONS, RESTAURANT#GENERAL …etc.) so we will have 12 classifiers, one for each class. When training each classifier, the samples of that class will be positive and samples from other classes will be negative. In other words, this classifier will only recognize sentences that have an aspect of that specific category. We used Sklearn's SVM with an RBF[16] kernel for that purpose. We set probability option to True, the classifier then returns the probability of a sentence to belong to certain class. For example: "The wine list is interesting and has many good values" (DRINKS# STYLE_OPTIONS, 0.52) (DRINKS#PRICES, 0.65) (SERVICE#GENERAL, 0.091). In the previous example we know there's high probablity this sentence has an opinion about drink prices. There is also a good probability there's another opinion about drink styles. There's also a very low possibility that this sentence has an opinion talking about restaurant's service.
Note: for sentences that has more than one opinion (like the example above), we train the classifiers once for each aspect category in that sentence. For example: "The food was lousy - too sweet or too salty and the portions tiny." (FOOD#QUALITY) (FOOD#STYLE_OPTIONS), this sentence is passed twice to the classifiers:
"The food was lousy - too sweet or too salty and the portions tiny." (FOOD#QUALITY)
"The food was lousy - too sweet or too salty and the portions tiny." (FOOD#STYLE_OPTIONS)

[16] Radial Basis Function (Gaussian): $\exp(-\| x - x' \|^2/2\sigma^2)$

After training all classifiers, we then use a threshold to decide which categories will be assigned to sentences. The threshold we used is 0.25. To determine this value for the threshold we performed 5-fold cross-validation on the training set. Refer to appendix F for training and testing process.

One-vs-Rest approach with CNNs

Preparing input for deep neural networks:

As input to all our deep learning models, we build an input vector for each sentence. Each sentence is represented by the concatenated word vectors (embeddings) of all words in that sentence.

We have 2 types of embeddings:

1. Pretrained embeddings:
    a. GoogleNews-word2vec[17]
    b. Common Crawl (glove.840B.300d)[18]
2. Embeddings learned from the training dataset

We have separately trained all our models using those different embeddings.

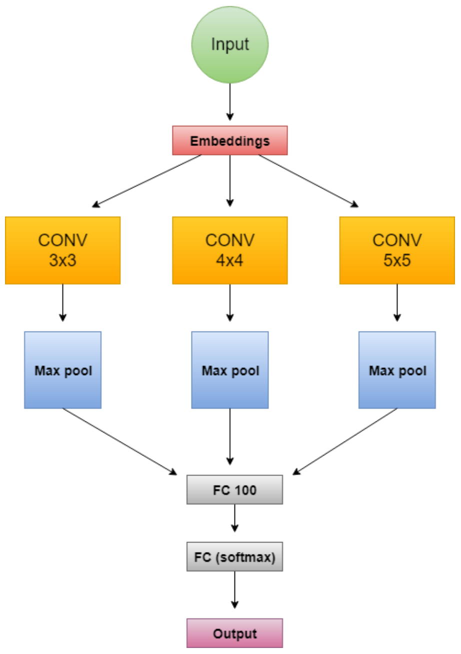
We also used multi-channel embeddings as input (all embeddings in the same time). This way every sentence in the corpus is represented by different embeddings.

CNN architecture:

Our CNN model is based on the model architecture proposed by Kim (2014) [7]

First a convolution operation is done on the input vectors to extract features. This is done by applying filters. There are filter of different sizes and n number of filter maps. Next, a max-pooling operation is applied to feature maps (convolution layers output). Max-pooling is used to capture the most important feature in each feature map. The pooled output is then fed to a fully connected layer with n neurons and a Rectified Linear Unit (ReLU) activation function. The output is then passed to a fully connected SoftMax layer, in which the probability distributions are calculated.

[17] https://code.google.com/archive/p/word2vec/
[18] https://nlp.stanford.edu/projects/glove/



Regularization:

To prevent overfitting the training data we employ the dropout technique with a keep probability of P. We also use L2-regularization to penalize the model when the model's weights (parameters) gets really high and it starts to memorize.

Model training and hyperparameters:

For training, Adam optimization algorithm is applied with Adam Optimizer on shuffled batches of the training set. Cross-entropy is used as the loss function. We use filter sizes of: 3x3, 4x4 and 5x5, and 20 feature maps for each filter size, dropout probability of 0.5, batch size of 128, and a 100 neuron in the fully connected layer.

Similar to the previous approach, we train a classifier for each label, and we generate the class probabilities and then using a threshold we assign the labels to each sentence. Only in this model we use a different threshold for each class classifier. We start with a threshold value of 0.4 for all classifiers (English restaurant dataset), and then we check the classifiers which have low accuracy on the training set, and for those classifiers we perform grid-search validation and decide on the best threshold for each class classifier.

Single deep CNN classifier

For this approach, we use the exact same CNN architecture and hyperparameters. The only difference is we use more filter maps for each filter size (we use 100).
We then train our model on the multi-label data and generate the class probability for each class using sigmoid, and then we use a threshold to assign the labels.

IV.C SLOT 3 (Aspect Polarity Estimation):

Single SVM classifier

Computing TF-IDF matrix and adding features:

Similar to SLOT1 approach, we first calculate the TF-IDF matrix by passing all sentences in the training data. We also transform all tokens into lower-case. We also extend this matrix to add two new columns/features to our input:

- Category feature: an integer valued feature where a number between 1-12 indicates the category of that opinion tuple.
- OTE: is a binary feature for our target, this feature has a value 1 if the OTE of that opinion tuple is explicitly stated in the sentence (not 'NULL') and 0 if it's implicit ('NULL').

Training classifier and model parameter optimization:

As we mentioned before, we use SVM as our classifier. For each opinion tuple in a sentence (target, category) we only have polarity value, so our classification task is pretty straight-forward. Since we don't any classification probabilities for this task, we can use only one SVM with 'OVR' option (one-vs-rest) that can handle this multi-class classification task (positive, negative, neutral). We then perform a 5-fold cross-validation to optimize two main parameters:

- ✓ **SVM hyper-parameters:** SVM has many parameters, C[19], kernel function[20], and Gamma[21].
- ✓ **Number of max features in TF-IDF:** the size of tf-idf's vocabulary where only top N features are considered (ordered by term frequency across the whole corpus/dataset)

Optimized Parameters:

- Kernel type: Linear

[19] C: Parameter for misclassification
[20] Kernel Function: type of function used for computing kernel matrix (linear, RBF, polynomial…etc.)
[21] Gamma: kernel coefficient for some types of functions. (Gamma is the inverse of variance in the RBF case)

- C = 1
- N_max_features = 2000

A demonstration for the applied cross-validation is available in appendix G.

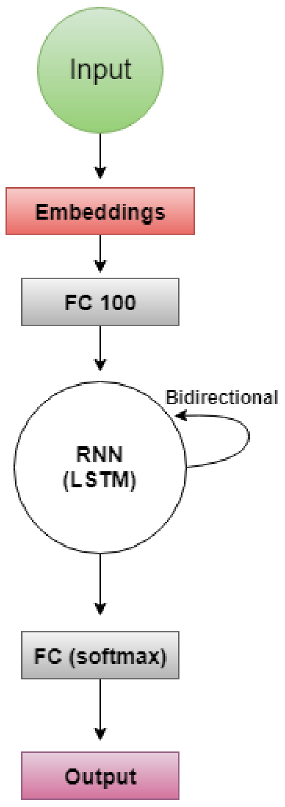
Single deep CNN classifier

We will use the same network architecture and the same hyper-parameters from our previous task. We then perform training and generate class probability and assign the label from the highest probability.

Single RNN classifier using bidirectional LSTM

Network architecture:

We have used Binh Do (2016)[22] proposed architecture for this subtask. Do shows good results for SLOT3 for SemEval 2016 and also good results in sentiment analysis tasks in other versions of SemEval.
First input embedding vectors are built. These vectors are then sent to a fully connected layer. Then the output of this layer is sent to a many-to-many bidirectional LSTM layer (forward layer, and backward layer). The output of the last time step is then passed through a fully connected layer with SoftMax to generate class probabilities.



[22] https://github.com/peace195/aspect-based-sentiment-analysis

Regularization:

L2 regularization and dropout is used on the hidden layer’s weights.

Model training and hyperparameters:

We use 248 hidden units in the forward and backward LSTM layers. 100 neurons were used in the fully connected layer. 0.5 is the dropout keep probability.

Results and Discussion

SemEval provides the evaluation and validation mechanism for each slot of the produced ABSA system.

- ### SLOT 2 Evaluation Method:

For SLOT 2, the evaluation assesses whether a system identifies and returns the set of targets that are used in a sentence to refer to specific aspects. Evaluation will be based on: precision, recall and F-1 scores. These scores are calculated by comparing the list of the targets that a system returned (for a sentence) to the corresponding gold list. The calculation discards NULL targets since they do not correspond to explicit target mentions. While evaluating we also ignore duplicate targets for the same sentence. Here's an example:

```
The food was lousy - too sweet or too salty and the portions tiny.

target="food" from="4" to="8"
target="portions" from="52" to="60"
```

The numbers (from, to) gives the position of the target. These number are used to make a list of targets and make the evaluation more precise in the case of the occurrence of a word twice in the same sentence.

```
targets = [(4,8),(52,60)]
```

- ### SLOT 1 Evaluation Method:

The evaluation assesses whether a system identifies and returns the set of aspect categories towards which an opinion is expressed. Similar to SLOT 1, precision, recall and F-1 scores are considered. These scores are calculated by comparing the list of the categories that the system returns (for a sentence) to the corresponding gold list. The calculations ignore duplicate categories for the same sentence. For example:

```
The food is very average...the Thai fusion stuff is a bit too sweet,
every thing they serve is too sweet here.

target="food" category="FOOD#QUALITY"
target="Thai fusion stuff" category="FOOD#QUALITY"
target="NULL" category="FOOD#QUALITY"
```

In the sentence above, only one category occurrence will be taken into consideration:

```
Categories = [FOOD#QUALITY]
```

• SLOT 3 Evaluation Method:

For polarity estimation, the total accuracy is used as only measure of performance. Total accuracy is calculated by comparing the polarity value of opinion tuples (target, category) of each sentence in the testing dataset.

• SLOT 2 Results (Rule-Based Approach):

Although the original work we followed (SenticNet aspect parser), had quite good performance on an earlier SemEval ABSA task (SemEval 2014), our system however had a low performance when tested on SemEval 2016 Task5 testing data. Here are the evaluation results:

- Recall Score: 0.258722
- Precision Score: 0.23116
- F1 Score: 0.24416564609436558

Saujanya et al (2016) implemented the same Sentic.Net rule-based approach and tested its performance on the same data we're using (SemEval 2016). The evaluation they got were really similar to ours. Here's the result they obtained: (As reported in their website[23])

- Recall Score: 0.2605156
- Precision Score: 0. 2570281
- F1 Score: 0.2587601

We also contacted Sentic.Net and asked them for some guidance on their aspect parser that we are trying to implement. The scores we got were much lower than expected. We figured we

[23] http://saujanyareddy.github.io/

could improve our model performance if we acquire more information about the aspect parser. But we didn't get a response for our request.

In the table below are shows the number of times each rule was used while running on Training data: (Note that this is not the number of aspects extracted by each rule. Some rules adjust or change an aspect extracted by other rules).

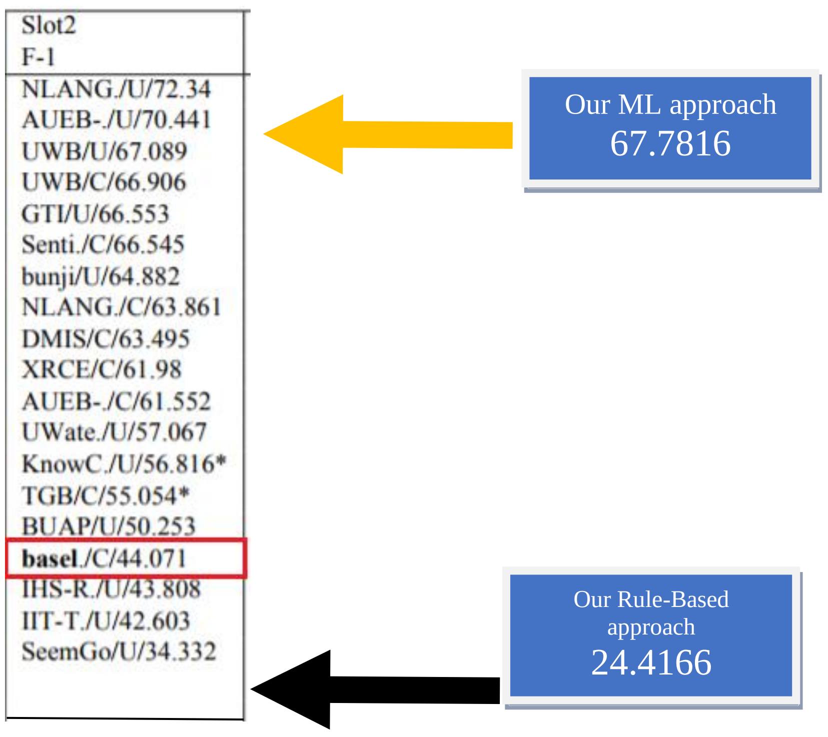
RULE	**rule_1**	**rule_2**	**rule_3**	**rule_4**	**rule_5**	**rule_6**	**rule_7**	**rule_8**
#used	790	266	62	41	1804	325	1065	1417

- ## SLOT 2 Results (Supervised Approach):

After performing cross-validation and optimizing CRF's hyperparameters, we test our system on the testing data provided. Here are the evaluation results:

  - Golden Aspect Terms = 612
  - System Aspect Terms = 535
  - Correct predictions = 413
  - Recall Score: 0.625000
  - Precision Score: 0.740385
  - F1 Score: 0.677816

And here's how our system fares comparing to other participants for the same slot:



Slot2 F-1
NLANG./U/72.34
AUEB-./U/70.441
UWB/U/67.089
UWB/C/66.906
GTI/U/66.553
Senti./C/66.545
bunji/U/64.882
NLANG./C/63.861
DMIS/C/63.495
XRCE/C/61.98
AUEB-./C/61.552
UWate./U/57.067
KnowC./U/56.816*
TGB/C/55.054*
BUAP/U/50.253
basel./C/44.071
IHS-R./U/43.808
IIT-T./U/42.603
SeemGo/U/34.332

Our ML approach
67.7816
Our Rule-Based approach
24.4166

I-O-B classes precision and recall scores are available in appendix D.

- **Slot 1 Results:**

Here's our models' performances on English restaurant dataset:

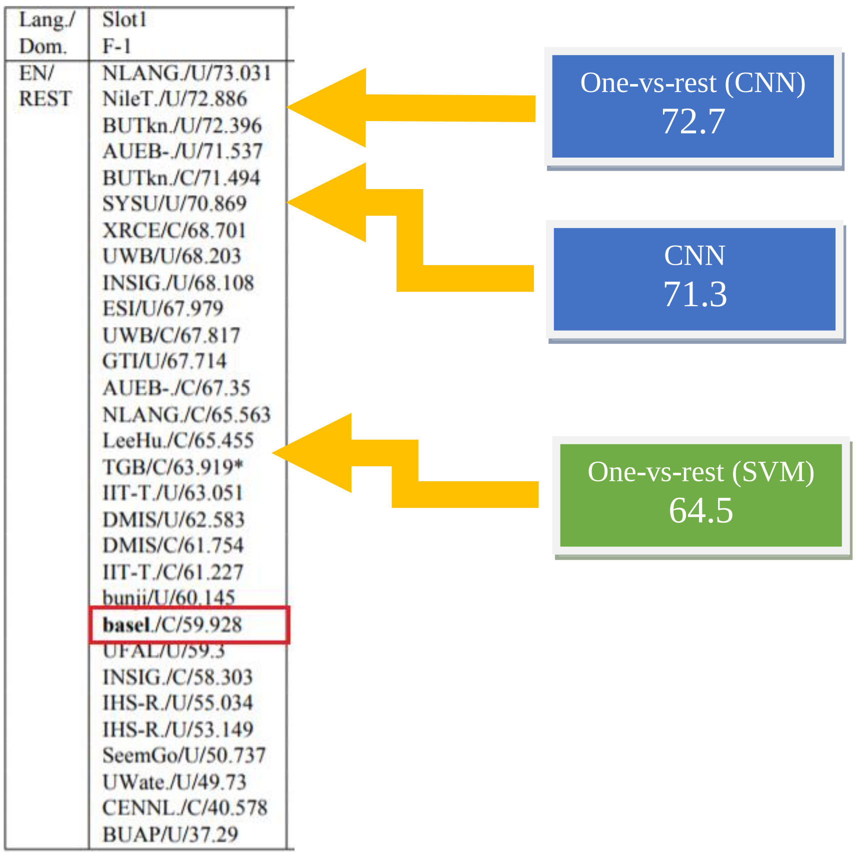
	Precision	**Recall**	**F1-Score**
One-vs-rest (SVM)	0.677	0.616	0.645
CNN (Google)	0.715	0.677	0.695
One-vs-rest (CNN)	0.764	0.693	**0.727**

and here's CNN performance using different embeddings on English restaurant dataset:

	F1-Score
CNN (Google_news)	**0.695**
CNN (Common_crawl)	0.658
CNN (non_trained)*	0.64
CNN (Google+Common_Crawl+non_trained)*	0.67
CNN (Google+non_trained)*	0.713

* Randomly initialized embeddings are learned from training dataset (longer training time is needed)

Below are the SLOT 1 results for all participating teams in SemEval 2016:



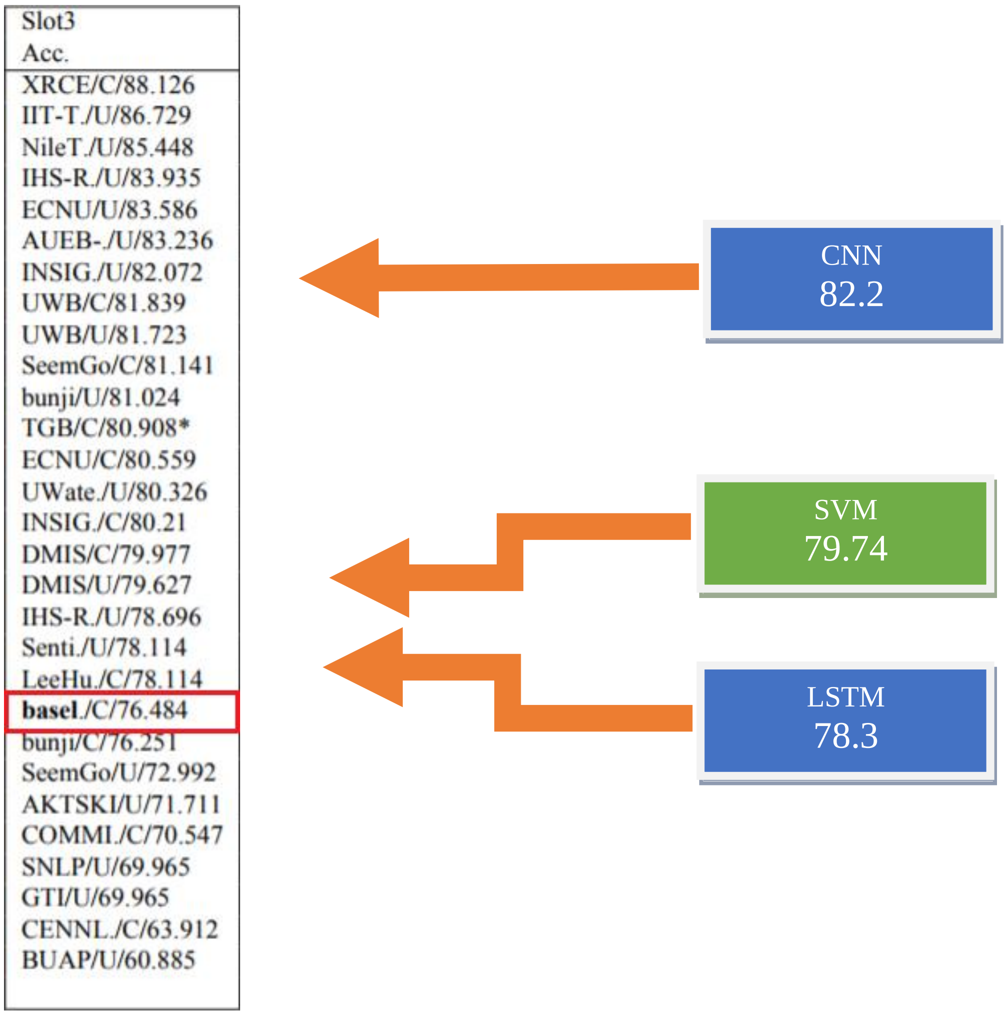
Lang./ Dom.	Slot1 F-1
EN/ REST	NLANG./U/73.031
	NileT./U/72.886
	BUTkn./U/72.396
	AUEB-./U/71.537
	BUTkn./C/71.494
	SYSU/U/70.869
	XRCE/C/68.701
	UWB/U/68.203
	INSIG./U/68.108
	ESI/U/67.979
	UWB/C/67.817
	GTI/U/67.714
	AUEB-./C/67.35
	NLANG./C/65.563
	LeeHu./C/65.455
	TGB/C/63.919*
	IIT-T./U/63.051
	DMIS/U/62.583
	DMIS/C/61.754
	IIT-T./C/61.227
	bunii/U/60.145
	basel./C/59.928
	UFAL/U/59.3
	INSIG./C/58.303
	IHS-R./U/55.034
	IHS-R./U/53.149
	SeemGo/U/50.737
	UWate./U/49.73
	CENNL./C/40.578
	BUAP/U/37.29

- ## Slot 3 results:

Here are our approaches performances:

	Accuracy
SVM	0.794
LSTM (Google)	0.783
CNN (Google)	0.822

Here's how our system faired compared with other teams results for Slot3:



Slot3 Acc.
XRCE/C/88.126
IIT-T./U/86.729
NileT./U/85.448
IHS-R./U/83.935
ECNU/U/83.586
AUEB-./U/83.236
INSIG./U/82.072
UWB/C/81.839
UWB/U/81.723
SeemGo/C/81.141
bunji/U/81.024
TGB/C/80.908*
ECNU/C/80.559
UWate./U/80.326
INSIG./C/80.21
DMIS/C/79.977
DMIS/U/79.627
IHS-R./U/78.696
Senti./U/78.114
LeeHu./C/78.114
basel./C/76.484
bunji/C/76.251
SeemGo/U/72.992
AKTSKI/U/71.711
COMMI./C/70.547
SNLP/U/69.965
GTI/U/69.965
CENNL./C/63.912
BUAP/U/60.885

and here's a table of F1-scores of CNN using different embeddings for training:

	F1-Score
CNN (Google_news)	0.82
CNN (Common_crawl)	0.776
CNN (learned from corpus)*	0.745
CNN (Google+Common_Crawl+non_trained)*	0.79
CNN (Google+non_trained)*	0.80

Below is a detailed SVM classification evaluation report for each class in our data (Negative, Positive, Neutral):

```
             Precision    Recall  F1-score   Instances

   Positive     0.8748    0.8920    0.8833       611
   Negative     0.5931    0.6716    0.6299       204
    Neutral     0.6000    0.0682    0.1224        44

avg / total     0.7938    0.7974    0.7841       859
```

We can see that the recall of 'Neutral' class is very low. We think that better scores can be obtained if we have a more balanced training dataset. Out of 2507 polarity label in the training dataset (ENG Restaurant), only 101 are of "neutral" class (4% of the dataset).

Also, here's CNN performance on different datasets compared with other participants:

Arabic hotel reviews:

Embeddings pretrained on Arabic Wikipedia are used.

	F1-Score
INSIG	0.827
Our model (CNN)	**0.818**
IIT-T	0.817
Baseline	0.764

Turkish restaurant reviews:

Embeddings pretrained on Turkish Wikipedia are used.

	F1-Score
IIT-T	0.842
Our model (CNN)	**0.754**
INSIG	0.742
Baseline	0.723

English laptops reviews:

	F1-Score
IIT-T	0.827
INSIG	0.784

ECNU	0.781
IHS-R	0.779
NileT	0.774
AUEB	0.769
Our model (CNN)	**0.76**
LeeHu	0.759
9 other partcipants...	
Baseline	0.7
6 other partcipants...	

VI. Related Work

Aspect-Based Sentiment Analysis has several subtasks. SemEval has done ABSA task for 3 years (2014, 2015, 2016) throughout these years, there have been a lot of participants with a variety of approaches. Generally, the proposed systems for ABSA are split into groups: systems that have different approaches for aspect detection (OTE, E#T), and polarity estimation, our system falls into this category. Others have a joint model (Unsupervised for example) that can be used to get both aspects and their polarities.

In previous SemEval versions, many participants including us, used machine learning algorithms. CRF for example, was used by many teams for OTE extraction: (Toh and Wang, 2014) [8] CRFs with lexical information, bigrams and POS as features. (Team AUEB, 2016) [9] also used CRFs and came in second in SLOT2 results of SemEval 2016. SVMs were also a favorite choice for many teams for aspect category and polarity detection: (Wagner et al., 2014) [10] used SVM along with Bag-of-N-gram Features, and (Kiritchenko et al., 2014) [11] had good scores using SVMs combined with rich linguistic features including dependency parsing.

Deep Learning has also been put to the test for this task. (Wang et al., 2015) [12] used CNN for aspect-based analysis of SemEval-2015 ABSA data and reported performance comparable to top systems of the 2015 task.

Our system has many similarities with other systems, especially ones that handled OTE extraction task as Multi-Class classification task and used lexical and syntactic features. Also, with the ones who used one-vs-rest classification for multi-label classification tasks.

VII. Conclusion and Future Work

Impact of the project:

Below is the project's impact on many domains. Note that sentiment analysis is an evolving topic. This project might also affect other domains in the future.

a. **Economics**:
Sentiment Analysis can be a useful source of information for determining marketing strategy and improving product messaging and customer service.

b. **Impact on Environment**:
This project has no impact environment.

c. **Sustainability**:
This project has no impact on sustainability

d. **Manufacturability**:
This project has no impact on manufacturability

e. **Ethics**:
This project has no impact on ethics

f. **Health**:
This project has no impact on health

g. **Security**:
SemEval's datasets doesn't include any information about the review's writer. Hence, this project has no impact on security

h. **Social and political issues**:

If applied on social media data, like tweets, Sentiment Analysis can provide an insight or a measure on the emotions behind social media mentions on a specific social topic or an incident. whether people are generally happy/sad, positive/negative, optimistic/pessimistic about a specific topic.

Conclusion and future work:

In this report, we described our system for SemEval 2016 Task 5.

For OTE extraction (SLOT 2), we used both a rule-based approach which is based dependency relations, and a supervised machine learning approach using CRF with many features for sequence labeling. For aspect category detection (SLOT 1), we trained a one-vs-rest classifiers using SVMs and CNNs, and we used thresholding to decide which categories should be assigned to a certain a sentence. We also used a single CNN multi-label classifier. For polarity estimation/detection (SLOT 3) we performed a simple polarity classification on all opinion tuples, using SVMs, CNNs, and LSTMs.

Our experiments showed that the rule-based model for extracting OTEs was not complex enough to identify target expressions from sentences and we probably need to add more rules to improve its performance. On the other hand, our supervised model performance confirmed the effectiveness of CRFs in this subtask and obtained good results in SLOT2 evaluations.

Our SLOT1 one-vs-rest using CNN approach was pretty successful compared to other competitors (top 3). SLOT3 results were above average. Our CNN classifier performed well. However, LSTM's results were barely above average.

Future work will focus more on other deep learning approaches especially for SLOT3 (Sentiment analysis) as we expect better results from CNN and LSTM. We will try different models and architectures. Also, another potential improvement to our deep learning models is using FastText to generate embeddings for words that are not in the pretrained embeddings vocabulary. This is an important step because almost 12% of the words are out-of-vocabulary words for English restaurant dataset.

We are also planning to improve our SLOT2 scores. We believe that adding more features to our system and performing feature-selection on CRF can increase its performance.

Acknowledgements

I would like to thank and express my gratitude to this project supervisor: Assistant Professor Reyyan Yeniterzi on her guidance and support throughout this project.

References

[1] Thet, T. T. & Na J. C. & Khoo C.S.G (2010) Aspect-based sentiment analysis of movie reviews on discussion boards. Journal of Information Science archive Volume 36 Issue 6 pp. 823-848 Sage Publications, Inc. Thousand Oaks, CA, USA

[2] Soujanya P. & Erik C. & Lun-Wei K. & Chen G. & Alexander G. A Rule-Based Approach to Aspect Extraction from Product Reviews: http://sentic.net/aspect-parser.pdf, Licence details: http://creativecommons.org/licenses/by/4.0

[3] Maria Pontiki; Dimitris Galanis; Haris Papageorgiou; Ion Androutsopoulos; Suresh Manandhar; Mohammad AL-Smadi; Mahmoud Al-Ayyoub; Yanyan Zhao; Bing Qin; Orphee De Clercq; Veronique Hoste; Marianna Apidianaki; Xavier Tannier; Natalia Loukachevitch; Evgeniy Kotelnikov; Núria Bel; Salud María Jiménez-Zafra; Gülşen Eryiğit. SemEval-2016 Task 5: Aspect Based Sentiment Analysis

[4] Rana, Toqir Ahmad and Yu-N Cheah. "Aspect extraction in sentiment analysis: comparative analysis and survey." Artificial Intelligence Review 46 (2016): 459-483.

[5] Poria S, Cambria E, Ku L-W, Gui C, Gelbukh A (2014) A rule-based approach to aspect extraction from product reviews. In: Proceedings of the second workshop on natural language processing for social media (SocialNLP), pp 28–37

[6] Ayush K. & Sarah K. & Amit K. & Asif E. & Chris B. (2016) IIT-TUDA at SemEval-2016 Task 5: Beyond Sentiment Lexicon: Combining Domain Dependency and Distributional Semantics Features for Aspect Based Sentiment Analysis. Proceedings of SemEval-2016, pages 1129–1135, San Diego, California, June 16-17, 2016.

[7] Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), 1746– 1751

[8]Zhiqiang Toh and Wenting Wang. 2014. Dlirec: Aspect term extraction and term polarity classification system. In Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 235– 240, Dublin, Ireland, August. Association for Computational Linguistics and Dublin City University.

[9] Androutsopoulos, I., Malakasiotis, P., Pavlopoulos, J., Theodorakakos, P., & Xenos, D. (2016). AUEB-ABSA at SemEval-2016 Task 5: Ensembles of Classifiers and Embeddings for Aspect Based Sentiment Analysis. *SemEval@NAACL-HLT*.

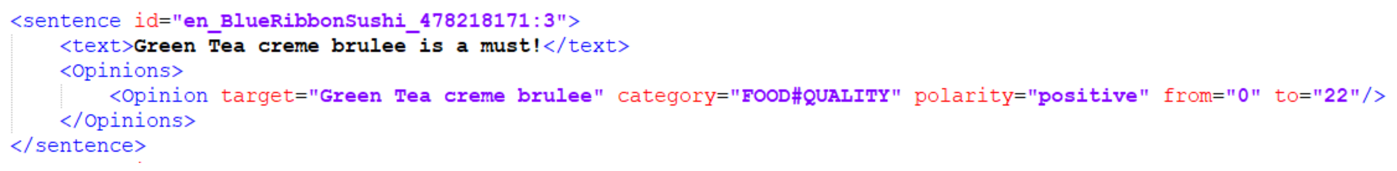
[10] Joachim Wagner, Piyush Arora, Santiago Cortes, Utsab Barman, Dasha Bogdanova, Jennifer Foster, and Lamia Tounsi. 2014. Dcu: Aspect-based polarity classification for semeval task 4. In Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 392–397, Dublin, Ireland, August. Association for Computational Linguistics and Dublin City University.

[11] Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. 2014. Nrc-canada-2014: Detecting aspects and sentiment in customer reviews. In Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 437–442, Dublin, Ireland, August. Association for Computational Linguistics and Dublin City University

[12] Bo Wang and Min Liu. 2015. Deep Learning For AspectBased Sentiment Analysis. Stanford University report, https://cs224d.stanford.edu/reports/WangBo.pdf.

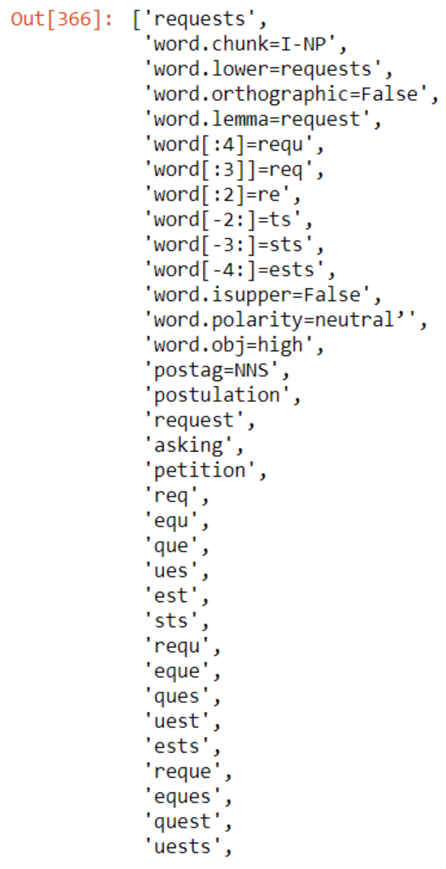
Appendix

A. *Given datasets:*



```
<sentence id="en_BlueRibbonSushi_478218171:3">
    <text>Green Tea creme brulee is a must!</text>
    <Opinions>
        <Opinion target="Green Tea creme brulee" category="FOOD#QUALITY" polarity="positive" from="0" to="22"/>
    </Opinions>
</sentence>
```

B. *Syntactic and lexical features input:*



```
Out[366]: ['requests',
           'word.chunk=I-NP',
           'word.lower=requests',
           'word.orthographic=False',
           'word.lemma=request',
           'word[:4]=requ',
           'word[:3]]=req',
           'word[:2]=re',
           'word[-2:]=ts',
           'word[-3:]=sts',
           'word[-4:]=ests',
           'word.isupper=False',
           'word.polarity=neutral’',
           'word.obj=high',
           'postag=NNS',
           'postulation',
           'request',
           'asking',
           'petition',
           'req',
           'equ',
           'que',
           'ues',
           'est',
           'sts',
           'requ',
           'eque',
           'ques',
           'uest',
           'ests',
           'reque',
           'eques',
           'quest',
           'uests',
```

C. ***Context features input:***

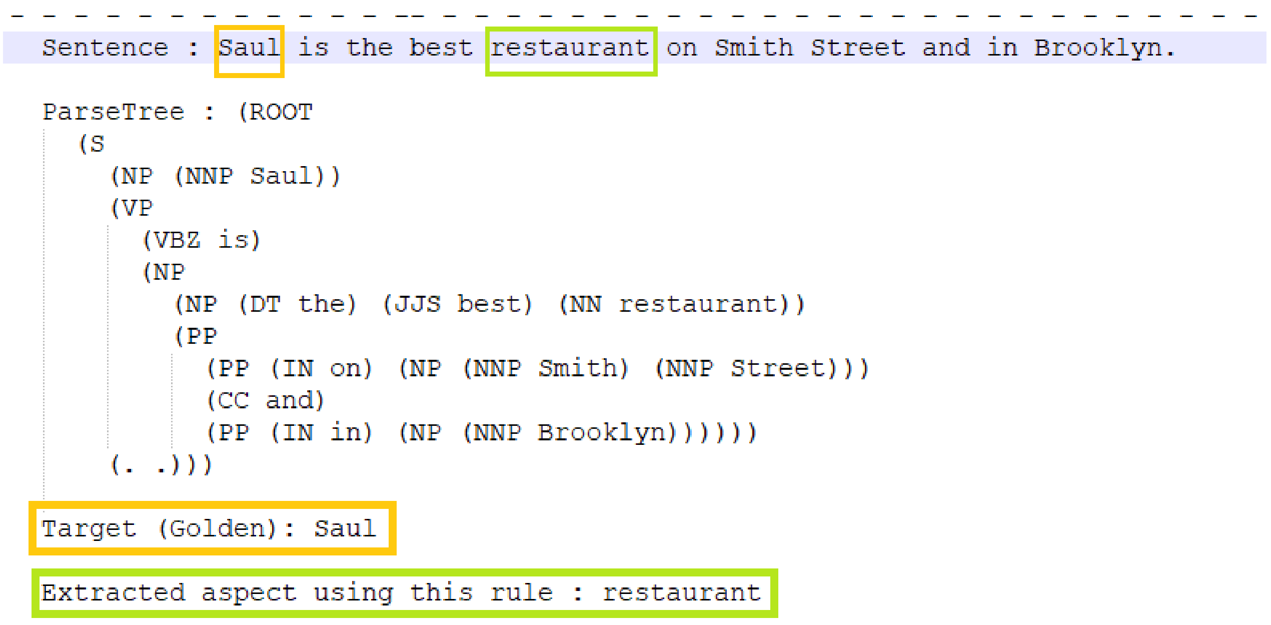
```
['-5:word.lower=complimentary'
 '-5:word.istitle=False',
 '-5:word.isupper=False',
 '-5:word.isdigit=False',
 '-5:word.polarity=mild',
 '-5:word.obj=high',
 '-5:postag=JJ',
 '-4:word.lower=noodles',
 '-4:word.istitle=False',
 '-4:word.isupper=False',
 '-4:word.isdigit=False',
 '-4:word.polarity=neutral’',
 '-4:word.obj=high',
 '-4:postag=NNS',
 '-3:word.lower=,',
 '-3:word.istitle=False',
 '-3:word.isupper=False',
 '-3:word.isdigit=False',
 '-3:word.polarity=neutral’',
 '-3:word.obj=neutral’',
 '-3:postag=,',
 '-2:word.lower=ignored',
 '-2:word.istitle=False',
 '-2:word.isupper=False',
 '-2:word.isdigit=False',
 '-2:word.polarity=mild',
 '-2:word.obj=high',
 '-2:postag=VBN',
 '-1:word.lower=repeated',
 '-1:word.istitle=False',
 '-1:word.isupper=False',
 '-1:word.isdigit=False',
 '-1:word.polarity=neutral’',
 '-1:word.obj=high',
 '-1:postag=VBN',
```

```
['1:word.lower=for',
 '1:word.istitle=False',
 '1:word.isupper=False',
 '1:word.isdigit=False',
 '1:word.polarity=neutral’',
 '1:word.obj=neutral’',
 '1:postag=IN',
 '2:word.lower=sugar',
 '2:word.istitle=False',
 '2:word.isupper=False',
 '2:word.isdigit=False',
 '2:word.polarity=neutral’',
 '2:word.obj=high',
 '2:postag=NN',
 '3:word.lower=,',
 '3:word.istitle=False',
 '3:word.isupper=False',
 '3:word.isdigit=False',
 '3:word.polarity=neutral’',
 '3:word.obj=neutral’',
 '3:postag=,',
 '4:word.lower=and',
 '4:word.istitle=False',
 '4:word.isupper=False',
 '4:word.isdigit=False',
 '4:word.polarity=neutral’',
 '4:word.obj=neutral’',
 '4:postag=CC',
 '5:word.lower=threw',
 '5:word.istitle=False',
 '5:word.isupper=False',
 '5:word.isdigit=False',
 '5:word.polarity=neutral’',
 '5:word.obj=high',
 '5:postag=VBD']
```

D. *IOB labels: precision recall and f1 scores:*

```
             precision    recall  f1-score   support

          B       0.80      0.70      0.75       616
          I       0.66      0.49      0.56       312
          O       0.96      0.98      0.97      7479

avg / total       0.94      0.94      0.94      8407
```

E. *Rule-Based system debugging:*



```
Sentence : Saul is the best restaurant on Smith Street and in Brooklyn.

ParseTree : (ROOT
  (S
    (NP (NNP Saul))
    (VP
      (VBZ is)
      (NP
        (NP (DT the) (JJS best) (NN restaurant))
        (PP
          (PP (IN on) (NP (NNP Smith) (NNP Street)))
          (CC and)
          (PP (IN in) (NP (NNP Brooklyn))))))
    (. .)))

Target (Golden): Saul

Extracted aspect using this rule : restaurant
```

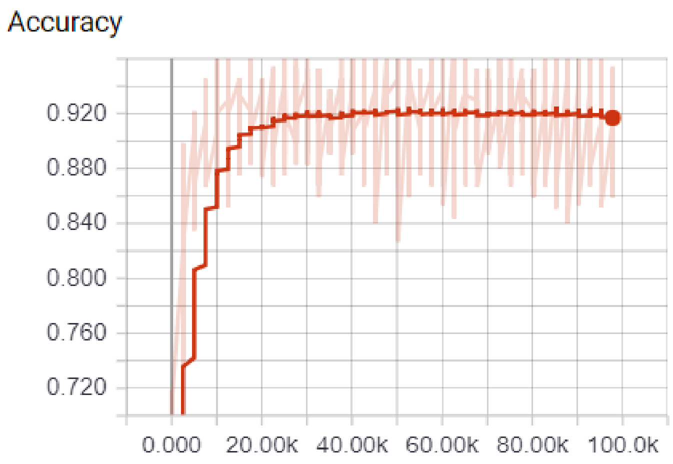
F. *Training 12 classifiers (OVR) and testing for SLOT1:*

```
Training classifier for class: AMBIENCE#GENERAL
Training classifier for class: DRINKS#PRICES
Training classifier for class: DRINKS#QUALITY
Training classifier for class: DRINKS#STYLE_OPTIONS
Training classifier for class: FOOD#PRICES
Training classifier for class: FOOD#QUALITY
Training classifier for class: FOOD#STYLE_OPTIONS
Training classifier for class: LOCATION#GENERAL
Training classifier for class: RESTAURANT#GENERAL
Training classifier for class: RESTAURANT#MISCELLANEOUS
Training classifier for class: RESTAURANT#PRICES
Training classifier for class: SERVICE#GENERAL

Testing our model on Restaurant Test dataset

Selecting the clf with confidence higher than threshold for each sentence.

Evaluation scores:
Recall = 0.6191117
Precision = 0.6744868
F1-Score = 0.645614
```

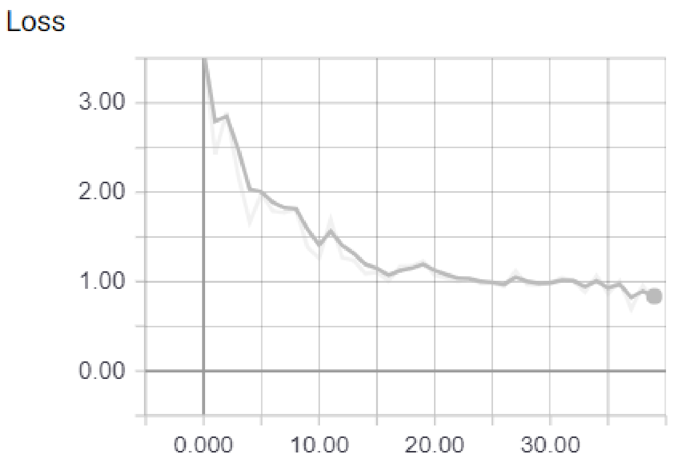
G. *SLOT3 SVM hyper-parameter optimization using cross-validation:*

```
# Performing 5-fold Grid Search Cross-Validation
# Tuning hyper-parameters for Accuracy scores


Best parameters set found on development set:

{'C': 1, 'kernel': 'linear'}

Grid scores on development set:

Average accuracy = 0.661 (+/-0.001) for {'C': 1, 'gamma': 0.001, 'kernel': 'rbf'}
Average accuracy = 0.661 (+/-0.001) for {'C': 1, 'gamma': 0.0001, 'kernel': 'rbf'}
Average accuracy = 0.661 (+/-0.001) for {'C': 10, 'gamma': 0.001, 'kernel': 'rbf'}
Average accuracy = 0.661 (+/-0.001) for {'C': 10, 'gamma': 0.0001, 'kernel': 'rbf'}
Average accuracy = 0.714 (+/-0.041) for {'C': 100, 'gamma': 0.001, 'kernel': 'rbf'}
Average accuracy = 0.661 (+/-0.001) for {'C': 100, 'gamma': 0.0001, 'kernel': 'rbf'}
Average accuracy = 0.742 (+/-0.041) for {'C': 1000, 'gamma': 0.001, 'kernel': 'rbf'}
Average accuracy = 0.708 (+/-0.033) for {'C': 1000, 'gamma': 0.0001, 'kernel': 'rbf'}
Average accuracy = 0.751 (+/-0.041) for {'C': 1, 'kernel': 'linear'}
Average accuracy = 0.725 (+/-0.050) for {'C': 10, 'kernel': 'linear'}
Average accuracy = 0.716 (+/-0.035) for {'C': 100, 'kernel': 'linear'}
Average accuracy = 0.705 (+/-0.028) for {'C': 1000, 'kernel': 'linear'}
```

H. *SLOT3 CNN training accuracy:*



I. *SLOT3 CNN training loss:*


Loss