

NLP Exercise 2: Aspect-Based Sentiment Analysis

S. Ait-Mokhtar

Deadline for project deliverable: **March 15, 2022**

Send deliverable to: salah.ait-mokhtar@naverlabs.com

1 Introduction

The goal of this exercise is to implement a classifier to predict aspect-based polarities of opinions in sentences. The classifier assigns a polarity label to every triple <aspect_category, aspect_term, sentence>. The polarity labels are: *positive*, *negative* and *neutral*.

An example of a (small) dataset containing only 2 instances

negative	SERVICE#GENERAL	Wait staff	0:10	Wait staff is blantly unappreciative of your business but its the best pie on the UWS!
positive	FOOD#QUALITY	pie	74:77	Wait staff is blantly unappreciative of your business but its the best pie on the UWS!

Each line contains 5 tab-separated fields: the **polarity** of the opinion, the **aspect category** on which the opinion is expressed, a specific **target term**, the **character offsets** of the term (*start:end*), and the **sentence** in which that opinion is expressed.

For instance, in the first line the opinion about the **SERVICE#GENERAL** aspect, which is associated to the term "**wait staff**", is *negative*.

In the example of the second line, the sentence is the same but the opinion is about a different aspect and a different target term, and is *positive*.

There are 12 different aspects categories:

AMBIENCE#GENERAL
DRINKS#PRICES
DRINKS#QUALITY
DRINKS#STYLE_OPTIONS
FOOD#PRICES
FOOD#QUALITY
FOOD#STYLE_OPTIONS
LOCATION#GENERAL
RESTAURANT#GENERAL
RESTAURANT#MISCELLANEOUS
RESTAURANT#PRICES
SERVICE#GENERAL

The training set has this format (5 fields) and contains 1503 lines, i.e. 1503 opinions.

File: traindata.csv

The classifier should be learned only from this training set.

A development dataset is distributed to help you set up your classifier and assess its performance. It has the same format as the training dataset.

File: devdata.csv (376 lines, i.e. 376 opinions).

We will perform the final evaluation by measuring the accuracy of your classifier on a test dataset that is not distributed. The majority class of the dev set is about 70% (*positive* labels), and will be used as a baseline.

2 How to proceed

1. Install and use **python >= 3.8.x** (required). Besides the standard python modules, you can use the following:
 - a. PyTorch >= 1.10.0
 - b. HuggingFace transformers, **version 4.16.x**.
 - c. scikit-learn >= 0.24.0
 - d. pandas >= 1.3.0
 - e. nltk >= 3.6.0
 - f. gensim >= 4.1.2
 - g. stanza == 1.3.0
2. You can work on the project **in groups of 4 max**.
3. Download the **exercise2.zip** file and uncompress it to a dedicated root folder. The root folder will contain 3 subfolders:
 - a. **data**: contains traindata.csv and devdata.csv
 - b. **src**: contains 2 python files: tester.py, classifier.py
 - c. **resources** : (empty) where you can put your resource files if needed
4. Implement your classifier by completing the "Classifier" class template in src/classifier.py, containing the following 2 methods:
 - a. The **train** method takes a training data file as input and trains the model
 - b. The **predict** method takes a data file (e.g. devdata.csv) and should return a python list of predicted labels. The returned list contains the predicted labels in the same order as the corresponding examples in the input file
5. You can create new python files in the src subfolder, if needed to implement the classifier.
6. You can train your model using gpu (assume max gpu memory == 11GB)
7. To check and test your classifier, **cd to the src subfolder** and **run tester.py**. It should run without errors, training the model on traindata.csv and evaluating it on devdata.csv, and reporting the accuracy measure.
8. Please **do not modify tester.py**! Your program must run successfully without having to modify this file.
9. The exact content of the deliverable is described in section 3 of this document

10. Your project deliverable must be a unique **zip** file (a compressed folder). No gz, or other compression format.
11. The **name of the zip file** must consist of the family names of the authors of the deliverable.
Example: Clouseau_Holmes_Velasquez.zip
12. The zip file size should not exceed 3 MBs.
13. Send the zip file by email to: **salah.ait-mokhtar@naverlabs.com**

3 Deliverable Content

When uncompressed, the main folder must contain the following elements:

Element	Description
README.txt	A plain text file that should contain a couple of paragraphs describing: <ol style="list-style-type: none">1. Names of the students who contributed to the deliverable (max=4)2. A couple of paragraphs describing your final system (type of classification model, feature representation, resources etc.)3. The accuracy that you get on the dev dataset.
resources	(optional): containing specific resources if used (e.g. polarity lexicons)
src	A subfolder containing ALL the python source files required to train and run your classifier using the unmodified tester.py : this file is used to run and evaluate your classifier.

Note:

- **Please make sure that when you cd to the src subfolder and launch `tester.py` (unmodified!) with the python interpreter, it runs without errors:** it trains the classifier on the train set and evaluates it on the dev dataset, outputting the average accuracy.
- You can use any type of models, including non-deep models. However, note that classification methods based on pre-trained, transformer-based language models usually yield better accuracy.
- You can also use deep models enriched with explicit or implicit linguistic features: lemmatization, shape, POS tags, and/or parsing dependency relations using the stanza parser (c.f. lecture on parsing), polarity lexicons, etc.