CS 585 Problem Set 3, Part 1

October 22, 2014

Name: _____

Collaborators _____
(See course policy here)

Please submit a paper copy of this at the start of lecture on Tuesday Nov 4.

These questions have 30 total points.

Intro: Structured Perceptron

Recall that the perceptron learning algorithm updates a parameter vector θ by iterating through training examples. For an example (x_i, y_i), you take two steps. First, predict a structure by computing the highest-scoring structure $y^* = \arg\max_y \theta^\mathsf{T} f(x_i, y)$ (the "decoding" prediction problem). Second, update the parameter vector:

$$\theta^{(new)} := \theta^{(old)} + \eta g \tag{1}$$

Here, η is a fixed step size (e.g., 0.01). The vector g is the difference between the gold feature vector versus the predicted feature vector:

$$g = f(x_i, y_i) - f(x_i, y^*)$$

We're using the letter "g" for this because it's a gradient of a particular objective function, though for this problem we can just think of it as an "update vector."

There are many possible options for the feature function $f(x, y)$. A first-order sequence model perceptron uses feature functions for local transitions (between neighboring tags) and observation features (for individual tags), which means the Viterbi algorithm can be used to decode. But more generally, the structured perceptron can be used any time you have a model with a decoding algorithm to solve that argmax structure prediction problem. The questions in this section are general for any structured perceptron.

1 Averaged (Structured) Perceptron

Both in theory and in practice, the predictive accuracy of a model trained by the structured perceptron will be better if we use the average value of θ over the course of training, rather than the final value of θ. This is because θ wanders around and doesn't converge well, because it overfits to whatever data it saw most recently.

After seeing t training examples, define the *averaged parameter vector*:

$$\bar{\theta}_t = \frac{1}{t} \sum_{t'=1..t} \theta_{t'} \tag{2}$$

where $\theta_{t'}$ is the parameter value of the perceptron algorithm after t' updates. For training, you still run the perceptron algorithm in the normal way using the current θ, but you save some information so once you're done training, you can calculate $\bar{\theta}$, which you want to make new predictions on new data.

In this section we'll derive an efficient algorithm to implement the averaged perceptron. First, assume that always $\eta = 1$. Define g_t to be the update vector g as described in update (1) at iteration t. With this, we write the perceptron update as

$$\theta_t = \theta_{t-1} + \eta g_t.$$

Thus in this notation, the averaged perceptron looks something like:

- Initialize $t = 1, \theta_0 = 0$
- For each example i (iterating multiple times through dataset),
    - Predict $y^* = \arg\max_y \theta^\mathsf{T} f(x_i, y)$
    - Update $\theta_t := \theta_{t-1} + \eta g_t$
    - (store something for later)
    - $t := t + 1$
- Calculate $\bar{\theta}$ based on stuff you stored during training

Now, we define an auxiliary vector S that we also update at every iteration using

$$S_t = S_{t-1} + (t-1)\eta g_t. \tag{3}$$

Our proposed algorithm computes $\bar{\theta}_t$ as

$$\bar{\theta}_t = \theta_t - \frac{1}{t} S_t. \tag{4}$$

Question 1.1. *[2 points]* What is the advantage of computing $\bar{\theta}_t$ using (4) rather than a more naive approach that manually averages all $\theta_{t'}$, as in (2)?

Question 1.2. *[2 points]* Using (2), for $t = 1, 2, 3$, what is $\bar{\theta}_t$ in terms of g_1, g_2, g_3?

Question 1.3. *[2 points]* Using (3), for $t = 1, 2, 3$, what is S_t in terms of g_1, g_2, g_3?

Question 1.4. *[4 points]* Using (4), show that the value computed by the proposed algorithm is correct for $\bar{\theta}_2$ and $\bar{\theta}_3$.

Question 1.5. *[12 points]* The previous step proved the correctness of (4) for a base case. Now, prove the overall correctness of (4) using induction. Namely, assume that (4) produces the correct value at step $(t-1)$ and prove that it the combination of (3) and (4) produces the right value at step t.

HINT: We recommend beginning your proof with the following three lines.

$$\bar{\theta}_t = \frac{1}{t} \sum_{t'=1..t} \theta_{t'} \tag{5}$$

$$= \left(\frac{1}{t} \sum_{t'=1..(t-1)} \theta_{t'} \right) + \frac{1}{t} \theta_t \tag{6}$$

$$= \frac{t-1}{t} \bar{\theta}_{t-1} + \frac{1}{t} \theta_t \tag{7}$$

2 Sparse Features

A common sitatuation in NLP is to have very *sparse* feature vectors. Namely, that the number of nonzero elements of each g is small relative to its length. Assume that g is of length J, but that it never has more than $L << J$ nonzero elements.

Question 2.1. *[2 points]* In terms of J and L, what is the time complexity per step of the perceptron update (Eq. (1)) for a naive implementation that does not exploit the sparsity structure of g? That is, an implementation that loops over all feature types, including ones that might potentially have zero values in g.

Question 2.2. *[2 points]* What is a smart way to implement Eq. (1) such that the time complexity of each update is much less than J?

Question 2.3. *[2 points]* After T steps of the perceptron algorithm, what is the most number of nonzero elements that θ_t can have in terms of J and L?

Question 2.4. *[2 points]* Why do sparse features often occur in NLP applications?