HOMEWORK 2

SENTENCE CLASSIFICATION FROM SCRATCH[1]

CS 678 ADVANCED NATURAL LANGUAGE PROCESSING (FALL 2023)
https://nlp.cs.gmu.edu/course/cs678-fall23/

OUT: August 22, 2023
DATA EXCHANGE: September 29, 2023
MODEL AND REPORT DUE: October 13, 2023

Based on assignments created by Mohit Iyyer and Graham Neubig.

Your name: _____

Your GID: _____

IMPORTANT SUBMISSION GUIDELINES: (1) The homework is accompanied by a python notebook. Please upload your completed notebook, along with any other code and data files, in one single zipped folder to Blackboard. (2) The report PDF should be submitted to Gradescope. **If you fail to do this, you will receive no credit for this homework!**

Graded Questions: 100 points
Bonus Points: 15 points/student
Total Points Available: 115/100

Additional Notes:

- Upload the whole report PDF (including all pages, also if you're scanning it).
- Some questions require writing Python code and computing results, and the rest of them have written answers. For coding problems, you will have to fill out all code blocks that say `YOUR CODE HERE`.
- For text-based answers, you should replace the text that says "Write your answer here..." with your actual answer.

[1]Compiled on Thursday 7th September, 2023 at 16:22

1 Introduction

So far in your machine learning classes, you may have experimented with standardized tasks and datasets that were provided and easily accessible. However, in the real world, NLP practitioners often have to solve a problem from scratch, which includes gathering and cleaning data, annotating the data, choosing a model, iterating on the model, and possibly going back to change the data. For this assignment, you'll get to experience this full process.

Please note that you'll be building your own system end-to-end for this project. While we provide starter code (see accompanying notebook), you're welcome to try anything beyond what we provide. You will have to collect your own data and train a model of your choice on the data. We will be releasing an unlabeled test dataset a few days before the assignment deadline, and you will run your already-constructed system over this data and submit the results. We also ask you to follow several experimental best practices, and describe the result in your report.

The full process will require the following steps:

1. Understand the task specification
2. Collect raw data
3. Annotate test and training data for development
4. Train and test models using this data
5. "Deploy" your System
6. Write your report (using this template)

Each of these are outlined in the sections below.

NOTE: This assignment will require time for each step, so make sure to plan ahead!

2 Task Specification

For this assignment, the scenario is that you want to help a social scientist friend, who is studying the *framing* of social issues in media and other documents.

What is framing? Framing refers to the selection of some aspects of a perceived reality and make them more prominent in a communicating text such as a news article and a Tweet. Examples[2]:

- When some news media emphasize the mental illness of gun shooters over other aspects of gun violence in covering the issue, this is framing.
- When you choose to purchase a yogurt product that is advertised "90 percent less fat" rather than the one saying "10 percent fat," this is when the framing effect occurs.

Why does it matter? Quoting from one of the seminal studies of the problem Liu et al. (2019):

> In a polarized media environment, partisan media outlets intentionally frame news stories in a way to advance certain political agendas. Even when journalists make their best efforts to pursue objectivity, media framing often favors one side over another in political disputes, thus always resulting in some degree of bias. Hence, a news framing analysis is helpful because it not only tells us whether a news article is left- or right-leaning (or positive or negative), but also reveals how the article is structured to promote a certain side of the political spectrum.

[2]Taken from http://www.openframing.org/home.html

> In communication research, manual identification of media frames is a challenging task due to the large amount of media data in this news-saturated environment. More importantly, there is a high level of complexity in framing analysis that often requires a careful investigation of nuances in news coverage, which is time-consuming.

Hence, an NLP solution that automates media framing identification would immensely help social scientists and other analysts.

The Task Specifically, we will ask you to identify the framing of a given parapgraph/sentence in several languages. Effectively, this is a straight-forward text classification problem in a multilingual setting.

Input: The input to the model will be a text file with one paragraph/sentence per line. The text will already be tokenized using the spacy tokenizer,[3] and you should not change the tokenization. An example of the input looks like this (notice the spacing of the punctuation):

```
Some economists say that immigrants , legal and illegal , produce a net
economic gain , while others say that they create a net loss .
```

Output: The output of your model should be a .tsv file, with one sentence per line, a tab, and then the corresponding label.

Please refer to the provided example data files for a specific example.

The Annotation Standard Social scientists have created a widely accepted list of 15 cross-cutting framing dimensions, such as economics, morality, and politics. These were originally developed by Boydstun et al. (2014) and are termed the "Policy Frames Codebook". The paper describing the codebook is publicly available and the codebook is described in depth in Section 3.2.[4] You should read carefully and make sure you understand the scheme. The framing dimensions are also listed in the Table to the right.

Economic	costs, benefits, or other financial implications
Capacity and Resources	availability of physical, human or financial resources, and capacity of current systems
Morality	religious or ethical implications
Fairness and Equality	balance or distribution of rights, responsibilities, and resources
Legality, Constitutionality, Jurisdiction	rights, freedoms, and authority of individuals, corporations, and government
Policy Prescription and Evaluation	discussion of specific policies aimed at addressing problems
Crime and Punishment	effectiveness and implications of laws and their enforcement
Security and Defense	threats to welfare of the individual, community, or nation
Health and Safety	health care, sanitation, public safety
Quality of Life	threats and opportunities for the individual's wealth, happiness, and well-being
Cultural Identity	traditions, customs, or values of a social group in relation to a policy issue
Public Sentiment	attitudes and opinions of the general public, including polling and demographics
Political	considerations related to politics and politicians, including lobbying, elections, and attempts to sway voters
External Regulation and Reputation	international reputation or foreign policy of the U.S.
Other	any coherent group of frames not covered by the above categories

Domains Your social scientist friend is planning on applying your system on news data related to two social issues:

- `immigration`
- `same-sex marriage`

Languages The seed data are all in English. However, your social scientist friend cares a lot about multilinguality, and ideally your system should be able to provide labels for sentences in the following languages: English (eng), Mandarin Chinese (cmn), Hindi (hin), Telugu (tel), Nepali (nep), Bengali (ben), Greek (ell), German (deu), Swahili (swa), as well as two (or three) surprise languages that will be revealed later (see Timeline).

[3] https://spacy.io/api/tokenizer/

[4] https://kilthub.cmu.edu/ndownloader/files/11903384

3 Data Collection [35 pts]

Seed Data To help you get started, we provide 250 sentences in English, annotated with their frames. All sentences are on the `immigration` domain. You may study them to understand the task and the annotation schema, and you may use them however you wish (e.g. as training or dev data or as a starting point for data augmentation).

First Round of Data Labeling In this first round, you will collect/create a labeled dataset for our task of **at least 150 sentences**. Notice that this is a *minimum* number of data you need to annotate, but annotating more is allowed.[5]

The purpose of this part is to familiarize everyone with the annotation process and the codebook. Also, as any good data scientist would tell you, a little bit of experience performing the task you are trying to automate is invaluable!

It is up to you to find (or create) sentences/paragraphs relevant to the task. We provide some ideas, pointers, and considerations about this part below.

- You could create a list of keywords related to the two domains, search the web for relevant articles, and scrape sentences from there.
- You could perhaps write some your own sentences (but be careful of your own biases!)
- You could perhaps use a writing assistant or a chatbot (e.g. ChatGPT) and create sentences relevant to a domain with a certain framing.
- There are two target domains and 15-ish output classes. How will you make sure that all classes and domains are represented in the data you will annotate? Remember that a neural model will only be able to predict something that it has been trained on!
- Can you use the seed data in any way?
- In which language(s) will you search for, create, or annotate data?
- What would you do if more than one labels seem appropriate for a sentence?
- Be creative!

Package your annotations in a file named `first_data.tsv`. The .tsv file should include the following five columns:

- `text`: the sentence that will be the input to the model
- `language`: the language of the sentence. If codemixed, use a comma-separated list of languages. Use either language names or language ISO-3 codes.
- `label`: the framing label of the sentence. Use either the label name (e.g. "economic") or the label ID according to the annotation schema.
- `source`: the source of the sentence. If scraped, provide the URL. If written by you or someone else, provide the author (or system) name. If created in another way, provide an explanation in your report.
- `annotator`: this should be your GMUID

1. (10 points) Describe the source of your unlabeled data, why you chose it, and what kind of sentence selection process you used (if any) to choose the (minimum) 150 sentences for annotation. If you chose to annotate more data, please justify your choices on the source and amount of data you create. **To receive the question credits, you must provide the above-described file with your data.**

[5] A good model will likely need more data than this, but be careful to not waste too much of your time annotating data though!

Data Source Description

Second Round of Data Labeling In this second round, you will collect annotations on another 150 sentences from at least two classmates. **Everyone in this class will need to both create their own dataset and also serve as an annotator for (at least) two other classmates. In total, each student will annotate a minimum of 450 sentences (150 in the first round, and then 150+150 in the second round). The data exchange between annotators should happen at the latest by Sept 29.**[6]

The idea now is that you will select another batch of 150 sentences to be annotated, but this time each sentence should be annotated by at least 2 annotators (that are not you).

In particular, you should:

1. Create another batch of 150 unlabeled sentences (minimum).
2. Find two classmates willing to label 150 sentences each (use the Piazza "search for teammates" thread if you're having issues finding labelers).[7]
3. Send the data to your annotators in a way that they can easily annotate the data (e.g., a spreadsheet, Google form, or by using Labelstudio).
4. Collect the annotations from your annotators and sanity check the data for basic cleanliness (are all examples annotated? are all labels allowable ones?)

Notes:

1. How will you select the data to share for annotation? What language(s) will they be in?
2. Plan ahead on your commitments/arrangements for data annotation!

2. (7 points) Write down the names and emails of the two classmates[8] who will be annotating your data below. Once they are finished annotating, create two .tsv files (annotator1.tsv and annotator2.tsv) that

[6] This is a **minimum**. You are allowed to annotate more than 300 sentences in this second round, but be mindful of your time.
[7] Or three classmates willing to label 100 sentences each. Or 5 classmates willing to label 60 sentences each. You get it.
[8] Feel free to modify if using more than two annotators.

contain each annotator's labels. Each file should have three columns with headers text, label, and GMUID respectively. You will include these files in your submission when you're finished with this homework.

Annotator 1	Annotator 2
Name: name here Email: email here GMUID: GMUID here	Name: name here Email: email here GMUID: GMUID here

Please contact the course staff with any issues regarding the data exchange. Email us if e.g. you committed to provide data for a classmate but cannot do so, or if a classmate committed to provide data for you but has not done so. We will try to resolve this in the best way possible. But the success of this homework relies on everyone behaving professionally: set and meet deadlines!

3. (8 points) Now, compute the inter-annotator agreement between your annotators. In the provided code cell in the accompanying notebook, read the data from the two files and compute both the raw agreement (% of examples for which both annotators agreed on the label) and the Cohen's Kappa.[9] Feel free to use implementations in existing libraries.[10] After you're done, report the raw agreement and Cohen's κ scores below:

Raw Agreement	Cohen's Kappa
Your answer here	Your answer here

If you're curious, Cohen suggested the Kappa result be interpreted as follows: values ≤ 0 as indicating no agreement and 0.01–0.20 as none to slight, 0.21–0.40 as fair, 0.41– 0.60 as moderate, 0.61–0.80 as substantial, and 0.81–1.00 as almost perfect agreement.

4. (5 points) To produce your final dataset, you will need to **aggregate** the annotations from your annotators. That means that for cases where they disagree, you will need to choose a single label. Use any method you like *other than random label selection* to perform this aggregation (e.g., have the two annotators discuss each disagreement and come to consensus, use Piazza to create a poll on a label, or choose the label you agree with the most). You will need to create a file `aggregated_data.tsv`, with the same format as your `first_data.tsv`.

   Below, describe your aggregation strategy and why you chose it. **To receive the question credits, you must provide the above-described file with your data.**

[9] https://en.wikipedia.org/wiki/Cohen%27s_kappa

[10] e.g., sklearn: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.cohen_kappa_score.html

Aggregation Strategy

5. (5 points) Based on what you learned about the task and your data, you might want to update your labels for the `first_data.tsv` file (remember, no one has "checked" this, you were the only one who provided annotations for this portion of the data). If you do so, provide the updated file as `updated_data.tsv` with the same format.

   Last, combine the files from the two rounds of annotations into a single file, named `final_data.tsv` with the same format as the others. **To receive the question credits, you must provide this file with your data.**

4 Model Training and Testing [25 pts]

The next step is to train a text classification model and try to get an estimation of how well it would perform, based on your data. We provide code to do this in the accompanying notebook. You will need to upload the data to the Colab session (if you use Colab, as recommended, just click the folder icon in the sidebar to the left of the screen).

Follow the steps in the notebook to connect to a GPU, install the Huggingface transformers library, read-in the data, create train/dev/test splits, set hyperparameters (feel free to play with these!), and then fine-tune your model.

6. (5 points) After you're satisfied with your hyperparameters (i.e., you're unable to achieve higher validation accuracy by modifying them further), it's time to evaluate your model on the test set! Run the corresponding cell in the accompanying notebook and report below the validation and test set accuracy you obtain.

Dev Acc	Test Acc

7. (10 points) Congratulations! You've now gone through the entire fine-tuning process and created a model for your downstream task. Now, describe your hyperparameter selection process in words. If you based your process on any research papers or websites, please reference them. Why do you think the hyperparameters you ended up choosing worked better than others? Also, is there a significant discrepancy between your test and validation accuracy? Why do you think this is the case?

Modeling Decisions

Provide your answer to the above questions as concisely as possible.

8. (10 points) Next, perform an *error analysis* on your model. This is good practice for your final project. Write some code in the provided code cell to print out the text of up to five test set examples that your model gets **wrong**. If your model gets more than five test examples wrong, randomly choose five of them to analyze. If your model gets fewer than five examples wrong, please design five test examples that fool your model (i.e., *adversarial examples*). Then, provide a qualitative analysis of these examples. See if you can figure out any reasons for errors that you observe, or if you have any informed guesses (e.g., common linguistic properties of these particular examples). Does this analysis suggest any possible future steps to improve your classifier?

Error Analysis Discussion

Provide your answer to the above questions as concisely as possible.

5 Data Augmentation [20 pts]

So far we have worked with whatever limited data you and your classmates were able to annotate in a small amount of time. But remember that in the end we want to deploy the model on a hidden test set that will include examples from 11 languages.

One approach is to use whatever data you have already created and perform *data augmentation*. There are several approaches for this, and we invite you to get creative with this! To help you get started, we provide some ideas below, but don't feel obliged to use any or all of them – you do need to do *something* though.[11]

Some ideas to get you started:

- Use automatic **translation** to translate your data in another language. For example, if you have annotated data in English, then you can translate the sentences into e.g. French. You can then use the same labels for the French sentences as the ones the corresponding English sentences had. Tada – now you have annotated data in French! We provide a helper file (translate.py) to illustrate how to query Google Translate and obtain translations, but feel free to use other translation tools.
  Food for thought: Which languages would you translate into? The target test languages? Others? What if training on some e.g. Spanish data would also help with some of our target languages? But what is training on some e.g. Spanish data interferes with learning and you end up performing worse on some target languages? In case you're curious, both of these are possible! The first is called "cross-lingual transfer learning", and the second is called "cross-lingual negative interference".
- Use **paraphrasing**. You can paraphrase the sentences that you currently have annotated – presumably the framing (and hence the label) shouldn't change! There's a multitude of models and approaches for doing this:
  - A simple example for English paraphrasing with the `Pegasus` model is here.[12]
  - Another way of paraphrasing is through round-trip translation. Let's say you have an English sentence. You can translate it into French, and then translate that French sentence back into English. It is very likely that you will now have a paraphrase of the original English sentence! The choice of the pivot language (e.g. French or Chinese or somethine else) could also lead to a different paraphrase.
- **Label propagation**. Instead of generating paraphrases, you might be able to find data that are similar or comparable to the ones you have already annotated. This would allow you to *propagate* the labels from your annotated examples to the unannotated ones. Or, you could make some assumptions about sentences with some careful data selection that will allow for label propagation. For example, imagine you have found long news articles with several sentences on the immigration topics, but you have only annotated a sample of these sentences – could you perhaps propagate the annotations to the rest of a paragraph's sentences or to all sentences in the article? Or maybe you could only annotate the article's title and propagate the label to all sentences in the article. Or you could try something else!
- The above are just some of the ideas that the instructors could come up with – get creative!

Compile all your augmented data in a file named `augmented_data.tsv` with a format similar to the other datasets. Please provide an indication of the augmentation process used for each example in the `source` column (e.g. you may use "eng→swa" to indication translation, "eng→swa→eng" to indicate round-trip translation, "paraphrasing-[model]" for paraphrasing, etc).

[11] We suspect that some combination of these ideas will work well.

[12] https://www.thepythoncode.com/article/paraphrase-text-using-transformers-in-python]

Load your augmented dataset to the notebook, and now train and evaluate your model. Does your model performance improve on your test set? (make sure to keep the same train/dev/test set on your original data – **you should only augment your train (and perhaps dev) data, not the test data!**)

9. (20 points) Describe your data augmentation approach. Did it result in improvements on your test set? Why do you think this happened?

Data Augmentation

Provide your answer to the above questions as concisely as possible.

6 Model Deployment [18 pts]

The final "deployment" of your model will consist of running your (best) model over a private test set (text only) and submitting your results to us. You should try to finish building your systems before this set is released, and basically not rely on these data for model training or testing. The test set will be released shortly (3-4 days) before the final submission deadline.

Several Bonus points are available (up to 15 per person) based on your models' performance on this hidden test set. See "Grading" section.

7 Timeline and Effort [2 pts]

Also, please complete below your effort for this assignment, taking all stages into account. Report the approximate time spent total, in hours (integer).

Suggested Timeline You have a total of 4 weeks for this assignment (although nothing stops you from starting this assignment early). We suggest that you keep track of your process and fill in the required parts of the report as you go. **Do NOT leave this for the last minute!** Our suggested timeline:

- week 1: understand the annotation schema, study the seed data, find data and perform the first round of annotation
- week 2: prepare data for second round of annotation, perform second round of annotation (for other students), try out existing code
- week 3: collect data, resolve conflicts, run first iteration of models, explore data augmentation
- week 4: [test set published, surprise languages revealed] explore model hyperparameters, deploy final model (aka run on test set), finalize report.

8 Grading

Beyond the 100 total credits available for completing all above steps in the assignment, we will assign bonus points according to the following guide. Remember than 15 extra points in this assignment correspond to 3 credits in your overall class grade, so this could be the difference between e.g. an A- and an A!

Bonus points will be applied greedily. The maximum bonus for each student is 15 credits.

Bonus points scheme:

1. +8 for the best performing system over the blind test set, averaged across all languages, +6 for the second best, +4 for the third best
2. +7 for the most *equitable system* across all languages, as measured by the standard deviation across the averages across languages (lower std is better). The system has to be in the top 50% of all submissions to be eligible. +5 for the second most equitable system, +3 for third best.
3. +5 for the best performing system on the surprise test languages, if it's different than the top system in criterion 1 above. +4 for the second best performing system in the surprise test languages, +2 for the third best.
4. +4 for the best performing system in each non-surprise test language, if it's different than the top system in criterion 1 above. +2 for the second best performing system in each test language.
5. +2 for each system that is Pareto optimal, in terms of effort reported and downstream performance

6. +5 for the submission that provides the most hand-annotated data, +3 for second most, +2 for third most
7. +4 for the submission that provides hand-annotated data in most languages, +2 for second most (to be shared with up to 3 submissions. If more, this bonus does not apply)
8. +4 for the submission that provides the largest final augmented dataset, +3 for second largest, +2 for third

References

Amber E Boydstun, Dallas Card, Justin Gross, Paul Resnick, and Noah A Smith. 2014. Tracking the development of media frames within and across policy issues.

Siyi Liu, Lei Guo, Kate Mays, Margrit Betke, and Derry Tanti Wijaya. 2019. Detecting frames in news headlines and its application to analyzing news framing trends surrounding U.S. gun violence. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 504–514, Hong Kong, China. Association for Computational Linguistics.

9 Collaboration Questions

Please answer the following:

1. Did you receive any help whatsoever from anyone in solving this assignment (beyond data annotation)?
   Yes / No.
   - If you answered 'yes', give full details: _____
   - (e.g. "Jane Doe explained to me what is asked in Question 3.4")
2. Did you give any help whatsoever to anyone in solving this assignment (beyond data annotation)?
   Yes / No.
   - If you answered 'yes', give full details: _____
   - (e.g. "I pointed Joe Smith to section 2.3 since he didn't know how to proceed with Question 2")
3. Did you find or come across code that implements any part of this assignment?
   Yes / No. (See policy on "found code")
   - If you answered 'yes', give full details: _____
   - (book & page, URL & location within the page, etc.).