Homework 2

Annotations-only submission due Oct 5 at 11:59PM
Final submission due Oct 8 at 11:59PM
This version: uploaded Oct 1

CS 490A, UMass Amherst, Fall 2021

This assignment consists of an annotation experiment where, given a set of sentences, your task is to design an annotation experiment, collect annotations, and analyze and model the results. **Important: You will need to submit your peer-annotated data as a CSV to Gradescope, THREE DAYS PRIOR TO THE FINAL DEADLINE, i.e. Oct 5 at 11:59PM.** For the final homework submission, you'll submit your PDF report and code also to Gradescope. See the "deliverables" paragraph in the Annotation Overview section below.

Overview: Annotation and Modeling Experiment

You will be given a sample of 250 tweets, and then conduct an annotation task, from start to finish:

1. Task design: Come up with a sentence-level classification NLP task for your data. Some examples include, but are not limited to: affect intensity, tense classification, topic classification, etc. Write up annotator guidelines/instructions.
2. On your personal dataset, collect annotations from two classmates for your task. In order to get everything done in time, we strongly suggest the following internal deadlines:
   - Sept 30: Researcher has found their annotators by now. (For example, via Piazza.)
   - Oct 1: Researcher sends annotation guidelines and spreadsheets to annotators.
   - Oct 3: Annotators hand off their completed spreadsheets back to the researcher.
   - Oct 5 (**annotations due date**): Researcher double checks basic data cleanliness (are all examples annotated? are all labels allowable ones?) and submits to Gradescope. This is described in section 1.2.
3. Collect feedback from annotators about the task including annotation time and obstacles encountered.
4. Calculate inter-annotator agreement and other related statistics.
5. Aggregate results to create the final dataset.
6. Perform NLP experiments on your new dataset! (Problem 2.)

Many more details on these steps are spelled out in the next section for Problem 1.

Roles: Every student in the class will take on two roles: **researcher** and **annotator**. As a **researcher**, you perform the above steps on your data. At the same time, you will also serve as an **annotator** for your fellow students.

Tasks: Every researcher will have a different set of data, and will define their *own* unique task. Therefore the researcher will have to draw up guidelines and explanations to allow their annotators to do a good job at reliably identifying whatever categories the researcher has a goal of collecting.

Annotator etiquette: Everyone must collect annotations from at least two people. Therefore everyone should be ready to annotate for two other people. If someone asks you to annotate for them, please accept, unless you have accepted more than two annotation jobs already! You should try to do a good job doing annotations for your fellow students.

It's OK if this is hard: Designing an annotation task, and collecting annotations, is tricky. This assignment is small-scale as far as annotation projects go, but hopefully it illustrates the challenges that underlie it.

Datasets. We have collected samples of publicly available English sentences from tweets for your experiments. These are sampled from August 2021 from publicly available tweets. (They may include offensive and inappropriate content.) They are available here; you must be signed in with your UMass account to access:
https://drive.google.com/drive/folders/1YTRApzteKYQx3iyQWBMaVTNtxshQP4Ka?usp=sharing

- Personal: A dataset of sentences we gathered just for you. Every student, in their researcher role, has a *different* set of 250 sentences. You can look at your own as much as you like. (You should download the file named "{YOUR_NETID}.json".)
- Note: The format is json. The text is encoded as UTF-8. In Python 3, you should be able to read the data correctly using the json package.
- Net1-10: We provide 10 extra datasets of 250 sentences for your reference, you can just look at or use in examples for your annotation guidelines.

1 Annotations (60pts)

1.1 Task design (20pts)

Conceptualize and design a classification task for your data. Take a look at your Personal dataset. What's an interesting, but also reasonable, way to annotate the classification problem for your data? It can be any binary or multiclass classification problem. Here are a few examples:

- Does the message contain humor? Sarcasm?
- Does the message contain a location?
- Does the message mention an entity like person/organization name?
- Does the message contain abbreviations?

- Is the message a passive sentence?
- Is the message about a certain specific topic: politics, movies, commercial spam, ...
- Which emotion does this message primarily express, if any?
- Does the message contain empty categories? (Please make sure you know what empty categories are before you choose this one.)

A few guidelines:

- You must have a fixed set of categories. Please use no more than 5 categories. Fewer tends to be easier to annotate.
- Do not have any categories that are very rare—say, less than 5% incidence. Annotators tend to find rare classes difficult to annotate.
- For many category systems, it's often useful to have a catch-all category like "Not Applicable" or "Other" or "Unknown." Of course, those three category labels can mean different things in different situations, or you might even need more than one of them. (For sentiment, "neutral" is a kind of catch-all.)
- Don't do a really simple problem with an obvious shallow textual indicator, like "Is the sentence a question?" since the question mark '?:' indicates this. Or something like "Does this sentence contain more than 10 words?" This violates the spirit that we want you to do a real NLP problem. However, seemingly simple tasks might be more subtle than they first appear. For example, "Does this sentence contain an emoticon?" can sometimes be hard, if there are creative or complex ASCII-art emoticons (e.g. horizontal emoticons).

Write **annotation guidelines** for your task. These are instructions you will give your annotators, that they will read before doing the annotations. It should be a written document, perhaps a half page long (or more if you think necessary). The guidelines should include:

- A list of the categories under consideration, including the exact string you want them to use when typing into a spreadsheet.
- Descriptions of the categories and what they mean.
- Example sentences that are illustrative of the categories. **Do not use examples copied directly from your Personal dataset.** We specify this so that you can't make the task too easy. Please make up synthetic examples, or use examples drawn from the extra Net1-10 datasets.
- A discussion of tricky corner cases, and criteria to help the annotator decide them. If you look at the data and think about how an annotator could do the task, you will realize a bunch of such issues!

You are in charge of defining your task! This process of boiling it down to something specific, actionable, and thus measurable, called *operationalization*. Everyone will be defining a different task, so feel free to make yours specific or unique in some way. But you will want to make it clear and as straightforward as possible for your annotators to do the task.

Deliverable: Include a copy of your annotation guidelines in your writeup.

1.2 Annotation collection (10pts)

It's time to collect annotations! Find two "volunteers" from class (or even friends not from the class) to be your annotators. If you don't know anyone, just post on Piazza to get assigned. Everyone is required to annotate for someone else if asked, to make this easy! :)

Do not communicate with your annotators about your task too much beforehand. For this homework, we want you to communicate about the task primarily through the annotation guidelines document.

Please collect your annotations through a Google Sheets document with three columns:

1. Text
2. Label from annotator
3. Notes from annotator

You should create two identical spreadsheets with just the text column, then send each, along with the guidelines, to each annotator. **Annotators should annotate independently of you, and of each other.** (If you like, you could use other software for this. A web form interface is best, but it may be too much trouble to set up. We recommend Google Sheets because it usually works well enough.)

EARLY Deliverable: In your Oct 5 Gradescope submission, include two CSV files, one for each annotator. Also include them with your final submission (they should be the same).

1.3 Annotator Feedback (10pts)

Ask your annotators for general feedback, and how long it took them. Ideally, try to interview them in a meeting or call; but written feedback might be ok too. You may be able to get some of this from the spreadsheet—Google Sheets tracks edit times to a certain extent, and you may be able to summarize some of the general issues encountered by your annotators if they left information in the 'Notes' column.

Deliverable: In your writeup, report the times and summarize the important issues in your annotation task. Did the annotators find it easy or hard? What were the biggest issues? Did the two annotators have similar or different experiences? What would you do differently if you were to revise your task?

If this were a real annotation project, what you just did would be considered the first pilot experiment, and you would work more with your annotators and iteratively refine your guidelines. But for this homework, one round is enough!

1.4 Inter-annotator agreement (10pts)

Now you'll conduct quantitative analysis of the agreement between annotators.

Deliverable: In your writeup, report:

1. The confusion matrix between the two annotators. Rows = category choices for Annotator1, Columns = category choices by Annotator2, Cells = the number of sentences with that pair of labels from the two annotators. The categories should be in the same order for both rows and columns, so the diagonal cells correspond to cases with agreement.

2. The observed agreement rate, the random chance agreement rate, and chance-adjusted agreement rate (Cohen's kappa). Explain the calculations you conduct.

1.5 Aggregate to final dataset (10pts)

Create the final annotated dataset, by combining the two sets of annotations. For cases where your two annotators agree, you should just use that label. But where they disagree, you will have to adjudicate and decide who's right.

Deliverable: What principles (if any) did you use to make these decisions? (Explain in your writeup). Also, include a copy of your final dataset in CSV format, as 'final.csv' in your code/data submission.

2 NLP experiments (40pts)

OK great, you have some labeled data. What about NLP modeling?

Conduct experiments to build and test classifiers on your dataset. This will be a little difficult, since it is very small; fancier machine learning methods may not work well.

Since the dataset is so small, please report evaluations from cross-validation, since that more efficiently uses all datapoints for evaluation. (A static train/test split is often better to use, if you have a large enough dataset.) Make sure to describe decisions you made in how cross-validation is set up. (If you use premade crossval routines, like from scikit-learn, you must describe what they do, and how evaluation metrics are calculated, in a manner specific enough to be interpretable and replicable.) Also describe any hyperparameter tuning or selection that you conducted.

2.1 Tokenization (10pts)

Choose a word tokenizer for your experiments. (INLP 4.3 has suggestions.) Justify it.

2.2 BOW LogReg (20pts)

Create a logistic regression model based on features of the presence or count of words.

2.2.1 Feature preprocessing (10pts)

Describe the features and preprocessing you do. You probably want to clean things a bit more beyond simply lowercasing the words. Also, choose a normalization method to normalize the features before using them for training the model. Justify your decisions.

2.2.2 Results (5pts)

Report overall accuracy, as well as the precision and recall for each class. We recommend you use the scikit-learn version of LogisticRegression with L2 regularization, and probably with DictVectorizer to aid construction of the feature vectors.

2.2.3 Model insight (5pts)

Look at your features' weights. Show the top-10 highest weighted words for each category. What do they indicate, if anything? Also, choose and show 10 incorrectly classified samples. Try to analyze why they are classified into the wrong classes? Can you learn any insight from these samples to help you improve your model?

2.3 N-grams (10pts)

Try adding n-gram features to your model. Note that typically a model with n-gram features includes counts/presence for all 1-grams, 2-grams, and so on up to n-grams, so the feature representation always contains the more basic information.

Describe what you did, and the results. Also, compare the results of Section 2.2.2 and the n-gram model in an overall results table, with one row per feature configuration.

2.4 Extra Credit (15pts)

If you don't want to do the extra credit, you can stop here! Otherwise keep reading. The extra credit can give up to 15 points extra (out of 100 for the rest of homework).

Please try some feature engineering here and compare the results. You must include this ablation experiment:[1]

- BOW features, but disable preprocessing from 2.2.1. Simply use unigrams as raw strings without normalization.

Also experiment with several new classes of features. Be creative and try whatever you like, such as these, others not listed here, or combinations thereof:

- Heuristics, regular expression patterns, and/or keywords suited for your task (for example, the negation heuristic in the reading)
- Counts/presence based on external resources, such as emotion lexicons (say, the NRC Lexicon), or distributional resources like word clusters (e.g. the CMU Twitter word clusters) or word embeddings (GLOVE). We'll cover word embeddings later in the course but feel free to use now if you can explain them.
- Vary the machine learning model, such as switching to L1 regularized logistic regression, or a different model than logistic regression. scikit-learn includes many options.

Report and compare all the accuracy of these experiments. Include all numerical results in a single results table with one row per model configuration. Write an overview of your findings. Which preprocessing or features were the most useful? Which hurt performance? Do you have any hypotheses why this might be the case?

s

[1] In artificial intelligence, an "ablation experiment" is when you take a model/system, then remove one thing to see what happens. This gives a carefully controlled experiment between the two system variations. If performance drops a lot, that implies the thing you took out is important.