DS-GA 1011 Fall 2019
Bag-of-Words based Natural Language Inference*

September 2019

In this assignment, you will train a Bag-Of-Words encoder to tackle the the Stanford Natural Language Inference (SNLI) and Multi-Genre Natural Language Inference (MNLI) task.

1 Dataset

The SNLI task poses the following problem: Given two pieces of text, a *premise* and a *hypothesis*, you (or a model) have to choose one of the following:

1. The premise **entails** the hypothesis.
2. The premise **contradicts** the hypothesis.
3. The premise neither entails nor contradicts the hypothesis, and is thus **neutral** to it.

For example:

- Premise: *A man inspects the uniform of a figure in some East Asian country.*
  Hypothesis: *The man is sleeping.*
  This is a **contradiction**, since the man cannot both be sleeping and inspecting the uniform.
- Premise: *A soccer game with multiple males playing.*
  Hypothesis: *Some men are playing a sport.*
  This is an **entailment**, because if the former is true, the latter must be true.
- Premise: *An older and younger man smiling.*
  Hypothesis: *Two men are smiling and laughing at the cats playing on the floor.*
  This is **neutral**, because the premise does not contain any information about cats that the hypothesis posits.

*Adapted from DS-GA 1011 Fall 2018

Being based on natural text, there will be some ambiguity regarding the answers. Nevertheless, the NLI-style tasks have shown to be effective for training models to capture aspects of semantic information from text. Refer to `https://nlp.stanford.edu/projects/snli/`, or the original paper (`http://nlp.stanford.edu/pubs/snli_paper.pdf`) for more details.

You will be provided with pre-tokenized training and validation sets of the SNLI in a `.tsv` format. The data set has already been reduced to include 100,000 examples in the training set and 1,000 examples in the validation set.

2 Model

At its core, the SNLI is a 3-class classification problem, where the input is two separate strings of text. Choosing what approach to use to integrate information from both strings of text is where the problem becomes interesting, and various approaches have been proposed. In our case, we will take the following simple approach:

1. We will use a BoW encoder to map each string of text (hypothesis and premise) to a fixed-dimension vector representation. At this point, we have one vector representation corresponding to hypothesis and one for premise.

2. We will interact the two representations and perform classification on this. For combining the two vector representations, we could simply concatenate the representations and a make a vector. You're required to try atleast one other method of combining the representations - you could take a sum, element-wise product etc.

3. Once we've the combined representation, now we've to do a 3-class classification problem on this input vector. For this we will use the following models:

   (a) Logistic Regression

   (b) A fully connection neural network with two hidden layers

3 Task

3.1 Training on SNLI

Your task is to implement, perform hyperparameter tuning, and analyze the results of the model.

Because the SNLI data set is relatively large, we recommend starting the assignment early. Likewise, because you are training on only a subset of the full data and using a relatively simple BoW encoder, you should not expect to get accuracies comparable to the published leaderboard.

Perform tuning over **at least two** of the following hyperparameters:

- Varying the size of the vocabulary and embedding dimension.
- Experiment with different ways of interacting the two encoded sentences (concatenation, element-wise multiplication, etc) (*This is required*)
- Optimization hyperparameters: Optimizer itself (SGD vs Adam), learning rate and whether or not you use linear annealing of learning rate (learning rate is reduced linearly over the course of training).
- Regularization (e.g. weight decay, dropout)

For each mode of hyperparameter tuning, report the training and validation losses and accuracies (in plotted curves), as well as the number of trained parameters in each model. Discuss the implications of each hyperparameter tuned. Finally, take your best model based on validation performance and highlight 3 correct and 3 incorrect predictions in the validation set. Describe why the model might have gotten the 3 incorrect predictions wrong.

We recommend that you refer to the PyTorch code from the lab session on BoW model to tackle the assignment. Please let us know if you have questions on Campuswire or visit us during office hours.

3.2 Evaluating on MultiNLI

The Multi-Genre Natural Language Inference (MultiNLI) task is a variant of SNLI which covers spoken and written text from different sources, or "genres".

You task is to take your best trained model, one each for Logistic Regression and Neural Net based classifier, and evaluate them on the provided MultiNLI data, for each genre. This means you should have a table of validation accuracies for Logistic Regression and Neural Net based classifier, for each genre. Briefly discuss the results, comparing them both to SNLI accuracies as well as comparing the results within MultiNLI across genres.

You are provided with subsets of the MultiNLI training and validation data set in the same `.tsv` format as the above, except with an addition "genre" column. Perform your evaluation on the validation data set for MultiNLI separately for each genre. (The training subset for MultiNLI is only used for the section below.)

3.3 Fine-tuning on MultiNLI

In this section, we will *fine-tune* our best trained SNLI model (you can pick the best over Logistic Regression and Neural Net based classifier in this case[1]). This means that we will take our pre-trained model, and training it further on a new task / data set, to fine-tune it for that task. Often, fine-tuning is done on tasks where we may have less data but wish to use information from both a pre-training task where we have a lot of data, and data from the new task.

[1]The choice should be obvious even before you do the assignment :-)

Specifically, we will fine-tine our SNLI model on training data for each MultiNLI genre (training set), and evaluate it on that genre (validation set). Since our SNLI model is already trained, we do not need (nor want) to over-train our model–hence, you can continue training just on a small number of epochs, or use a lower learning rate. Take note that we will fine-tune separately for each genre, so be sure to save/load the `state_dict` of your respective models when you fine-tune on a new genre.

In short:

1. For each genre, take your best SNLI model and train it a number of epochs on the training set of each genre.

2. Evaluate your fine-tuned models on that genre.

Report the validation accuracies for each genre with and without fine-tuning. If you have the time, further evaluate each fine-tuned model on every other genre, to see if that fine-tuning carries over to the other genres.

3.4 Pre-Trained Word Embedding

Repeat Sections 3.1 and 3.2 by using frozen pre-trained word embeddings. To re-emphasize, we will use frozen word embeddings and will not train the word embeddings part of the encoder. In essense, you'll only be training your classifier here. We recommend using one of the fastText word vector sets.[2]. You can use the hyper parameters that worked best for you in sectoin 3.1, however, you're free to search over this space as well.

3.5 (Bonus) Analyzing Learned Word Embedding

In Section 3.1 you learned word embedding. Select your best model (for both classifier) and find the 10 most *similar* pairs of words. How you define *similar words* is up to you, but you are expected to justify your notion of similarity. Do these pairs of similar words remain unchanged for the word embedding you use in Section 3.4? If you're still not tired of working on this homework, you could check how the word embedding changed from your model in section 3.1 to when you did fine tuning in Section 3.3 - an idea would be to check the embedding which *changed the most*[3] and report those words. Can you relate these words to the domain you trained on?

4 Submission

Your submission will be a LaTeX-formatted PDF with analysis and discussion of results (maximum of 4 pages), as well as a zip file with your code.

[2]https://fasttext.cc/docs/en/english-vectors.html

[3]Again, how would you define this change? Maybe you could look at the angle or you could look at the distance between the embeddings. You're free to come up with your metric but you should justify your choice of metric.