NegSelReport

Name: Kush Kapur User-ID: ccbd24

Two-letter code for your chosen negative selection algorithm:

***Algorithm Implemented**: V detector*
***Best Parameters**: **expected coverage rate for non self** (**c_0**): 0.999, **Expected coverage rate for Self(c_1):** 0.9999, **Self-radius**: 0.0081**, Number of detectors** = 200*
***Best result**: 95.3% detection rate and 0.4% false alarm rate (Training for 30 seconds)*
***Hyperparameter tuning: C_0 and C_1 (Self-radius:0.01, max detector:200)** Tuning parameter which control termination of algorithm, being the **c_0** and **c_1**. Setting **c_0** means judging the self/non-self-divide, and as our aim is to maximise non-self-detection, I chose **c_0** as 0.98, **c_1** was 0.99. This is because I aimed to fit the maximum number of detectors available which I chose to be a constant 200 for this parameter. What I realised after toggling c_0 from 0.9 to 0.999 is that when **c_0** = 0.95 although the false alarm rate is 0.2%, but the detection rate is 85.8% because it only uses 64 out of the max 200 detectors available, which limits the detection rate of the algorithm. Whereas when increasing c_0 = 0.999 I found the detection rate to significantly increase to 95.8%, with a slight increase in false detection rate of 0.6%, but more significantly all 200 detectors available were used, which resulted in the detriment of 0.4% in false detection rate but an increase of almost 10% in the detection rate. Similarly with c_1 referring to the expected coverage of self if set too low would result in the algorithm terminating too early, for example when c_1 = 0.96 the detection rate as 58.6% this is because if c_1 is too low it forces termination by assuming that its hard to squeeze a detector into the non-self-space. Therefore, I realised to maximise c_0 and c_1 to both 0.99 to prevent termination, and utilising max available detectors, which obviously results in increase in false alarm rate, but poses significant improvements in detection rate.*
Self-radius(C_0:0.99, C_1:0.9999, Max detectors:200)
Secondly, I was curious regarding the self-radius parameter for V detector, and I increased it in from 0.01 to 0.005 to find that there was a detection rate vs false-alarm trade-off. As I decreased the self-radius the false alarm rate increased, for example at self-radius = 0.004 the false alarm rate was 4.8%, and the detection rate was 93.9%. Decreasing the self-radius to a certain extent results in a decrease of false alarm rate, however at a point of self-radius < 0.005 the false alarm rate starts increasing. Although decreasing the self-radius whilst keeping all other parameters constant also results in a more efficient algorithm, this is because a lower self-radius means that their a is a lower probability of v detector assuming a s from self does not lie in the extremity of self. Therefore, it results in a higher false alarm rate if decreased below 0.007. Although to maximise detection rate and minimise false alarm rate, the self-radius was found to be in the range of 0.0085 and 0.007, for example at self-radius =0.007, and number of detectors = 200 the detection rate was 94.8% and false alarm rate was 0.6% with a 14.8 second training time. Whereas by changing the self-radius and keeping all other variables constant to 0.0081 I found a slight increase in the detection rate of 95.3% and false detection rate of 0.4%. However, the training time for the second experiment was 30 seconds, double the training time for self-radius as 0.007.
***Detector Number (C_0:0.99, C_1:0.9999, Self-radius:0.0081):** Finally, for choosing number of detectors I increased detector numbers from 170 to 220. At 170 I found the detection rate to be 94.4% and 0.4% false alarm rate, at 200 detection rate was 95.3% and 0.6% false alarm rate, but at 220 even though false alarm rate increased to 95.6% but also false alarm rate 1.6% because too many detectors are resulting in higher false alarm rate there's an excess number of detectors*