Concept Change Detection – Write-Up

Bin Gao, Bin Yan

# Part I Numeric change detection:

Numeric change detection uses the sliding window size of 50, and increments the sliding window by one number at a time. The algorithm will detect the first position that falls out of the calculated confidence interval. If the detected change persists for at least window size, it’s confirmed to be a change and outputted. This is based on the assumption that there is only one change in a test file.

If the file doesn’t have enough numbers after the first detected position to confirm that change has persisted for at least window size, the program will check if the change has persisted for more than half of the window size at the end of execution. If so, the suspected change is outputted, since the test file doesn’t have enough data to overthrow the suspected change, and it’s based on the assumption that there is only one change in a test file.

Two confidence intervals are calculated in the algorithm, the mean confidence interval based on the t-test and the variance confidence interval based on the “Bayesian perspective”, <http://hdl.handle.net/1877/438>. Since mean change will always cause variance change, but variance change may not cause mean change. At any moment of the execution, if the program finds both potential mean change and potential variance change, mean detection will dominate the execution, but variance detection will still be running in the background. It means if the program finds both mean change and variance change at the end of execution, mean change position will be outputted.

Because large variance has a large confidence interval, and small variance has a small confidence interval, detecting variance going up is usually faster than detecting variance going down. It means detection of variance going down has some delays. Therefore, when variance moving down is detected, the program uses a backtrace procedure to pinpoint the more accurate position of variance moving down. The backtrace procedure basically use the same algorithm to detect the variance moving up in the reverse direction, since detecting variance moving up is more accurate.

# Categorical change detection:

test

test