Técnicas de Perceção de Redes
Network Awareness

Network Profiling and Entity Classification

Python references: *SciPy* – SciPy.org, http://www.scipy.org/

NumPy – NumPy Routines, https://docs.scipy.org/doc/numpy/reference/routines.html

matplotlib.pyplot - http://matplotlib.org/api/pyplot_api.html

scikit-learn, Machine Learning in Python - http://scikit-learn.org/stable/

Feature Extraction (time-independent)

File 'baseStats1.py' contains the base code to use in this section.

The Base data files 'YouTube.dat', 'Browsing.dat', 'Mining.dat' contain the download and upload data (byte counts per 1 second interval) for YouTube video playing, user web browsing, and one active crypto-coin miner data streams, respectively.

Note: The traffic data streams were captured with a single application running at a time, and capturing all traffic from and to a specific machine using a specific port: (TCP 443 for YouTube using automatic video playing at 720p), (TCP 443 and 80 for browsing multiple web pages with Firefox), and (TCP port 3357 for the miner).

1. Read data for all three files, and plot and analyze the download and upload traffic profiles over time.

2. Divide each data stream in (sequential) observation windows of 5 minutes (300 seconds), and divide (randomly) a set of observations for training and another for testing. Plot and analyze, separately, the train and test dataset for each application.

3. From the train observations, create a single data file with Time-independent Descriptive Statistics features and a vector with the known classification of each observation.

Note1: The best format is 2-D matrix, where each line contains the set of features for each observation. The classification vector should contain the known data classification (e.g., 0-YouTube, 1-Browsing, 2-Mining) of each observation, in the same order used in the features.

Note2: The descriptive statistical features (for upload and download) may be: mean, standard deviation, and percentiles for 75%, 90% and 95%.

4. Make 2-D log-log plots using different pairs of features, where each point will have a color dependent on the known classification. Try different feature combinations, and try to find find possible discriminators between applications.

Note: Example features pair: (mean upload, mean download).

Feature Extraction (time-dependent)

5. Using the data observations, extract (for each one) the download and upload silence duration and calculate its mean and variance as four additional features.

Make 2-D plots (or log-log plots) using different pairs of the new silence based features. Try different feature combinations, and try to find find possible discriminators between applications.

Note: you may consider a silence period when less than 256 bytes (~4 small packets) are transmitted.

6. Calculate and plot the scalogram for one YouTube, Browsing, and Mining observation (download traffic). Observe the differences and explain how these relate with inherent service characteristics.

7. Using the data observations, extract (for each one) the download and upload data Wavelet scalogram and use the "Energy" at predefined scales (e.g., [2,4,8,16,32,64,128,256]) as additional features.

Make 2-D plots using different pairs of the new scalogram based features. Try different feature combinations, and try to find find possible discriminators between applications.

Feature Reduction

8. Considering the features (for upload and download): mean, variance, median, variance, skewness, kurtosis, and percentiles for 25%, 50%, 75%, 90% and 95%; use Principal Components Analysis (PCA) to reduce this features to **3 main components**. Make 2-D plots using the 2 of the new features outputted by PCA, where each point will have a color dependent on the known classification. Try to find find possible discriminators between applications.

9. Considering only the features extracted from the Wavelet scalogram (for upload and download); use Principal Components Analysis (PCA) to reduce this features to **3 main components**. Make 2-D plots using the 2 of the new features outputted by PCA, where each point will have a color dependent on the known classification. Try to find find possible discriminators between applications.

10. Considering now all descriptive statistical features inferred;use Principal Components Analysis (PCA) to reduce this features to **3 main components**. Make 2-D plots using the 2 of the new features outputted by PCA, where each point will have a color dependent on the known classification. Try to find find possible discriminators between applications.

Classification (Statistical Analysis)

11. Using all inferred features, calculate the centroid of each class. Using the test dataset calculate for each observation the features (the same used above) and calculate the Euclidean distance to each centroid. Classify each observation pf the test dataset based on the lowest distance to the respective class centroid.

12. Considering only the new features outputted by applying the PCA to all features (point 10 above), infer the multivariate normal (Gaussian) distribution parameters for each class. Classify each observation of the test dataset based on the highest probability returned by the respective distributions of each class.

Classification (Machine Learning)

13. Using the K-means method perform a cluster analysis with the **normalized** PCA reduced features assuming 2, 3 or 4 clusters. Analyze the results and propose classifier for the test dataset.

Note: To infer the **normalized** PCA reduced features, apply the PCA to all features normalized to mean zero, variance one.

14. Repeat the previous point using know the DBSCAN method of clustering. Test multiple values for the maximum distance parameter (eps).

15. Using support vector machines with linear, RBF, polynomial (degree 2) kernels and LinearSVM develop a simple classifier for the test dataset (using all inferred features **normalized**).

16. Repeat the previous point using only the **normalized** reduced features (PCA).

17. Using Neural Networks develop a simple classifier for the test dataset traffic that outputs the class identifier (from all inferred features or PCA reduced features, both **normalized**). Start with a single 100 nodes hidden layer, and test other hidden layer sizes.

18. Improve the previous classifier by using a neural network for each class that outputs 1 or 0.