Procesamiento de Lenguaje Natural – PLN – 2021/22

Lab assignments: Aspect opinion extraction

Universidad Autónoma de Madrid

Escuela Politécnica Superior

Luis Castillo Cisnero & Guillermo Hoyo Bravo

List of assignment and tasks:

1. Assigment 1
    a. Task 1.1
    b. Task 1.2
2. Assigment 2
    a. Task 2.1
3. Assigment 3
    a. Task 3.1
4. Assigment 4
    a. Task 4.1
5. Assigment 5
    a. Task 5.1
    b. Task 5.2
    c. Task 5.3

Main Issues

The main issues discussed during this lab assigment were understanding apendixes C and D (and structure generated) for tasks 4 and 5. When this was achieved correctly, the next big issue were some coding decitions.

This decitions can be obseved from task 4.1 until the last one. The structures split and incom are the result from our doubts. Split is the dictionary tuple that contains all sentences with at least 1 jj and 1 nn. Incom contains the sentences with empty jj or nn value.

This decition was taken because we think that if there is no adjective to achive a polarity, it makes no sense adding this tuple. The tuple contain no valuable information at all, so it is just garbaje data that is better to move aside.

We also took the decition to add a neutral polarity = 0 to the words not contained by the list positiveWord or negativeWord. We think this is important for the final polarity avegare of the summary, obtaining a more valuable, reliable and meaningful analisys.

Last but not the least, for implementing tasks 5.2 and 5.3 we decided adding another dictionary structure counting the positive, negative and neutral sentences of a review, and an average field (Task 5.3). This structure is duplicated for the same propurse but instead of a single review, for all the reviews of a item/hotel (Task 5.2). This way we generate a better undertanding of what the opinions of each revie and of each hotel are.

Now there is a longer explanations of how each assigment and task have been develop.

Assigments

Assigment 1: Review datasets

Task 1.1 - mandatory: loading all the hotel reviews from the Yelp hotel reviews file.

At the first cell its open the file yelp_hotels.json. This file is a JSON, and it is loaded into a variable named reviews. After, the longitude or number of lines of this file is extracted, and finally the first of the array is printed. This way it is checked the correct working of the code and it allows an easier comprehension.

The result of printing this code is the next:

{'reviewerID': 'qLCpuCWCyPb4G2vN-WZz-Q',

'asin': '8ZwO9VuLDWJOXmtAdc7LXQ',

'summary': 'summary',

'reviewText': "Great hotel in Central Phoenix for a stay-cation, but not necessarily a place to stay out of town and without a car. Not much around the area, and unless you're familiar with downtown, I would rather have a guest stay in Old Town Scottsdale, etc. BUT if you do stay here, it's awesome. Great boutique rooms. Awesome pool that's happening in the summer. A GREAT rooftop patio bar, and a very very busy lobby with Gallo Blanco attached. A great place to stay, but have a car!", 'overall': 4.0}

Task 1.2 - optional: Loading line by line the reviews from the Yelp beauty/spa resorts and restaurants reviews files

The first cell opens the file yelp_beautyspas.json, and loads the reviews like in the task 1.1. In the same way, the second cell opens the files yelp_restaurants.json and loads the reviews.

Both cells have the same output format as the one shown at task 1.1

Assigment 2: Aspect vocabularies

Task 2.1 - mandatory: loading (and printing on screen) the vocabulary of the aspects_hotels.csv file, and directly using it to identify aspect references in the reviews. In particular, the aspects terms could be mapped by exact matching with nouns appearing in the reviews.

This Task/Cell contains 3 arrays: servicios, amenity & amenities. After the aspects_hotels.csv, and loops the iteration of cleaning each end of line (\n) and splitting each line by the comas it contains. The structure of the file is like this: token0, token1. So we are spliting the tokens, knowing the token0 are the amenities and the token1 the amenty. Servicios is an array that saves both token on a single array.

After, the result of the generation of the array token is printed:

amenities amenity

amenities amenities

amenities services

atmosphere atmosphere

atmosphere atmospheres

atmosphere ambiance

atmosphere ambiances

atmosphere light

atmosphere lighting

atmosphere lights

atmosphere music

bar bar

bar bars

bar bartender

bar bartenders

bathrooms bathroom

bathrooms bathrooms

Assigment 3: Opinion Lexicon

Task 3.1 - mandatory: Loading Liu's opinion lexicon composed of positive and negative words, accessible as an NLKT corpus, and exploiting it to assign the polarity values to aspect opinions in assignment 4. Instead of this lexicon, you are allowed to use others, such as SentiWordNet. See Appendix F.

First, the necesary libraries are imported, and we download the necesary pakages: 'opinion lexicon' & 'sentiwordnet'. Now we load the opinion lexicon arrays: negativeWords & positiveWords.

Printing this output shows this result:

['2-faced', '2-faces', 'abnormal', 'abolish', ...]

4783

['a+', 'abound', 'abounds', 'abundance', 'abundant', ...]

2006

The same process is applied to sentiwordnet. Printing the list resulting for the "happy feeling" words, printing the polarity for "happy.a.01"

[SentiSynset('happy.a.01'), SentiSynset('felicitous.s.02'), SentiSynset('glad.s.02'), SentiSynset('happy.s.04')]
pos 0.875 neg 0.0

Assigment 4: Opinion Lexicon

Task 4.1 - Mandatory: Once the aspect vocabulary and opinion lexicons are loaded, the opinions about aspects have to be extracted from the reviews. For this purpose, POS tagging, constituency and dependency parsing could be used.

• POS tagging would allow identifying the adjectives in the sentences.

• Constituency and dependency parsing would allow extracting the relations between nouns and adjectives and adverbs.

See Appendix C and Appendix D for code examples.

Te first two cells show the Appendix C and D, to learn how they work. Once understanded, this knowledge is used for spliting in sentences the reviews from yelp_hotels.json and seving them into an array called children. This splited sentences only contain the adjectives and nouns words: ‘jj’ & ‘nn’.

Then the structure generated is printed to comprehend it better and make the job easier. This is the result. We also print ta reduced children structure.

(JJNN Great/JJ hotel/NN)

('Great', 'JJ')

Great

JJ

('hotel', 'NN')

hotel

NN

[Tree('JJNN', [('Great', 'JJ'), ('hotel', 'NN')]), Tree('JJNN', [('Central', 'NNP'), ('Phoenix', 'NNP')]), ...

The next cell contains 3 functions: polarity_finder, amenties_finder & separator.

Polarity_finder: (entry: only one adjective)

This function search the adjective/jj in the positiveWord & negativeWord list from task 3 to obtain the adjective polarity: Positive = 1, Negative = -1, Neutral = 0.

Amenties_finder: (entry: only one noun)

This function search the noun/nn in the amenity list created at task 2. If the noun is found, the amenities list is indexed with the same index from the amenity, returning the aspect for this word.

Separator: (entry: full children array)

This function generates two arrays structures: clasifier & incomplete. Both arrays contain the dictionary structure called: newEntry{'jj': '', 'nn': '', 'pol': 0, 'amen': '',}.

Afterwards, the structure cleans the incoming sentences with a regular expresion and appends to the dictionary the jjs, their polarity, the nns and their amenties, thanks to the finder functions.

This is the printed Output:

{'jj': 'Great ', 'nn': 'hotel ', 'pol': 0, 'amen': '-'}

{'jj': 'little ', 'nn': 'vaca ', 'pol': 0, 'amen': '-'}

{'jj': 'first ', 'nn': 'time ', 'pol': 0, 'amen': '-'}

{'jj': 'favorite ', 'nn': 'spa ', 'pol': 1, 'amen': 'spa'}

{'jj': 'fourth annual ', 'nn': 'user conference ', 'pol': 0, 'amen': '-'}

{'jj': 'favorite ', 'nn': 'hotel ', 'pol': 1, 'amen': '-'}

{'jj': 'great ', 'nn': 'value ', 'pol': 1, 'amen': '-'}

{'jj': 'late springearly ', 'nn': 'summer ', 'pol': 0, 'amen': '-'}

{'jj': 'friendly ', 'nn': 'place ', 'pol': 1, 'amen': '-'}

{'jj': 'beautiful ', 'nn': 'resort ', 'pol': 1, 'amen': '-'}

{'jj': 'amazing ', 'nn': 'Mini Vacation ', 'pol': 1, 'amen': '-'}

{'jj': 'blah ', 'nn': 'decor ', 'pol': -1, 'amen': 'building'}

…

…

…

Assigment 5: Opinion summarization

Task 5.1 - Mandatory: Visualizing on screen the aspect opinions (tuples) of a given review

The process of this task is making the same procees from task 4.1, but with only onw review. So the process is basically the same except an aux is needed to extract each sentences from a review without crashing the cp.parser function.

The output ptinted is the full review and after the dictionary structure for all of it:

I stayed here last month. It's your average motel (since the doors face outside, not a hallway)… … …

{'jj': 'last ', 'nn': 'month ', 'pol': 0, 'amen': '-'}

{'jj': 'average ', 'nn': 'motel ', 'pol': 0, 'amen': '-'}

{'jj': 'next ', 'nn': 'door ', 'pol': 0, 'amen': '-'}

{'jj': 'darkcolored ', 'nn': 'Lincoln ', 'pol': 0, 'amen': '-'}

{'jj': 'actual ', 'nn': 'taxi ', 'pol': 0, 'amen': 'transportation'}

{'jj': 'main ', 'nn': 'complaint ', 'pol': 0, 'amen': '-'}

{'jj': 'entire ', 'nn': 'time ', 'pol': 0, 'amen': '-'}

{'jj': 'front ', 'nn': 'desk ', 'pol': 0, 'amen': '-'}

Task 5.2 - Mandatory: Visualizing on screen a summary of the aspect opinions of a given item. Among other issues, the total number of positive/negative opinions for each aspect of the item could be visualized

For this task the process created consist of 4 functions; process_review, review_hotel, find_all_hotels, group_hotels.

Process_review: (Entry: Full Review Text)

This function generates a new children structure processed by the function separator. To generate this new children structure we make again use of pos_tagging, cp-parse and separator function.

Review_hotel: (Entry: reviewList – with name = asin) - Encontrar todas las reviews para un asin

This function implements a comprehension list returning all the reviews from a hotel

Find_all_hotels: (Entry: None) - Encontramos todos los asins

This function finds and collect all the hotels form the reviewList (file yelp_hotels.json), saving a list, without repetition, of the asin or hotel_id token, in a set structure, returning this as a result.

Group_hotels: (Entry: None)

This function generate the necesary structures to save all the reviews by hotel and save each result of the polarity for each review and for each hotel (reviews list).

Printed result:

[{'asin': 'eBslbGvlDI92ppOi6CnTDw',

'reviews': [{'review': "Okay so lets clear one thing up.....",

'value': [{'amen': '-', 'jj': 'extra ', 'nn': 'cash ', 'pol': 0},

{'amen': 'location',

'jj': 'impeccable ',

'nn': 'location ',

'pol': 1}]}]}

Task 5.3 - optative: conducting and reporting a manual evaluation of the implemented aspect opinion approach

o Precision can be computed by checking the correctness of extracted aspect opinion tuples

o Recall can be computed by checking whether real aspect opinions were not extracted by the implemented approach

This taks required the implementation of a function called calculate_polarity_per_review which calculates the total number of positive, neutral and negative opinions, calculating, too, the average opinion review for one review. This function also need the structures generated at the previpus function in task 5.2, and generates the average review values for a total set of reviews (staked by hotel), including like previously done, the total number of positive reviews, negative reviews and neutral reviews.

Printed Result:

{'asin': 'yR6kgWuMUNG6fjOrzzhfeQ', '

reviews': [

{'avg': 0.5,

'neg': 0,

'neu': 3,

'pos': 3,

'review': 'Back in town and back to our favorite hotel in the area. Right across from North Mountain preserve, where we love to hike in the morning ..... '}],

'stats': {'avg': 1.0, 'neg': 0, 'neu': 0, 'pos': 1}}]

By computing the total polarity of each review, and comparing it to the actual review, it is observed that the values obtained are accurate. Other optional tasks would have made some review scores better. Although, the calculations made are good enough. By reading reviews manually we can observe all of them have scores acordingly to what we have understood, as it is expected, being an accurate result.