CS 6320.002: Natural Language Processing
Fall 2020

Homework 2 Written Component — 45 points
Issued 09 Sept. 2020
Due 11:59pm CDT 23 Sept. 2020

Deliverables: Answers can be typed directly into Gradescope. LaTeX can be hand-typed or generated using Mathpix Snip. See the assignment guide for more details.

What does it mean to "show your work?" Write out the math step-by-step; we should be able to clearly follow your reasoning from one step to another. (You can combine "obvious" steps like simplifying fractions or doing basic arithmetic.) The point of showing your work is twofold: to get partial credit if your answer is incorrect, and to show us that you worked the problem yourself and understand it. We will deduct points if steps are missing.

1 Sentiment Analysis & Classification

The problems in this section are based on the material covered in Week 3

1.1 Naive Bayes — 10 points

We have a training corpus consisting of three sentences and their labels:

- The cat sat on the mat, 0
- The dog sat on the log, 1
- The fish sat in the dish, 0

A. Suppose we train a Naive Bayes classifier on this corpus, using maximum likelihood estimation and unigram count features without any smoothing. What are the values of the parameters $p(c)$ and $p(f|c)$ for all classes c and features f? You can simply list the parameters and their values; no need to show the arithmetic. You can skip parameters with value 0.

B. What class would our Naive Bayes classifier predict for the test sentence "The cat sat on the fish"? Show your work, ie. show the calculations for the predicted probabilities of both classes.

1.2 Logistic Regression — 5 points

The last step of the programming component asks you to get the top k most important features for your sentiment classifier. When doing this, why do we sort by absolute value? Explain why we do this rather than sorting by the raw weight values (1-2 sentences).

1.3 Gradient Descent — 5 points

Suppose you are training a model using stochastic gradient descent, and you have a held-out validation set to check the performance of your model. Your loss function gives the following values on the training and validation sets as you train:

Training Steps	Training Loss	Validation Loss
100	0.9494	0.9952
200	0.8652	0.8921
300	0.7345	0.7671
400	0.6253	0.6937
500	0.5145	0.5877
600	0.4112	0.4528
700	0.3434	0.3514
800	0.2346	0.3133
900	0.1384	0.3240
1000	0.1261	0.3258

You have saved a copy of your model every 100 training steps. Which of the 10 saved models should you use for testing? Explain your answer (1-2 sentences).

2 Part-of-Speech Tagging

The problems in this section are based on the material covered in Week 5.

2.1 HMMs and the Viterbi Algorithm — 15 points

Suppose we have a training corpus consisting of two tagged sentences:

-

The	can	is	in	the	shed
DT	NN	VB	PP	DT	NN

-

The	dog	can	see	the	cat
DT	NN	VB	VB	DT	NN

A. Suppose we train a simple HMM part-of-speech tagger on this corpus, using maximum likelihood estimation, bigram tag transition probabilities, and a single meta-tag `<s>` (the start tag). What are the values of the parameters $p(t_i|t_{i-1})$ and $p(w_i|t_i)$ for all tags t and words w? You can simply list the parameters and their values; no need to show the arithmetic. You can skip parameters with value 0.

B. What parts of speech would the trained HMM tagger in the previous problem predict for the test sentence “The cat can see the can,” using Viterbi decoding? Show your work, ie. the dynamic programming table V (you use the `array` environment in LaTeX to format this).

C. Suppose we have an HMM tagger that uses 5-gram tag transition probabilities, ie. the parameters are $p(t_i|t_{i-1}, t_{i-2}, t_{i-3}, t_{i-4})$ and $p(w_i|t_i)$. Let T be the number of tags in the tagset, and let n be the length of the input sequence to be tagged. What is the runtime, in big-O, of the vanilla Viterbi algorithm for this HMM? What is the runtime if we use beam search Viterbi with beam size k? Briefly explain your answers (a single sentence is fine).

2.2 Tagsets — 5 points

The Penn Treebank tagset is not the only one out there; there is also the Universal Dependencies tagset, which has less than half as many tags. For example, instead of a different tag for each tense of verb, Universal Dependencies has a single tag for all verbs, regardless of their tense. What are some advantages and disadvantages of using a smaller tagset, as opposed to a larger one? Give at least one advantage and one disadvantage and briefly explain (a single sentence each is fine).

2.3 MEMMs and Feature Engineering — 5 points

Another powerful feature type for part-of-speech tagging MEMMs, in addition to word and tag n-grams, are word and tag *skip-grams*. For example, from the sequence "The happy cat", we can get two bigrams, `(the, happy)` and `(happy, cat)`; one trigram, `(the, happy, cat)`; and one skip-gram, `(the, cat)`. Why do we use skip-grams when we already have bigrams and trigrams? What advantages do skip-gram features offer? Give at least one advantage and briefly explain (a single sentence is fine).