Machine Learning & Data Mining
Homework 5

Caltech CS/CNS/EE 155
February 2^{th}

This set is due 2:30pm, February 16^{th}, via Moodle. Note that you get two weeks for this set. You are free to collaborate on all of the problems, subject to the collaboration policy stated in the syllabus. It is highly recommended that you start the set early, as it is quite time consuming.

1 Conditional Random Fields

In the last set, we used Hidden Markov Models (HMMs) in order to predict Ron's mood based on his music interests. Now, we will train a Conditional Random Field (CRF) for the same task. The CRF will use features of the form Y^t, Y^{t+1}. The Y^t's are the hidden states (moods), and the X^t's are the observables (music genres). With a CRF, the probability of $Y^{1:T}$ given a corresponding $X^{1:T}$ can be represented as:

$$P(Y^{1:T}|X^{1:T}) \propto \exp\left[\sum_{t=1}^{T}\langle\theta^1, \phi^1(X^t, Y^t)\rangle + \sum_{t=1}^{T}\langle\theta^2, \phi^2(Y^t, Y^{t-1})\rangle\right],$$

where θ denotes the model parameters.

Question A [50 points]: Implement CRF gradient descent. Train the CRF and print out your model parameters. Submit neat commented code.

Hints:

- Make assumptions when necessary but clearly state your assumptions.
- There are files with skeleton and helper functions that are very useful, but it is not required for you to use them.
- There is only one training example, so stochastic gradient descent and gradient descent do the exact same thing
- Start with $\theta = 0$, and terminate gradient descent under a reasonable stopping condition (e.g., the relative decrease in log loss compared to the first iteration is less that 0.01 or 0.001).
- You'll probably have to try a range of step sizes to find one that has good convergence behavior.
- There is no need to use regularization in your implementation. However, you may find your gradient descent code behaving more stable if you introduce a very small amount of regularization.
- Write your code in a modular way. In particular, gradient descent requires running the Forward-Backward algorithm, which should ideally be implemented as a function that you call during gradient descent. It's also fine to write subroutine functions to make the Forward-Backward algorithm more modular as well.

Question B [20 points]: Implement the Viterbi algorithm for CRF. Submit neat commented code.

Hints:

- Make assumptions when necessary but clearly state your assumptions.
- The code should be essentially identical as Viterbi for HMMs, so you should be able to copy-paste most of the solution and change just a couple things.

Question C [10 points]: Train a supervised HMM on Ron's music tastes using the HMM M-step code you wrote for the previous set (or use the class solution). Using only the sequence of genres from the training set (i.e., the x from the training set), compute the most probable mood sequence $\arg\max_y P(y|x)$ using both the trained HMM and trained CRF. Measure the Hamming loss between the predicted sequences and the ground truth in the training set. In other words, we're measuring how well the two models (HMM and CRF) were able to fit to the training set.

Question D [10 points]: (Short Answer) State the optimality condition of CRF training (i.e., what condition must be satisfied if the gradients of the model components are 0). Compare and contrast the optimality condition to the optimal solution to HMMs.

Question E [10 points]: (Short Answer) Under what conditions do you feel a CRF is a more fitting model than an HMM? What situations do you think HMMs are more applicable?