CSE 555 Spring 2011 Project

Jason J. Corso
Computer Science and Engineering
SUNY at Buffalo
jcorso@buffalo.edu
Date Due Monday 2 May 2011

This document contains the project description. It is a programming-based project with a written report. You are free to choose your language (Matlab, Python, C/C++ or Java). You should program and write alone but should discuss with classmates (away from the terminal).

You may NOT use ANY off-the-shelf library or tool other than for basic IO and basic computation.

The project is worth 100 pts equally divided between (1) implementation, (2) report, and (3) accuracies.

The report should be max 4 pages in the two-column IEEE format (`http://www.ieee.org/web/publications/authors/transjnl/index.html`). You should briefly describe the findings (not the methods).

Three extra-credit options are associated with the project for an additional X points (the actual project here is considered to be 100 points).

Pen-Gesture Recognition with Hidden Markov Models

1. Data: `http://www.cse.buffalo.edu/~jcorso/t/CSE555/project-hmm-data.tar.gz`. This data has 5 files, one for each vowel of the alphabet. These are xml files and the format is self-explanatory. Take the odd indexed entries as training data and the even ones as testing data.

   NOTE: One of the extra credit options below allows you to use a different data set. If you choose to opt for that extra-credit option, then only use the other data set for the whole project, not the one I am providing here.

2. Implement a spatial clustering algorithm (e.g., K-Means) on the 2D training data.

3. Implement the standard Hidden Markov Model. The input will be the cluster index associated with each 2D point.

4. Train a separate HMM for each vowel (do separate clustering for each and then learn the HMM independently). Quantify accuracy against the training and testing set; for each candidate datum, compute its log-likelihood against each HMM and take the label of the HMM that gives the highest log-likelihood. Compute a confusion matrix of results. Describe your findings.

5. Vary the number of clusters in step 2 and the number of hidden nodes (and their connectivity) in step 3 and repeat the quantification. Do this for an additional three variants. Describe your findings.

   Option (+15pts) Implement Dynamic Time Warping and repeat the experiments.

   Option (+20pts) Implement an approach that would learn a single HMM that could use the Viterbi decoding to perform classification rather than a battery of Evaluations on single classifiers. Compare the accuracy again the earlier separate HMMs.

Option (+25pts) Rather than pen-gesture recognition, carry out the project on a real video action classification dataset. Do all of the same steps and the other two extra-credit options are also available to you. If you choose this extra-credit option, then you do not need to also work with the pen-gesture data. For this, use the KTH Dataset available for download at `http://www.nada.kth.se/cvap/actions/`. KTH is a six class dataset; work with them all. Working with this dataset may be more interesting to you but it is more open-ended and hence will be more challenging than the pen-gesture dataset because you have to decide how you will turn the video in a set of features on which to train the HMMs.

How and what to submit: The entire project, including the report, code (whether it is Matlab, Python, C or Java), should be tar'd and submitted via the CSE submit script.

1. Login to a department student Unix machine, hadar, metallica, nickelback, pollux, styx, timberlake (the code MUST work on the CSE machines).
2. Use "tar -cvf project.tar list-of-files-or-directories".
3. Then, type "submit_cse555 project.tar".

Include a README file to explain what is there, but only submit the code. If it is C/C++ code, you need to include a makefile that will compile on the CSE machines (it is thus encouraged to link to any libraries available there rather than submitting them). If it is Java code, you need to include an ant file that will compile on the CSE machines. In all cases, you need to include a script named "go" that will (once compiled) do everything requested in the assignment (in the case of C or Java, it is sufficient to generate image files of the plots and save them to reasonably named files). The script can be bash/tcsh for C/Java and should be a simple Matlab/Python script for Matlab/Python.