Web Science 2020: Final Project

Last updated 19 March 2020
This project description is likely to be adapted throughout the course.

The project will be graded as a whole and all parts of this project are compulsory. The project should be completed **individually**. You should submit a report detailing the project, what you have implemented, and your results and observations no later than **Friday 3 April 2020 at 12h00**. The format of the report should be a PDF document using the ACL template, no more than 4 pages (not including references, if needed). You should turn the report in on Absalon.

The aim of the project is to create a question recommender system, an answer quality assessment system, and an "Answer Detector" system, all from crowdsourced data collected from the web. The project also requires students to evaluate how their models generalize to unseen topics, and analyze errors resulting from domain shift. 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.

1 Week 6 (4 February)

The purpose of the first week is to familiarize yourself with the programming basics needed for the project.

You will be given a dataset of 1066 question-answer pairs. Please read the README file, so you understand the format. The dataset will be made available on Absalon.

- Load the dataset and compute the topic-wise statistics of the number of questions and mean/standard deviation of question and answer length
- Train a classifier to predict the topic of a question. (In Weeks 7-8 you will re-use this part to predict the quality of answers) and report your results in terms of both accuracy and F1 score.

  We suggest using a bag-of-words representation of each document and a Random Forest as the classification model. Perform a stratified split of the provided data into 80% for training, 10% for validation, and 10% for testing.
- Train a neural classification model (e.g., using the PyTorch library) to also classify the topic of each question. If you do not have the necessary

computer resources yourself, you can familiarize yourself with the Google Colab platform[1]. Google Colab provides free GPU resources for small-scale experiments (up to 12 hours at a time).

We suggest using trained word embeddings[2] as word representations, a recurrent neural network with LSTM cells as the model, and cross-entropy as the loss function. Use the Adam optimizer for training the model. Split the provided data in a stratified fashion into 80% for training, 10% for validation, and 10% for testing.

- The above is a description of what we expect you to implement, but you are allowed to expand upon this to explore more complex models.

We recommend the use of Python Version 3 with the SciPy stack[3], scikit-learn, and PyTorch. These are only recommendations, and you are allowed to use any programming language or libraries you want.

2 Week 7-8 (11 & 18 February)

The purpose of the second and third week is to (a) create a new dataset using crowdsourcing, and (b) predict the quality of the crowdsourced questions and answers. This is explained next.

2.1 Create a new dataset using crowdsourcing

Each student must determine if a given answer says "yes," "no," or "na" for a given question. "na" indicates that the answer does not say either yes or no; for example, it could be undetermined, or it might not be a yes or no question. You will be given a set of 100 question-answer pairs.

Example of a factual question from the "zoology" topic:

Q: Do wombats share their burrows during fires?

A: Yes, they do share... but not specifically because of the fires or out of a sense of altruism.

Example of a unanswerable question from the same topic:

Q: Are wombats kind?

A: (Many possible answers)

You will be given 100 unseen questions. To do this, we will generate a spreadsheet with all the questions-answer pairs, and assign 100 questions for each student. For each of these questions, you must do the following:

1. Label the answer with "yes," "no," or "na" (if neither yes or no applies) for how it answers the question. (For example, in the question, "Do wombats share their burrows during fires?" and the answer, "Yes, they do share... but not specifically because of the fires or out of a sense of altruism", the answer responds "yes" to the question.)

[1] `https://colab.research.google.com`
[2] e.g. word2vec skip-gram, `https://code.google.com/archive/p/word2vec/`
[3] `https://www.scipy.org/stackspec.html`

2. Label how much you like the question on a scale from 1-3, where (1) you do not like the question, (2) you have no preference, or (3) you like the question.
3. Rate the answer in terms of quality in how well the answer addresses the question, on a scale from 1-3, where (1) the answer does not answer the question, (2) answer somewhat answers the question, but not completely, (3) answer correctly and completely answers the question.
4. Give a binary score to indicate whether it is a factual question (1) or not (0).

Your completed dataset should be turned in through Absalon by Wednesday, 19 February, 12h00. The TA's will provide more details on how exactly to submit this.

2.2 Clean the new crowdsourced dataset

All the data provided by all students as described above will be collected. This will be our crowdsourced dataset. You will be given this crowdsourced dataset and must aggregate the scores given by many students to the same question and also clean the dataset.

- To determine the final question-answer label, use majority voting.
- To aggregate the answer quality scores of many users: Use majority voting to estimate the final answer quality score for each question. Compute the average if there's no consensus.
- To clean the dataset: Use majority voting to estimate the final factuality label for each question and discard those questions that are judged as non-factual (e.g. subjective).

You should report relevant dataset statistics and inter-annotator agreement in your final report. In addition, list any details of how you resolved ambiguity among different crowdworkers when no label received a majority vote.

2.3 Quality Prediction

The quality prediction component will evaluate the answer quality based on learned features using a neural model. To do this, you may reuse components from week 6, where instead of classifying the topic you will try to predict what quality rating an answer will receive. The course TAs will provide a list of question IDs corresponding to the training set and testing set. You can choose to build either a classifier and report accuracy and F1, or a regressor and report mean squared error (MSE).

3 Week 9-10 (25 February & 3 March)

The purpose of the ninth and tenth week is to get some experience in developing a recommendation system, with a particular emphasis on domain adaptation. You will use the user data in the crowdsourced dataset to recommend questions to users. Thus, you need to build profiles for users and recommend questions to users with similar profiles. You must implement a recommender that considers the topic and the question text (documents) as features to build the profile. You will then perform an error analysis, paying particular attention to domain adaptation – how well does your recommender trained on one topic perform on a different topic? Where does it fail, and why could this be? The domains are based on question topic: chemistry, climate-change, medical-science, technology, and psychology. Repeat the following steps for every topic pair, i.e. you will run the following 25 times, using one topic for training and another topic for testing. You will be given a training file with tuples of the form `(userId, questionId, rating)` and a testing file containing tuples of the form `(userId, questionId, liked)`. You should report accuracy on the test set in a 5x5 matrix, where a prediction is correct if the question you recommend was actually liked by the user.

Recommend users according to features in question text

- For each question text, compute the TF-IDF score for each word and use top n words as vectors so that each question is represented as a vector of features (words). "n" is the parameter that you can adapt.
- For each user, collect those questions that the user has rated as "like the question" based on the provided rating, and compute the average of their vectors as the profile vector.
- Use cosine similarity to calculate the similarity between every two users and create an N*N matrix, where there are N users.
- For each user in the test set, rank that user's row of similar users in a descending way and observe the top-K friends. Report the value of K that you choose.
- Select the items that these top-K friends have liked as your recommendations. Compare these to the liked items in the test data and report accuracy.

In the report, write down any observations you had e.g. what value of K did you pick? Do different values of K work for different domains? Which domain pairs seemed most compatible in terms of recommending questions and why do you think that is?

4 Week 11-12 (10 & 17 March)

You will now train an "Answer Detector" system on the crowdsourced data. Your system should be able to accurately predict if the answer to the question is "yes" based on the provided answer, if the answer to the question is "no" based on the provided answer, or if the answer to the question cannot be determined based on the provided answer. You will preform an in-depth error analysis in regards to the common mistakes your "Answer Detector" system makes. Again, we will provide you a split of training data and test data. Report your results on the full train and test split, as well as on training on a single topic and testing on another topic (for all topic pairs). Here you will also report prediction accuracy. 'x

4.1 "Answer Detector" Component

The "Answer Detector" component is essentially a sequence-based classifier, similar to the stance detection models you have learned about in class. The point of this module is to detect if the answer to the question is "yes" based on a given answer passage, if the answer to the question is "no" based on a given answer passage, or if the answer to the question cannot be determined based on a given answer passage. This part of the project is very open, and we encourage your to approach the problem in whatever way you think will perform well. You should measure your performance in terms of prediction accuracy.

4.2 Error Analysis Component

Do the same thing you did in the recommender system error analysis, but this time more in-depth, using techniques covered in the lab sessions.

- Print question-answer pairs that are wrong and look for patterns.
- Look between topics - Which topics did your system perform well or poorly on?
- Look at answer quality - is there a relationship between answer quality and your "Answer Detector" system?

5 Project Writeup Addendum

Due to lack of oral presentation, you will have an **additional 2 pages** (total of 6 pages for the full report) of space to write and reflect upon your work.

- For Part 3 (Recommender Systems), you should discuss what recommender systems method you used and explain why, comparing and contrasting that with other methods you have learned about in the course. Note that you do not need to experimentally compare different systems, you just need to add a discussion about pros and cons of your choice.
- You should also explain what method, if any, you have used for domain adaptation, and similarly explain why, comparing and contrasting that with other methods you've learned about in the course. Again, you do not need to experimentally compare different methods, but you need to add a discussion about pros and cons of your choice.
- For Part 4 (Answer Detector), you should explain what method you used and why, and how it relates to what you have learned about sentiment analysis and opinion mining in the course. For example, did you apply some approaches from sentiment analysis or opinion mining? Could you use those approaches directly or did you integrate those approaches with other components? Did you face some of the limitations or challenges presented in the lecture?

- 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?

- Describe the limitations of what you did. What could have lead to improvements in model performance? How could you have approached automatic recommendation, detecting answers and domain adaptation differently? What was particularly challenging about working with this dataset 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 lecture.

In all cases you should cite relevant literature which informed your choices in terms of modeling, analysis, etc.

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.