Bayesian Network-based Classifiers for Face Detection

Yan Zhao, Minjun Huang

Problem Statement

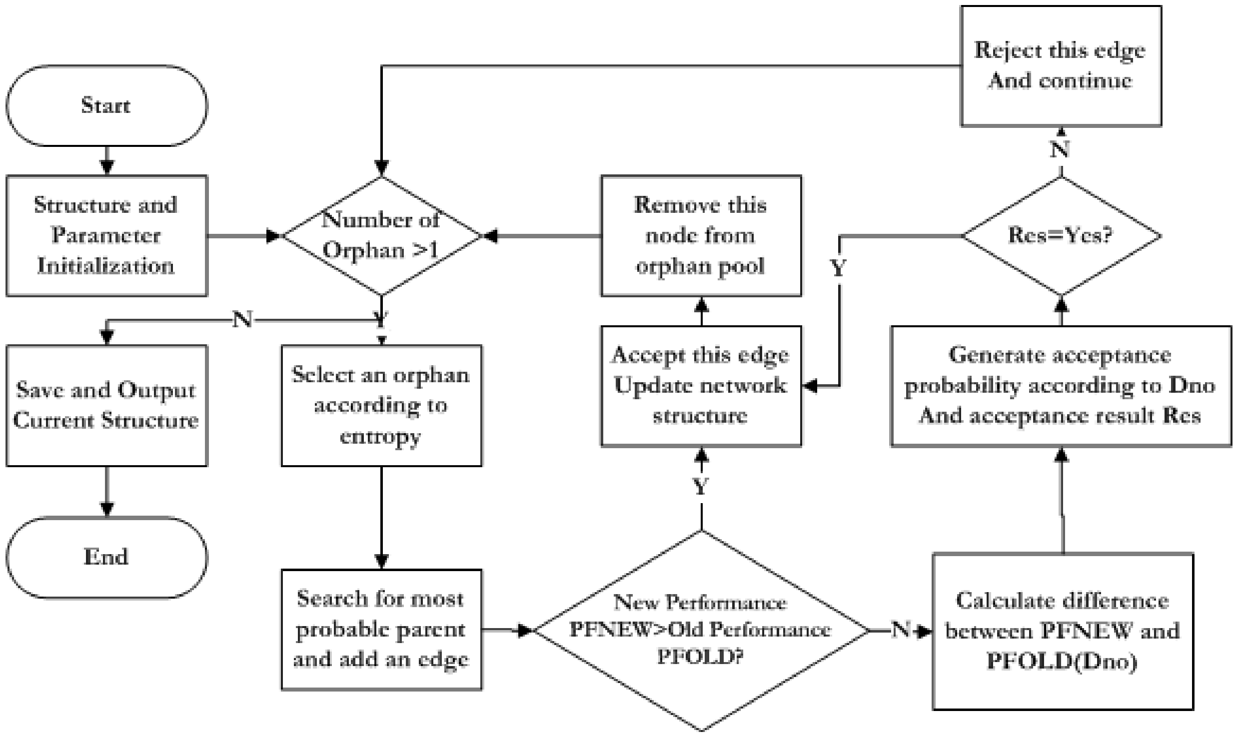
Face detection is an important prior step of face recognition. Accurate detection of human faces in any arbitrary scene is the most important prerequisite of face recognition. But face detection is easily subject to scales, illuminations, colors, orientations and head poses in an image. Also detection of for profile faces or faces with perturbations such as shadows, glasses and beard remains a challenging task. Currently, major methods of face detection are based on maximum likelihood learning method, which is computationally efficient but might converge to suboptimal due to mismatch between its optimization criteria (parameter likelihood) and its actual goal of classification (label prediction accuracy). Therefore, our research concentrates on faces that are hard to detect with current algorithms [1]. Bayesian Network (BN) can reflect both the conditional independencies and dependencies among features of human face and therefore is more efficient and accurate [2]. Some existing algorithms [3] are able to learn BN from data in an acceptable time and space complexity. Therefore, we choose to base our face detection classifier on learning BN.

Methods

The first step of our proposed method is to learn a single BN classifier by Augmented Bayesian Network-Simulated Annealing (ABN-SA) algorithm. Then we learn the weight of each single classifier by AdaBoost algorithm, in which the subsequent classifier learning is tweaked in favor of those instances misclassified by previous classifier. Finally, we get a linear combination of BN classifiers based on their weights learned from AdaBoost. We anticipate that the final classifiers obtained from AdaBoost will yield a better training result in hyper parameters selection, which will ultimately improve the overall model performance in terms of lowering error (misclassification) rate.

1) ABN-SA

The following flow chart represents our ABN-SA structure learning for a single classifier.



We applied the hill-climbing algorithm in BN structure learning [4]. The steps are described as follows:

⓪ Initialize network (G_1) to Naive Bayes (G_{NB})

① Evaluate the current classifier through misclassification rate by 10-fold cross validation on training set

② Consider adding every legal arc from Xi to Xj to the current classifier (Gs), where Xj belongs to Orphan node sets and Xi is not equal to Xj

③ Compute mutual conditional entropy (CCE, which is defined as $CE(X,Y|Z) = \sum_z P(z) \sum_{x,y} P(x,y|z) \log(\frac{P(x,y|z)}{P(x|z).P(y|z)})$) between Xj and Xi with each j. Add arc (j, i) that has Max CCE to current Gs. Remove Xi from Orphan set. Go to step ①

④ Repeat ①-③ until there is only one node in the Orphan set or no improvement can be made to Current Gs

2) AdaBoost Parameter Learning

We applied the AdaBoost algorithm in parameter learning in combining different BN classifiers [3]. The steps can be described as follows [5]:

⓪ Given a base structure G and the training data, initialize training data weights (averaged)

① Repeat for k =1, 2, … : Given G, θ_k is learning through ML on the weighted data D_k, compute the weighted error $Err_k = E_\omega[1_{x_c \neq f\theta_k(x_a)}]$, $\beta_k = 0.5log\frac{1-Err_k}{Err_k}$, where $f\theta_k(x_a)$ is classifier outcome on selecting label x_c given x_a during boosting iteration k, x_c is the true label of x_a, and β_k is the corresponding weight. Update weights $\omega_i = \omega_i \exp\{-\beta_k x_c^i f(x_a^i \,|\theta_k, G_k)\}$ and normalize.

② Ensemble output: sign $\sum_k \beta_k f(x_a \,|\theta_k, G_k)$

Dataset

The face data in our dataset is obtained from the FDDB database [6]. It contains 1097 faces (including frontal face 498, left 45 degree 166, left 90 degree 131, right 45 degree 172, right 90 degree 130) We also obtained 6325 nonfaces images from online database. The whole dataset is then split equally into 2 subsets for training and testing.

Result and Discussion

Four experiments have been conducted for comparison studies in order to access our BN classifiers.

1) Comparison of Naive Bayes and ABN-SA classifier

ABN-SA classifier refers to the first single classifier output by the ABN-SA algorithm, which means that it is generated without Adaboost parameter learning process. We compare its performance with Naive Bayes classifier through comparing the cross validation error rate on different image sizes (by resizing), from 4×4 to 20×20 respectively.

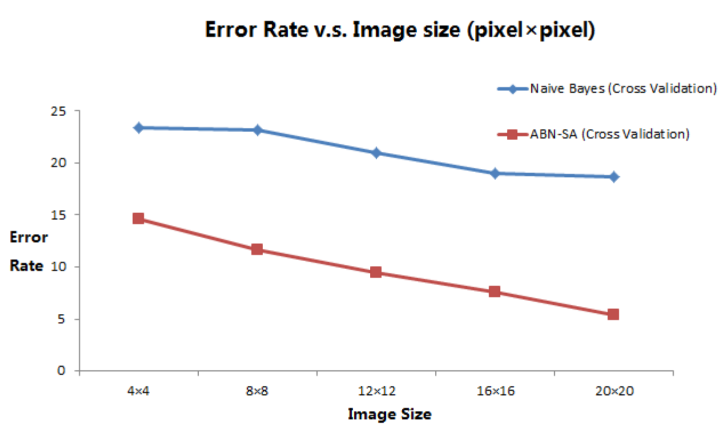


Fig 1 Comparison of error rate over different image size

By doing cross validation, for one thing, we can provide some information about the performance in actual testing. More importantly, we would know from the decrease of error rate that adding extra edges to the NB network is actually improving its performance, which is shown in Fig 1 that ABN-SA is improving the performance by at least about 10%.

From Fig 1, we can also see that larger image size would yield a larger network size, which would eventually yield higher accuracy. However, as a trade-off, it will also lead to a longer training time, which is what we observed in our training.

2) Comparison of Naive Bayes, ABN-SA and AdaBoost-ABN in testing

Next, we compare the testing error rate of the Naive Bayes, ABN-SA and AdaBoost-ABN in the testset.

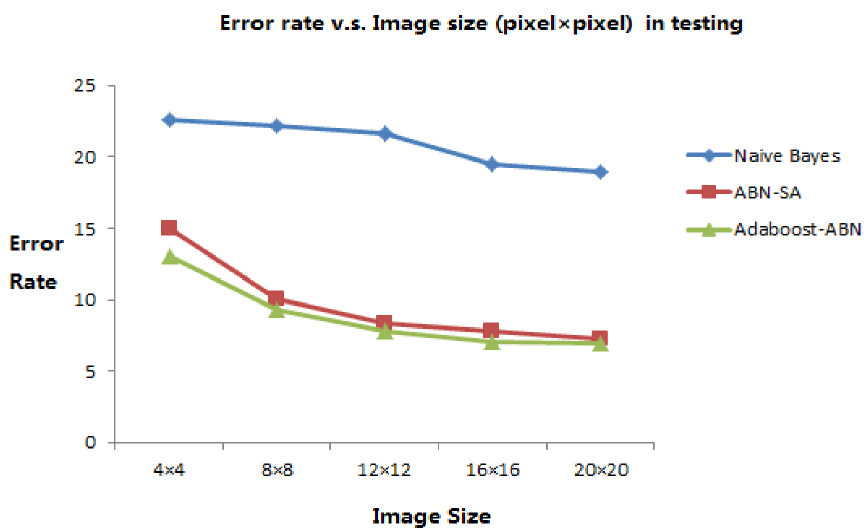


Fig 2 Comparison of error rate in testing

We can see from Fig 2 that AdaBoost is actually improving the performance. However, the improvement is in a small scale. There are several possible reasons for this. First, we might need more samples in the training set to cover the feature space as much as possible. Second, there might be some improper reallocation of sample weights in the AdaBoost. Third, the single classifier we got from ABN-SA is not a weaker learner, which makes the effect of including all the weak learners in AdaBoost very marginal.

3) Comparison of network complexity in different image size

Since the number of weak leaners is an indicator of network complexity, we compare the number of weak leaners in different image sizes and see that it's not monotonously increasing as the image size increases. This implies that we cannot increase the convergence rate of AdaBoost by simply increasing image size (network complexity).

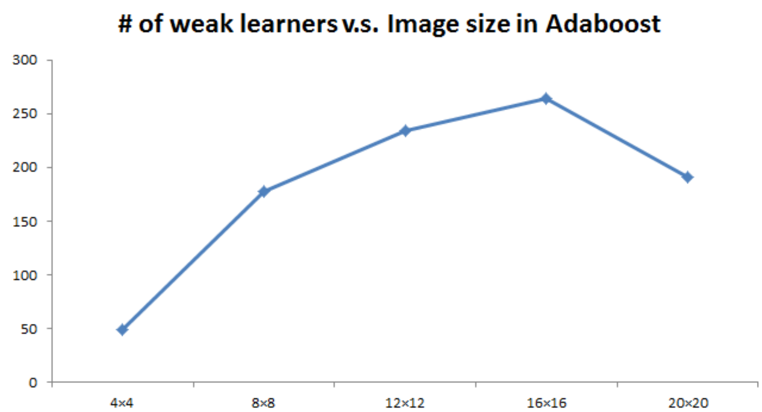


Fig 3 Comparison of network complexity in different image size

4) Comparison of error rate in different face subclasses

Finally, we want to examine the misclassification rate in different subclasses (Fig 4). As we can expect, larger image size would yield a low error rate among all subclasses. But what should we pay attention to is that the misclassification rates in face groups are much higher than the Nonface group, which mean that our relatively low overall error rate is largely contributed by the low nonface error weight. We should improve the ability of our BN to better classify the face subclasses.

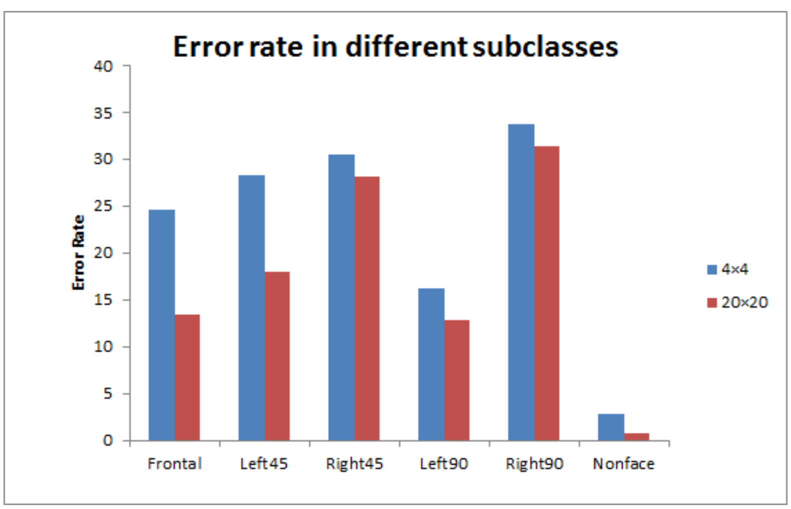


Fig 4 Misclassification rate in different subclasses

References

[1] Pham, T. V.; Worring, M. & Smeulders, A. W. Face detection by aggregated Bayesian network classifiers. *Pattern Recognition Letters*, **2002**, *23*, 451 – 461

[2] Jing, Y.; Pavlović, V. & Rehg, J. Boosted Bayesian network classifiers. *Machine Learning, Springer US*, **2008**, *73*, 155-184

[3] Neapolitan, R. E. Learning Bayesian networks. *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM*, **2007**, 5:1-5:1

[4] Keogh, E. J. & Pazzani, M. J. Learning Augmented Bayesian Classifiers: A Comparison of Distribution-based and Classification-based Approaches, **1999**

[5] Friedman, Nir and Geiger, Dan and Goldszmidt, Moises. Bayesian Network Classifiers. *Machine Learning*, **1997**, 29, 131 – 163

[6] http://vis-www.cs.umass.edu/fddb/