Carnegie Mellon University
15-826 Multimedia Databases & Data Mining
Fall 2013 - C. Faloutsos

Project1 Description: Insect Dataset

Contact TA: Vagelis Papalexakis, `epapalex@cs.cmu.edu`

1 Introduction & Dataset Description

In this project, you will work on time series classification, visualization and anomaly detection. In particular, you will focus on a dataset that categorizes insects based on the sound that their wings make when they fly.

There is a total of 11 species of insects (classes) in the dataset that you are given. For each species, you are given 100 time series, summing up to a total of 1100 time series. You can download the training data from the supporting web-page: `http://www.cs.cmu.edu/~epapalex/15826F13/`.

For tasks 1-3, you have to split this dataset into a training and a test portion, making sure that you have representatives of all classes in both the training and test set (e.g. you can take all time series for class 1, take a random sample that will be your training set, and what's left will be your test set; repeat that for all classes). Class labels are in the file `classLabel.txt` (which is in Matlab format in `classLabel.mat`, for your convenience). The first column is the filename of the data, the second column is the class label for that file, e.g., [1 6] means the file '0001.wav' is a wingbeat sound of insect species 6. We will evaluate your work on a separate test set that will **not** be given to you.

For the rest of the tasks, feel free to use the entire dataset.

Your code should be written in **Matlab** (alternatively, you may use Octave, but please make sure that the code that you will turn in, is compatible with Matlab).

2 Introductory Material

We will cover time series later in the class - we give the urls for last year's foils, below. In the meanwhile, you should check

- The insect-mining project web site `http://www.cs.ucr.edu/~eamonn/CE/contest.htm`
- The tutorial by Prof. Keogh in SIGMOD'07 `http://www.cs.ucr.edu/~eamonn/SIGKDD_2007.ppt`

- and the tutorial by the instructor [KDD'10] http://www.cs.cmu.edu/~christos/TALKS/10-KDD-tutorial/

3 Project Tasks

1. *Nearest Neighbor Classifier* Write code that does Nearest Neighbor classification; for the purposes of this project, it suffices to have one Nearest Neighbor (since it makes easier to compare distance functions, without having to tune the number of neighbors). Make your code generic, with respect to the distance function used.

2. *Standard Distance Functions* Apply your NN classifier using the following distance functions you have seen in class

   (a) Euclidean distance
   (b) Cosine similarity (actually, this is a similarity and not a distance function, so you should take that into account)
   (c) Time Warping distance
   (d) Cepstrum

   Check the lecture foils (eg., from [1] as well as the papers by Eamonn Keogh [2] starting from his KDD'12 one[3] For each of the above distance functions, report the classification accuracy.

3. *Custom Distance Function* Design a distance function that takes two time series and returns their distance. Feel free to use anything that you have learned in the class, including feature extraction (e.g. your distance function might first extract features from the time series). The goal here is to come up with a function that ideally outperforms the aforementioned distance functions (or at least performs equally well), in Nearest Neighbor classification.

   The distance function you will submit must be in the format of distanceFunctionName(s1,s2), where s1 and s2 are the sound signals, e.g, s1 = wavread('1.wav'), s2 = wavread('2.wav'); and the distance is dis = distanceFunctionName(s1,s2).

   You should describe in detail all the steps and the logic behind your distance function. You don't necessarily have to provide analytic proofs, but at least, you have to provide *intuition* in order to justify your choices.

4. *Visualization* After feature extraction, our data probably lives in a high dimensional space, thus, we cannot visualize it as it is. In class, you have seen several dimensionality reduction methods that serve the need to visualize high dimensional data. Implement and apply the following methods of high dimensional data visualization. See foils from last year's class[4] and the corresponding chapters in the 'multimedia' textbook.

[1] http://www.cs.cmu.edu/~christos/courses/826.F12/FOILS-pdf/320_DSP.pdf

[2] http://www.cs.ucr.edu/~eamonn/selected_publications.htm

[3] See http://www.cs.ucr.edu/~eamonn/SIGKDD_trillion.pdf

[4] See http://www.cs.cmu.edu/~christos/courses/826.F12/FOILS-pdf/310_multimediaDB.pdf

(a) PCA/SVD
(b) Multidimensional Scaling
(c) FastMap

You may use core functions, needed for the visualization, which already exist in Matlab (e.g. SVD). However, you are not encouraged to use some black-box visualization toolbox (i.e. load the data, hit run, and get the plot), since the goal of this task is to understand the mechanics behind visualization of high dimensional data.

Besides your implementation, you should also turn in a plot with the visualization produced by each method. What are the differences between the three methods? Do these differences stand out in our case?

5. *Anomaly Detection* Download the following piece of the insect data: `http://www.cs.cmu.edu/~epapalex/15826F13/data/Anomalies.zip`. Are there any outliers in the data? Implement and apply the LOF algorithm. For any outliers that you find, provide a justification as to why you believe they were selected by the algorithm (listening to the .wav files might help).

4 Logistics: What to turn in?/Group members

The deliverables and the grading scheme are in the project guidelines `http://www.cs.cmu.edu/~christos/courses/826.F13/proj.html`

1. For Phase 1, you are expected to implement tasks 1 and 2(a,b,c) (in addition to your literature survey).
2. For Phase 2, you are expected to have about half of the remaining tasks done;
3. for Phase 3, all of the above tasks (including your own distance function).

Reminders:

- As mentioned in the project guidelines, all projects will be done in **groups of two** (with exceptions only under special circumstances, after instructor's permission.)
- *Academic Attribution*: It is fine to use ideas, code, algorithms of another person or corporation, but *please cite your sources*.