CptS 577 Structured Prediction and Intelligent Decision-Making, Spring 2019
Homework #1

Due Date: Tue, Feb 12

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 at the begining of class on Feb 27. The rationale is that it is sometimes hard to read and understand the hand-written answers.

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. (**100 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

```
1: Initialize the best scoring output $\hat{y}$ randomly
2: Initialize the best score $S_{best}$ as $S(x, \hat{y}) = w \cdot \phi(x, \hat{y})$
3: for $R$ iterations do
4:     Pick a random structured output $y_{start}$ as starting point
5:     while NOT reached local optima do
6:         Perform a greedy search step by considering all one-label changes.
7:         Score each candidate structured output (corresponding to one-label change) $y$:
           $S(x, y) = w \cdot \phi(x, y)$
8:         Pick the best scoring candidate output and continue search
9:         Update the best scoring output $\hat{y}$ and the corresponding best score $S_{best}$
10:    end while
11: end for
12: return the best scoring output $\hat{y}$
```

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

```
1: Initialize the weights of the scoring function $w = 0$
2: for MAX iterations or until convergence do
3:    for each training example $(x, y^*) \in \mathcal{D}$ do
4:       Make prediction: $\hat{y}$ = RGS-Inference($x$, $\phi$, $w$, $R$)
5:       Compare $\hat{y}$ and $y^*$ to check for error
6:       If error, perform weight update:
         $w = w + \eta(\phi(x, y^*) - \phi(x, \hat{y}))$
7:    end for
8: end for
9: return weights $w$
```

(a) Implement the structured perceptron training algorithm and the randomized local search inference algorithm from scratch as shown in the above pseudo-code. Optionally, you can try exploring Illinois-SL library (`https://cogcomp.org/page/software_view/Illinois-SL`) that allows you to define your joint feature function ϕ and inference algorithm, and automatically performs training using structured perceptron or structured SVM. This will allow you to debug your own implementation.

(b) Plot the Hamming accuracy over the training and testing set as a function of the number of online learning iterations (say MAX=20, $\eta = 0.01$, $R = 10$); 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=20) 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=1, 5, 10, 20.

(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.

Instructions for Code Submission and Output Format.

Please follow the below instructions. It will help us in grading your programming part of the homework. We will provide a dropbox folder link for code submission.

- Supported programming languages: Python, Java, C++
- Store all the relevant files in a folder and submit the corresponding zipfile named after your student-id, e.g., 114513209.zip
- This folder should have a script file named

  ```
  run_code.sh
  ```

  Executing this script should do all the necessary steps required for executing the code including compiling, linking, and execution
- Assume relative file paths in your code. Some examples:

  ```
  ``./filename.txt'' or ``../hw1/filename.txt''
  ```
- The output of your program should be dumped in a file named "output.txt"
- Make sure the output.txt file is dumped when you execute the script

  ```
  run_code.sh
  ```
- Zip the entire folder and submit it as

  ```
  <student_id>.zip
  ```