Image Classification using Random Forests and Ferns

Some technical update(s) in the ML-related component:

In my paper the technique used for classification is Random forest and random fern. As we observe Performance is 32.4% for D=10, 36.6% for D=15 and 43.5% for D=20. When using ferns performances are 30.3%, 35.5% and 42.6% for S=10, 15 and 20 respectively. In ferns the training time increases linearly with the number of tests S, while for random forests it increases exponentially with the depth D. Increasing the depth increases performance however it also increases the memory. Also increasing the number of trees also increases the performance but it also increases the memory to store a greater number of trees. But we have to minimize the memory and maximize the performance. As we know for large number of trees performance is highest and also there is low chance of overfitting in random forest. Also, for larger depth we have better performance. To do so we would take a larger value of tress for forest and iterate in either forward or in backward direction. Also, for best number of trees we take a depth and also iterate the depth in either forward and backward direction. We would stop at a point when we get minimum number of trees and also minimum depth. For example, let us assume we would take 300 trees and depth is 50 at first, we would increase number of trees and if performance does not increase, we would iterate in reverse direction. From that we get optimum number of trees for highest performance. Now for our obtained number of trees we would iterate for depth size. From that we will get minimum number of trees and also minimum size of depth having largest performance. Also, by using this technique we can reduce memory requirement as minimum number trees are stored.

Also, there may be chance of improvement in performance if we choose arbitary shape of region of interest instead of choosing rectangular region of interest. As in arbitrary shape we would selecting only object instance so there will be high chance of visual similarity. So, from this image can be matched in better way and hence the performance may increase.

In this paper the classifier detects the image and select this based on shape and appearance instead of this we can do an alternative way we take a forest let say it contains 300 trees and we take both the descriptor shape and appearance and classify the image like choosing 150 tree and classify the image based on shape and for rest 150 trees classify based on appearance and merge them to find the final classification result. So, this method reduces the time to select each image first and decide whether it can be classify based on shape or appearance. But there may be chance of decrease in performance of the model.