Web Science 2021: Final Project

February 9, 2021

The project will be graded as a whole and all parts of this project are compulsory. The project description is likely to be adapted throughout the course. The project should be completed **individually**. You should submit:

- A **report** detailing the project, what you have implemented, and your results and observations;
- **Code** to run your experiments and **documentation** (`readme` file) on how to run it.

The final submission is a `.zip` file, which should contain each of the items listed above. The submission deadline is no later than **Friday 9 April 2021 at 12h00**. The format of the report should be a PDF document using the ACL template[1], no more than **6 pages** (not including references, if needed).

1 Overview of the project

The aim of the project is first to apply techniques to aggregate labels collected through crwodsourcing, second to build different sentiment classifiers, and third to construct content-based and collaborative filtering recommendation systems. The sections below provide details on each step of the project, but generally, the project is open-ended, and you will have the opportunity to explore different solutions. At the end of this course, you will present your work to the course instructors.

In all cases you should describe what you tried, what worked, what did not work, and why you think it did or did not work. For example, did you use an off-the-shelf method? How did you adapt it for the given task? Why does this adaptation work or not? Did you do some preprocessing or cleaning of the dataset? Was is it useful? Why or why not?

Moreover, you need to describe the limitations of what you did. What could have lead to improvements in model performance? How could you have approached sentiment analysis and automatic recommendation differently? What was particularly challenging about working with these datasets and why do you think that was? This discussion does not require further experiments, but requires you to examine your experimental results, critically think about your choices and the assumption you made, make hypothesis on how you can overcome some limitations and improve your solution. Furthermore, your discussion should be based on evidence, e.g. lecture material, relevant literature.

[1] `https://2021.aclweb.org/downloads/acl-ijcnlp2021-templates.zip`

2 Week 6 - Familiarize with the Datasets

The purpose of the first week is to familiarize yourself with the programming basics needed for the project. Two datasets will be given, one for sentiment analysis (Section 2.1) and one for recommender systems (Section 2.2). They can also be found on Absalon. You will need to do statistical analysis to attain insights into both datasets.

2.1 Sentiment Analysis - SST Dataset

You will explore Stanford Sentiment Treebank Dataset (SST)[2]. You can find the dataset on Absalon (in `Files -> dataset -> sst2`), otherwise the direct download link is:

`https://absalon.ku.dk/courses/47158/files/folder/dataset/sst2`

The dataset consists of 10 605 pieces of processed snippets from Rotten Tomatoes. Each snippet contains either one or multiple sentences. Each sentence was parsed into multiple phrases and annotated with a label of positive (1) or negative (0) sentiment. In the `sst2` folder, you will find the following files: `stsa.binary.phrases.train`, `stsa.binary.dev`, and `stsa.binary.test`, which correspond to the training, validation and test sets respectively. The files are in `CSV` format and should be opened with a suitable reader library (e.g., pandas for Python[3]). Each line in the files is formatted as follows:

`index,label,text`

where `index` is the phrase ID, `label` is the sentiment label with values 0 or 1, denoting negative and positive sentiments respectively, and `text` is the actual text content of the phrase.

During this first week you need to:

- Download and import the data-set.
- Analyze the data to find out:
  - Is the dataset balanced (Does it have same number of positive or negative labels)?
  - What are the most common n-grams (for $n = 1, 2, 3$).
  - What are the most common n-grams (for $n = 1, 2, 3$) for each class label?
- You can analyze the dataset to find other interesting insights: for example: is there a correlation between the length of the text and its label?

[2] `https://nlp.stanford.edu/sentiment/`
[3] `https://pandas.pydata.org/`

2.2 Recommender System - Amazon Review Data

You will use Amazon Review Data (2018). You can download the dataset and find information about it here:

`https://nijianmo.github.io/amazon/index.html`

The complete review data contain millions of entries, which can be computationally expensive to process. We will use the "Small" (5-core) subsets, specifically the "Software" category. The citation of the dataset and other dataset details can be found on the linked website page.

The dataset is in `JSON` format, you thus need read-in tools. You can use the built-in Google Colab notebook to pre-process the dataset or subsets through this link: `https://colab.research.google.com/drive/1Zv6MARGQcrBbLHyjPVVMZVnRWsRnVMpV`. The notebook uses gzip and pandas for data wrangling. You are free to utilize other tools though.

During this first week you need to:

- Download and import the dataset (you can also try other subsets from "Small subsets for experimentation") and compute the product and user statistics (such as mean/standard deviation of rating scores).
- Explore the metadata of the downloaded dataset and write a discussion; what metadata might be related/useful if you are asked to recommend products to a user?
- Construct a user-product matrix of the datasets you downloaded, you will reuse this matrix to build two recommender systems in later weeks.

3 Week 7-8 - Crowdsourcing Exercise

The purpose of the second and third week is to (1) label phrases from the SST dataset; (2) aggregate all the labels applying majority vote or other approaches to determine the final label; (3) compare the final labels with the labels included in the SST dataset. This is explained next.

3.1 Relabel the SST Dataset with Crowd-sourcing

You will use the SST dataset presented in Section 2.1. You will be given 100 texts with their identification numbers. You will label each of them and meanwhile the same comments will be sent to multiple students. You will generate a spreadsheet with all the comments and labels (you will find a template on Absalon). For each of these comments, you must do the following:

1. Label the texts with a number between 1 and 5, where 1 ="very negative", 2 ="negative", 3 ="neutral", 4 ="positive", 5 ="very positive".
2. Identify the words or phrases that contributed towards your label. For example, if the text is "a great performance and a reminder of dickens' grandeur", the label will probably be "very positive", and one of the contributory phrases can be "a great performance".

3. **Your re-labelled texts should be turned in through Absalon by Monday, 22 February, 12h00.** Then you will receive a global spreedsheet which contains all the collected labels.

4. You need to use the global spreadsheet to compute relevant dataset statistics and inter-annotator agreement for the collected labels. You can use Krippendorff's alpha[4] or any other coefficient.

5. You need to use the global spreadsheet to determine the final label. To aggregate the labels you can use majority vote or any other aggregation approach. List any details of how you resolved ambiguity among different crowd workers when no label received a majority vote.

6. You need to convert the 5-scale labels to 2-scale labels. For example, you can convert the labels 1 and 2 to 0 (negative) and $3, 4, 5$ to 1 (positive). Note that you can convert the labels from the 5-point scale to the 2-point scale before or after using majority vote to aggregate them. Is there a difference between these 2 approaches? What about the inter-assessor agreement, is it the same with 2-scale or 5-scale labels?

7. Compare these final labels with those included in the original SST dataset. You can consider the SST labels as the gold-standard (correct) labels and compute accuracy or other relevant metrics.

4 Week 9-10 - Sentiment Analysis

In these two weeks, you will reuse the SST dataset presented in Section 2.1. The goal is to build a sentiment classification model for text data. You will represent the texts in two ways:

1. The texts should be converted to a term document (a term is a word in the phrase and the text is the document) matrix with tf-idf values.

2. You will use continuous representations for terms which will come from DNN word embedding models (Glove or Word2Vec).

For computationally expensive deep learning training, you can run your code on Google Colab. By the end of this session, you should be able to select, implement, evaluate the performance of appropriate classifiers for a certain sentiment analysis task. Specifically, the tasks are:

1. Clean the text data, for example by removing punctuation, stemming, lemmatization.

2. Tokenize the text to create a term-document matrix using tf-idf tokenizers[5].

3. Use unsupervised dimension reduction techniques (PCA, random projection) on the tokenized data[6].

[4] `https://pypi.org/project/krippendorff/`

[5] `https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html`

[6] `https://scikit-learn.org/stable/modules/unsupervised_reduction.html`

4. Use a logistic regression classifier and another linear classifier of your choice to classify the data. Does data cleaning help in improving accuracy? Does dimensionality reduction help?

5. **Extra credit**: Which features (in your case, words) are found *important* by the logistic regression classifier? Do they match with the words you annotated in section 3.1?

6. Use a pre-trained word embedding model (Glove/Word2Vec or something of your choice) to generate continuous representation for the words. You can use a library such as Gensim[7] for this. Convert the whole text to a continuous vector using aggregation methods (average or max pooling). Use the same classifiers to build a model. Compare the performance with the other representation.

7. **Extra credit**: Use a word convolutional network for the classification.

8. Discuss the performance of various classifiers, for example why a classifier outperform the rest or why it can not.

5 Week 11-12 - Recommender Systems

The purpose of weeks 11 and 12 is to get some experience in developing a recommender system. You will reuse Amazon Review datsets described in Section 2.1 with the training/validation/test split 80% for training, 10% for validation, and 10% for testing (available on Absalon `Files -> dataset -> amazon_review` from Week 11).

Collaborative filtering System:

- Define a function to calculate the similarity score for either users or product for all pairs of instances (a similarity metric based on cosine similarity for instance).
- Predict ratings using the similarity matrix and training data.
- Report prediction mean squared error.

Content-based System:

- Factorize the "comments" column of each product (representing sentences with a numerical value), select other factors that can be used as product or user features.
- Define a function to calculate the correlation of a target product and other products.
- Sort products according to the correlation scores (products with few rating might need to be removed) to collect predicted labels (from correlation scores to ratings).
- Report prediction mean squared error.

[7] `https://radimrehurek.com/gensim/models/word2vec.html`

Error Analysis:

- Compare the two types of recommender systems, which one works best? Why? What are the advantages and limitations of each approach?
- Print user-product pairs that are wrong and look for patterns.
- For users or products whom/whose your model performs poorly on, discuss the potential reasons behind.

The above list is not an exhaustive list, meaning that you can exploit other approaches, use other evaluation measures, and explore different tools for error analysis.

6 Academic Code of Conduct

You are welcome to discuss the project with other students, but sharing of code is not permitted. Copying code directly from other students will be treated as plagiarism. Please refer to the University's plagiarism regulations if in doubt. For questions regarding the project, please ask on the Absalon discussion forum.