CPTS 580 Structured Prediction: Algorithms and Applications, Spring 2017
Homework #1
Due Date: Tue, Feb 14

NOTE 1: Please use a word processing software (e.g., Microsoft word or Latex) to write your answers and submit a printed copy to me. The rationale is that it is sometimes hard to read and understand the hand-written answers. Thanks for your understanding.

NOTE 2: Please ensure that all the graphs are appropriately labeled (x-axis, y-axis, and each curve). The caption or heading of each graph should be informative and self-contained.

1. (**15 points**) Tom wants to apply the LaSO framework to the problem of co-reference resolution that arises in natural language processing. Coreference resolution is a structured prediction problem where the set of mentions $m_1, m_2, \cdots, m_D$ extracted from a document D correponds to a structured input x and the structured output y corresponds to a partition of the mentions into a set of clusters $C_1, C_2, \cdots, C_k$. Each mention m_i belongs to exactly one of the clusters C_j. Tom needs your help in designing a concrete search space. Can you give some concrete search space definitions, and discuss their pros and cons in terms of learning an accurate heuristic function to guide the search process?

2. (**85 points**) Please implement the online structured perceptron training algorithm for sequence labeling problems and experiment with two sequence labeling datasets: handwriting recognition, and text-to-speech mapping.

   In a sequence labeling problem, the structured input $x = (x_1, x_2, \cdots, x_T)$ is a sequence of input tokens, where each input token x_i is represented as a m-dimensional feature vector; and the structured output $y = (y_1, y_2, \cdots, y_T)$ is a sequence of output labels, where each output label y_i comes from a label set $\{1, 2, \cdots, k\}$. You were provided with a set of training examples $\mathcal{D} = \{(x, y^*\}$, where y^* is the correct structured output for the structured input x.

   You need to learn a scoring function $S(x, y) = w \cdot \phi(x, y)$, where $\phi(x, y) \in \Re^d$ is a joint feature representation over a structured input x and candidate structured output $y \in Y(x)$ and $w \in \Re^d$ corresponds to the weights (or parameters) of the cost function. Essentially, you need to learn the weights $w \in \Re^d$ from the given training data.

Algorithm 1 Randomized Greedy Search (RGS) Inference

Input: x = structured input, ϕ = joint feature function, w = weights of scoring function, R = number of restarts
Output: $\hat{y}$, best scoring structured output

```
Initialize the best scoring output ŷ randomly
Initialize the best score S_best as S(x, ŷ) = w · φ(x, ŷ)
for R iterations do
    Pick a random structured output y_start as starting point
    while NOT reached local optima do
        Perform a greedy search step by considering all one-label changes.
        Score each candidate structured output (corresponding to one-label change) y:
        S(x, y) = w · φ(x, y)
        Pick the best scoring candidate output and continue search
        Update the best scoring output ŷ and the corresponding best score S_best
    end while
end for
return the best scoring output ŷ
```

Algorithm 2 Online Structured Perceptron Training

Input: $\mathcal{D}$ = Training examples, ϕ = joint feature function, R = number of restarts, η = learning rate, MAX = maximum training iterations

Output: w, weights of the scoring function

```
Initialize the weights of the scoring function w = 0
for MAX iterations or until convergence do
   for each training example (x, y*) ∈ D do
      Make prediction: ŷ = RGS-Inference(x, φ, w, R)
      Compare ŷ and y* to check for error
      If error, perform weight update:
      w = w + η(φ(x, y*) − φ(x, ŷ))
   end for
end for
return weights w
```

(a) Implement the structured perceptron training algorithm and the randomized local search inference algorithm as shown in the above pseudo-code.

(b) Plot the Hamming accuracy over the training and testing set as a function of the number of online learning iterations (say MAX=100, $\eta = 0.01$, $R = 20$); for both handwriting and text-to-speech mapping problem.

(c) Repeat (b) with different feature representations ϕ.
First-order: unary + pairwise features ($d = m \cdot k + k^2$)
Second-order: unary + pairwise + triple features ($d = m \cdot k + k^2 + k^3$)
Third-order: unary + pairwise + triple + quadruple features ($d = m \cdot k + k^2 + k^3 + k^4$)

(d) Fix the number of online learning iterations (MAX=100) and learning rate ($\eta = 0.01$) to reasonable values (based on (b)); and plot the Hamming accuracy for first-order representation as a function of the number of restarts R=10, 25, 50, 100, 200.

(e) How will you diagnose the performance of learning algorithm? Please list your ideas and explain your intuition and rationale. Implement your ideas to perform diagnosis and test your hypotheses.

(f) Please feel free to try other ways of randomizing the local search and list your observations in comparison to RGS.