CSC 483/583: Programming Project (200 pts)

Recognizing Textual Entailment using SNLI

Due before 11:59 P.M., May 1

For this project you must submit:

- An archive containing all the source code.
- Binaries/jars (if a language that requires compilation is used) and/or instructions how to run the code.
- A PDF document that must contain at least:
  - Description of the command line to run the python code, with an example.
  - Description of the code. You don't have to describe every function implemented. But you should describe the main part of the code and indicate where each question is addressed.
  - Results, i.e., the output of your code, for all the questions that required programming.
  - Answers for all questions that do not require programming.

Answers are always graded by inspecting both code and documentation. Missing code yields no credit. Missing documentation yields partial credit (if code exists and produces correct results).

Because the credit for graduate students adds to more than 200, graduate students' grades will be normalized at the end to be out of 200. For example, if a graduate student obtains 220 points on this project, her final grade will be $220 \times \frac{200}{250} = 176$. Undergraduate students do not have to solve the problems marked "grad students only." If they do, their grades will not be normalized but will be capped at 200. For example, if an undergraduate student obtains 210 points on this project (by getting some credit on the "grad students only" problem), her final grade will be 200.

Data:

The SNLI corpus (`https://nlp.stanford.edu/projects/snli/`) is a collection of 570k human-written English sentence pairs manually labeled for balanced classification with the labels entailment, contradiction, and neutral, supporting the task of natural language inference (NLI), also known as recognizing textual entailment (RTE). It is a widely used benchmark for evaluating representational systems for text, especially including those induced by representation learning methods, as well as a resource for developing NLP models of any kind.

In the data files provided, you will encounter the following fields for each of the data point (see D2L project folder):

- sentence 1: The premise sentence.
- sentence 2: The hypothesis that a human came up with after reading the premise sentence.
- gold_label: This is the true/gold label that was arrived at by human annotators for the given hypothesis and premise. These will be either of NEUTRAL, ENTAILMENT or CONTRADICTION

  Examples:

  - Premise: Two women are embracing while holding to go packages.

    Hypothesis: Two woman are holding packages.

    gold_label: ENTAILMENT

  - Premise: Two women are embracing while holding to go packages.

    Hypothesis: The men are fighting outside a deli.

    gold_label:CONTRADICTION

  - Premise: Two women are embracing while holding to go packages.

    Hypothesis: The sisters are hugging goodbye while holding to go packages after just eating lunch.

    gold_label :NEUTRAL

- sentence1,2_parse: The parse produced by the Stanford Parser in Penn Treebank format.
- sentence1,2_binary_parse: The same parse as in sentence1,2_parse, but formatted for use in tree-structured neural networks with no unary nodes and no labels.

- annotator_labels (label1-5 in the tab separated file): These are all of the individual labels from annotators in phases 1 and 2. The first label comes from the phase 1 author, and is the only label for examples that did not undergo phase 2 annotation. In a few cases, the one of the phase 2 labels may be blank, indicating that an annotator saw the example but could not annotate it.
- captionID: A unique identifier for each sentence1 from the original Flickr30k example.
- pairID: A unique identifier for each sentence1–sentence2 pair.

Notes:

- For this project you really need only sentence 1 and sentence 2 and the gold label. However, you can choose to use the other features provided.
- If you see a '-' in place of label, ignore it. It means that data point doesn't have a label/ the human annotators couldn't agree upon one single label for it.
- The 3 relations/labels (NEUTRAL, ENTAILMENT or CONTRADICTION) are directional from PREMISE to HYPOTHESIS,

  e.g., if a premise ENTAILs a hypothesis, it is not necessary that hypothesis ENTAILs premise.
- captionID and pairID contain information that can be useful in making classification decisions and should not be included in model input (nor, of course, should either annotator_labels or gold_label).
- You shouldn't try too hard to over fit on the dev or training data since the data partition you will finally *test* your data will be different.
- Test data will be released closer to the final project submission date.
- json vs txt: While we provide both json and txt format files, we encourage you to use the jsonl file. Also note that both json and txt files differ in the order of the fields.

Questions:

Your project should address the following points:

1) (100 pts) Text classification: You must create a text classifier which, given a pair of premise and hypothesis (a.k.a sentence1 and sentence2), should classify them into any of the three classes via., NEUTRAL, ENTAILMENT or CONTRADICTION. Note that you need to decide yourself the features and the classifier that you will use.

2) (25 pts) Measuring performance: Measure the performance of your RTE system, using an F1-score against the dev/test dataset.

3) (25 pts) Feature and classifier selection: Describe all the features that you provided as input to your classifier. For example, are you using all the words in the premise (and or hypothesis) or a subset? Did you use some kind of syntactic parsing in additional lemmatized features? How did each feature group affect (increase or decrease) the performance of your system? What classifier did you use? Was it linear or non-linear? What prompted that choice?

4) (50 pts) Error analysis: Perform an error analysis of your best system. How many pairs were classified correctly/incorrectly? What problems do you observe for the pairs answered incorrectly? Try to group the errors into a few classes and discuss them. Lastly, what is the impact of stemming and lemmatization on your system? That is, what is your best configuration: (a) no stemming or lemmatization; (b) stemming, or (c) lemmatization? Why?

5) (50 pts) Improved model + Leader board (GRAD STUDENTS ONLY): Informed by the above error analysis, propose and implement at least one improvement over your initial solution. While this is not mandatory, try to get a score at par (or above) the leader in the SNLI score board `https://nlp.stanford.edu/projects/snli/`. As of Feb 2019, the highest accuracy (on test partition) in feature-based models section is 78.2%. For this task you have more freedom in choosing a solution, and I encourage you to use your imagination. For example, you could implement a positional index instead of a bag of words; you could use a parser to extract syntactic dependencies and index these dependencies rather than (or in addition to) words, etc.

Files:

These are the list of files that will be provided to you.

- snli_1.0_train.jsonl/txt: This is your training data. This will contain the aforementioned fields and it is in the jsonlines/txt format.
- snli_1.0_dev.jsonl/txt: This is your dev data. Once you have a trained model, you should validate/try it out/improve your model using the data given here.
- snli_1.0_test.jsonl/txt: This is the test data which will be used in your leader-board scores (for graduate students only).
- README.txt: the official readme file from SNLI competition.

Sample Data: Json

"annotator_labels": ["neutral", "entailment", "neutral", "neutral", "neutral"],

"captionID": "4705552913.jpg2", "gold_label":

"neutral", "pairID": "4705552913.jpg2r1n",

"sentence1": "Two women are embracing while holding to go packages.",

"sentence1_binary_parse": "((Two women) ((are (embracing (while (holding (to (go packages)))))) .))",

"sentence1_parse": "(ROOT (S (NP (CD Two) (NNS women)) (VP (VBP are) (VP (VBG embracing) (SBAR (IN while) (S (NP (VBG holding)) (VP (TO to) (VP (VB go) (NP (NNS packages)))))))) (. .)))",

"sentence2": "The sisters are hugging goodbye while holding to go packages after just eating lunch.",

"sentence2_binary_parse": "((The sisters) ((are ((hugging goodbye) (while (holding (to ((go packages) (after (just (eating lunch))))))))) .))",

"sentence2_parse": "(ROOT (S (NP (DT The) (NNS sisters)) (VP (VBP are) (VP (VBG hugging) (NP (UH goodbye)) (PP (IN while) (S (VP (VBG holding) (S (VP (TO to) (VP (VB go) (NP (NNS packages)) (PP (IN after) (S (ADVP (RB just)) (VP (VBG eating) (NP (NN lunch))))))))))) (. .)))"

Sample Data: txt

gold_label sentence1_binary_parse sentence2_binary_parse sentence1_parse sentence2_parse sentence1 sentence2 captionID pairID label1 label2 label3 label4 label5

neutral ((Two women) ((are (embracing (while (holding (to (go packages))))))) .)) ((The sisters) ((are ((hugging goodbye) (while (holding (to ((go packages) (after (just (eating lunch))))))))) .)) (ROOT (S (NP (CD Two) (NNS women)) (VP (VBP are) (VP (VBG embracing) (SBAR (IN while) (S (NP (VBG holding)) (VP (TO to) (VP (VB go) (NP (NNS packages)))))))) (. .))) (ROOT (S (NP (DT The) (NNS sisters)) (VP (VBP are) (VP (VBG hugging) (NP (UH goodbye)) (PP (IN while) (S (VP (VBG holding) (S (VP (TO to) (VP (VB go) (NP (NNS packages)) (PP (IN after) (S (ADVP (RB just)) (VP (VBG eating) (NP (NN lunch))))))))))))) (. .))) Two women are embracing while holding to go packages. The sisters are hugging goodbye while holding to go packages after just eating lunch. 4705552913.jpg2 4705552913.jpg2r1n neutral entailment neutral neutral neutral