Plagiarism Scan Report

Summary	
Report Genrated Date	21 Apr, 2018
Plagiarism Status	**100% Unique**
Total Words	1000
Total Characters	6436
Any Ignore Url Used	

Content Checked For Plagiarism:

Typed dependencies rules used for aspect sentiment extraction are:
Nominal Subject (nsubj) Rule:

When the sentence has a copular verb, the nsubj dependency has an adjective governor and noun dependent. Hence, here the aspect is the dependent and the sentiment is the governor.

Adjectival Modifier (amod) Rule:

When the sentence has a copular verb, the amod dependency has an noun governor and adjective dependent. Hence here the aspect is the governor and the sentiment is dependent.

Adverbial Complement (advcl) Rule:

When the governor is a non-copular verb for nsubj dependency as well as advcl dependency, the dependent of nsubj is the aspect and the dependent of advcl is the sentiment.

Nominal Subject + complement (acomp/xcomp) Rule:

When the governor is a non-copular verb for nsubj dependency as well as xcomp/acomp dependency, the dependent of nsubj is the aspect and the dependent of xcomp/acomp is the sentiment.

Adverbial Modifier (advmod) Rule:

When the governor is a non-copular verb for nsubj dependency as well as advmod dependency, the dependent of nsubj is the aspect and the dependent of advmod is the sentiment.

Direct Object (dobj) Rule:

When the governor is a non-copular verb for nsubj dependency as well as dobj dependency, the dependent of nsubj is the aspect and the dependent of dobj is the sentiment.

Passive Nominal Subject (nsubjpass) Rule:

When the sentence has a copular verb, the nsubj dependency has an adjective governor and noun dependent. Hence, here the aspect is the dependent and the sentiment is the governor.

Passive Nominal Subject (nsubjpass) + complement (acomp/xcomp) Rule:

When the governor is a non-copular verb for nsubj:pass dependency as well as xcomp/acomp dependency, the dependent of nsubj:pass is the aspect and the dependent of xcomp/acomp is the sentiment.

Relative Clause Modifier (acl:relcl) + complement (acomp/xcomp) Rule:

When the dependent is a non-copular verb for acl:relcl dependency as well as xcomp/acomp dependency, the governor of acl:relcl is the aspect and the dependent of xcomp/acomp is the sentiment.

Some rules are such that they need to be associated with each independent rule as they are the most general rules. We have implemented three such rules, namely the conjunction rule, the compound rule for bigrams and the negation rule. These rules are common across the primary rules and their module needs to be implemented as a function only once. This function is called regardless of which rule is triggered by the conditions satisfied. The description of the rules is as follows:
Conjunction Rule:
The conjunction rule is applicable when a review text contains multiple sentiment words associated with a single aspect.

In the above example, the conjunction rule is associated with the simple nsubj rule for copular verbs. The basic nsubj rule will only associate the adjective robust with the aspect interface. It will not understand that the adjective easy is also associated with the same aspect. This is where the conj dependency plays a pivotal role. Every word following robust which is separated by commas or conjunctions has a conj dependency with the word robust. In this case, the word easy is associated with robust with the conj dependency. Hence, easy is also associated with the aspect interface as its sentiment.

Negation Rule:
The negation rule is necessary because sometimes inverting terms are associated with the sentiment words. In such a case, a word of the opposite meaning of the negated sentiment word is added to the sentiment for the aspect.

In the above example, the negation rule is associated with the simple nsubj rule. Here, the aspect processor is associated with the sentiment good. Actually, the reviewer has written an inverting term not, before the sentiment word. If it was not for the neg dependency, the aspect-sentiment pair would be processor-good which is completely the opposite of what the reviewer means. The negation rule will help extract the true meaning of what the user has written. It looks for a word closest to the opposite polarity of the extracted sentiment word. This is done by using the getAntonyms() function of the textblob package. In this example, the word having the closest of the opposite polarity of good is bad. Hence, the word bad will be added as the sentiment of the aspect processor.

Compound Rule:
If multiple words are playing a role in naming the aspect or sentiment, both the words play a role in constructing the aspect-sentiment lexicon.

Here, the Compound rule is associated with the simple nsubj rule. The simple nsubj rule extracts the noun life as an aspect and the adjective excellent as the sentiment. The aspect life is completely irrelevant to the product since the reviewer was actually talking about the battery life. Now, the Compound rule checks whether the extracted aspect is the governor for a compound dependency. If such a dependency is found, the dependent of this dependency is appended with the governor, and the bigram is now considered to be an aspect as a whole. Hence, the sentiment is excellent and the aspect is battery life.

VI. Comparative analysis with existing system

Indian e-commerce website, flipkart.com provides aspect specific ratings on a certain limited set of products. We compared how our system fared with the aspect-sentiment extraction system implemented by flipkart. Bold letters in the flipkart reviews indicates sentiment extracted for the aspects.
FLIPKART

ASPECT BASED RECOMMENDATION SYSTEM
Camera:[“good”]
Display:[“good”,”crispy”]
Processor:[“powerful”]

FLIPKART

ASPECT BASED RECOMMENDATION SYSTEM
Overall:[“awesome”]
Camera:[“best”]
Display:[“super”]
FLIPKART

ASPECT BASED RECOMMENDATION SYSTEM
Processor:[“raging”,”excellent”]
Display:[“dynamic”]
Battery:[“incredible”]
Overall:[“top notch”]

We parsed 50 reviews to measure the accuracy. The accuracy is calculated by aspects extracted/aspects available in review. There were a total of 108 aspects which were relevant to our defined aspects. But our system extracted 92 aspects in all.

Therefore, the accuracy is given by

(92/108)*100 = 82.14%

The effectiveness of the system is evident by the fact that it uses an all-rounded parser like the StanfordCoreNLP and a well-defined rule set which abides by the typed dependencies of the StanfordCoreNLP parser. We used a dictionary based approach to fetch only the relevant aspects.