

基于动态权重裁剪的快速 Adaboost 训练算法

贾慧星^{1),2)} 章毓晋^{1),2)}

¹⁾(清华大学信息科学与技术国家实验室 北京 100084)

²⁾(清华大学电子工程系 北京 100084)

摘 要 提出了基于动态权重裁剪的快速 Adaboost 训练算法,当训练数据集较大时,可以大大提高训练速度. 基于动态权重裁剪的 Adaboost 训练算法在每次迭代过程中舍去权重较小的大多数样本,保留权重较大的少数样本进行训练,迭代完成后检查这个利用少量样本训练得到的弱分类器在所有样本上的分类性能,如果错误率大于 0.5,则扩大样本的数量重新训练本次迭代的弱分类器. 由于在大多数迭代过程中,只利用了少量样本进行弱分类器的训练,从而提高了整个算法的训练速度.

关键词 Adaboost; 动态权重裁剪; 机器学习

中图法分类号 TP391

DOI 号: 10.3724/SP.J.1016.2009.00336

Fast Adaboost Training Algorithm by Dynamic Weight Trimming

JIA Hui-Xing^{1),2)} ZHANG Yu-Jin^{1),2)}

¹⁾(National Laboratory for Information Science and Technology, Tsinghua University, Beijing 100084)

²⁾(Department of Electronic Engineering, Tsinghua University, Beijing 100084)

Abstract This paper presents a novel fast Adaboost training algorithm by dynamic weight trimming, which increases the training speed greatly when dealing with large datasets. At each iteration, the algorithm discards most of the samples with small weight and keeps only the samples with large weight to train the weak classifier. Then it checks the performance of the weak classifier on all the samples, if the weighted error is above 0.5, it will increase the number of training samples and retrain the weak classifier. During training, only a small portion of the samples are used to train the weak classifier, so the speed is increased greatly.

Keywords Adaboost; dynamic weight trimming; machine learning

1 引 言

Boosting 是一种重要的机器学习算法,其基本思想来源于 PAC (Probably Approximately Correct) 学习模型^[1]. 在 PAC 学习模型下,如果一个学习算法能够以足够高的准确率和足够高的置信度来解决一个分类问题,并且该学习算法需要的运行时间是

准确率、置信度、样本数目、特征数目的多项式级数,那么称这个分类问题是强可学习的,该学习算法称为强学习算法;而如果学习算法分类的准确率仅比随机猜测略好,那么该学习算法称为弱学习算法. Kearns 等^[2]证明了当训练样本充分时,能够将仅比随机猜测略好的弱学习算法提升为准确率足够强的强学习算法,而不必直接去寻找通常很难得到的强学习算法. 1990 年, Schapire^[3]通过一个构造性的方

法对该问题进行了肯定的证明,这就是最初的 Boosting 算法. Freund^[4]在此基础上进行了改进,提出了一个效率更高的 Boosting 算法. 这两个算法有一个致命的缺陷,即需要事先给定弱学习算法错误率的上界,这在实际问题中很难做到,使得 Boosting 的应用受到了限制. 不久, Freund 和 Schapire^[5]提出了 Adaboost 算法(Adaptive Boosting),突破了早期 Boosting 算法的一些限制,结构简单,实用性强,逐渐成为机器学习领域最为重要和常见的算法之一. Adaboost 算法在许多分类问题中显示了比较成功的应用,例如文本分类^[6]、人脸检测^[7]等.

Adaboost 算法的一个缺陷是,当训练样本和特征较多的时候,训练时间过长. 因为在每次迭代后,各个样本的权重都要发生变化,要重新训练最佳的弱分类器. 例如, Viola 为了训练一个人脸检测器,需要数周的时间^[7]. Adaboost 算法训练时间过长的问 题,限制了可以使用的样本和特征的维数,影响最终训练得到的分类器的性能,所以一些学者提出了许多方法用来降低 Adaboost 的训练时间. 现有的降低 Adaboost 训练时间的方法有两种,第一是对特征进行处理,第二是对样本进行处理. Wu 等人^[8-9]针对特征提出了两种快速训练方法:前向特征选择(Forward Feature Selection, FFS)算法和 Faster Adaboost 算法,其思路都是将和特征有关的每次迭代共享的计算放到迭代过程之前,通过共享的方式提高训练速度. Domingo 等人^[10]提出的 Madaboost 算法和 Bradley 等人^[11]提出的 Filter-Boost 算法通过样本采样的方式来提高训练速度,可以有效地处理大型数据库和串行数据. Friedman 等^[12]提出了一个基于权重裁剪的快速训练方法,即在每次迭代过程中,只保留权重较大的样本进行弱

分类器的训练,随着迭代次数的增加,只有很少一部分分类错误的样本的权值较大,保留下来进行训练,从而大大提高了训练速度,该方法简单易用,在实际问题中有广泛的应用.

Frideman 提出的权重裁剪方法在每次迭代过程中,裁剪系数不变,我们称之为静态的权重裁剪方法(Static Weight Trimming Adaboost, SWTAdaboost). 该方法的一个问题是,当裁剪系数较大时,由一小部分样本训练得到的弱分类器在全部样本上的加权错误率容易大于 0.5,使得算法过早地中止,达不到原始 Adaboost 算法可以达到的分类性能. 本文对 Friedman 的方法进行了改进,提出了动态权重裁剪方法(Dynamic Weight Trimming Adaboost, DWTAdaboost),即对由少量样本训练得到的弱分类器在整个样本集合的性能进行检验,如果加权错误率小于 0.5 则保留该弱分类器,否则扩大训练样本集合,继续训练,这个过程不断重复下去,直到找到加权错误率小于 0.5 的弱分类器或者用掉全部样本为止. 与 SWTAdaboost 相比, DWTAdaboost 在提高训练速度的同时,保留了原始 Adaboost 算法的分类能力.

本文第 2 节介绍经典的 Adaboost 算法;第 3 节介绍 Friedman 提出的 SWTAdaboost 算法及其存在的问题;第 4 节介绍我们提出的 DWTAdaboost 及其优点;第 5 节给出这 3 个算法在一个实际数据集上的测试结果;第 6 节总结全文.

2 Adaboost 算法

Adaboost 算法的流程图如图 1 所示,算法的基本思想如下:首先给定训练集(步 1)和一种弱学习

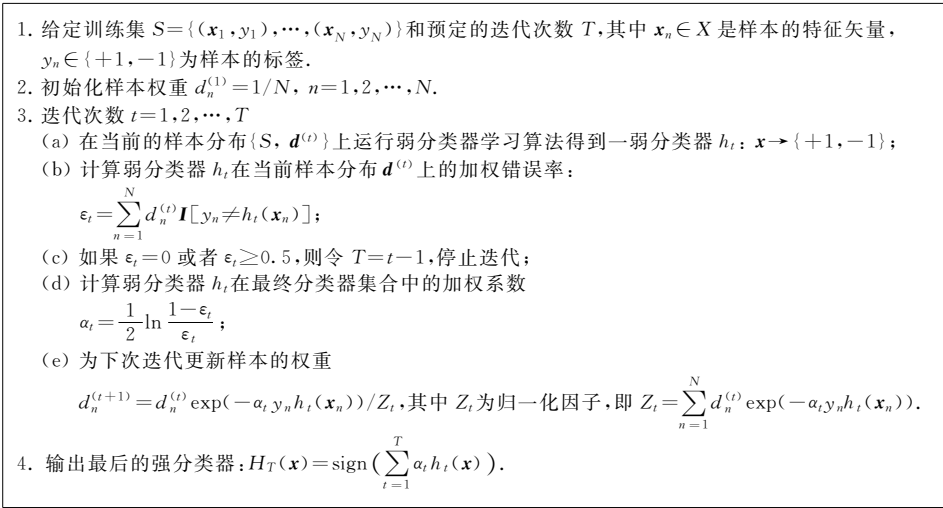


图 1 Adaboost 算法流程^[5]

算法, 然后对每个训练样本赋予一个初始权重(步 2), 利用样本的权重分布使用弱学习算法迭代地训练弱分类器(步 3(a)). 在每次迭代中的弱分类器训练完成后, 根据分类结果对所有的权重进行调整(步 3(e)), 提高分类错误的样本的权重, 降低分类正确的样本的权重, 使得下一次迭代更加关注那些分类错误的样本. 最后将由此得到的若干个弱分类器按照其在训练集上的错误率加权组合成一个强分类器(步 4), 其中错误率越小的样本对应的权重越大. Freund 和 Schapire^[5] 证明了如果 Adaboost 算法每次得到的弱分类器在训练集上的加权错误率严格小于 1/2, 上述算法在训练集上的错误率以指数的速度收敛于 0.

上述训练过程中, 时间主要损耗在步 3(a) 即弱分类器的训练上, 当样本数目和特征数目都较大时上述弱分类器的训练极为耗时, 从而导致整个 Adaboost 算法的训练速度较慢. 例如 Viola 等人训练的正面人脸分类器, 正面人脸样本数目有 4916 个, 特征有 45396 个, 在一台 466MHz 的 CPU 上训练一个分类器的时间需要数周左右, 极为耗时. 即使现在的机器性能有了较大的提高, 要想针对不同的姿态训练相应的分类器需要的训练时间也几乎是不可以忍受的. 为了将 Adaboost 应用到大规模数据

的学习上, 必须加快 Adaboost 算法的训练速度, 最基本的做法包括减少每次训练弱分类器采用的样本或者特征的数目. 针对降低训练样本数目, Friedman^[12] 提出了 SWTAdaboost 算法, 大大提高了训练速度.

3 SWTAdaboost 算法

Adaboost 算法不断降低正确分类样本的权重, 不断增加错误样本的权重, 这样权重较大的样本离分类界面较近, 权重较小的样本距离分类界面较远, 如果仅仅用权重较大的样本进行训练, 不会对最终的分界面产生大的影响. 假设每次舍去的样本的权重之和为 β , 我们称之为裁剪系数. 那么样本权重小于阈值 $t(\beta)$ 的样本就会被舍去, $t(\beta)$ 的定义式如下:

$$\sum_{n=1}^N d_n \mathbf{I}[d_n < t(\beta)] = \beta \tag{1}$$

之所以称 Friedman 提出的算法为静态权重裁剪 Adaboost 算法(SWTAdaboost)是因为每次迭代过程中 β 的取值不变. 该算法仅仅对原始的 Adaboost 算法的步 3(a) 做了更改, 算法的流程步 3(a)~3(c) 如图 2 所示, 其余部分与图 1 相同.

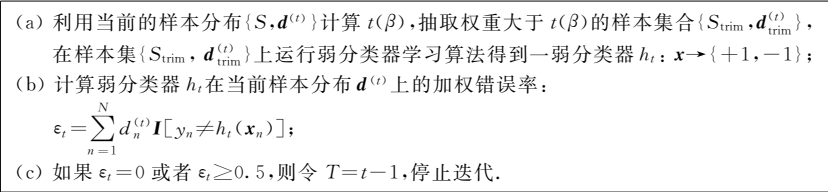


图 2 SWTAdaboost 算法流程

SWTAdaboost 算法的缺点是如果裁剪系数选择不当, 随着迭代次数的增加, 选取的样本越来越少, 同时得到的弱分类器也越来越弱, 导致得到的弱分类器在全部样本集合上的加权错误率超过 0.5, 从而使得算法过早中止, 达不到最佳的分类性能. 针对不同的训练问题, 合适的裁剪系数的选择需要反复实验来确定. 为了克服这个问题, 我们提出了动态的权重裁剪 Adaboost 算法(DWTAdaboost).

4 DWTAdaboost 算法

DWTAdaboost 算法的基本思想是在每次迭代开始的时候本次迭代的裁剪系数设为采用固定

的裁剪系数 β_0 , 然后利用抽取得到的权重较大的样本进行训练, 当训练得到的弱分类器在全部样本集合上的加权错误率超过 0.5 时, 减小本次迭代的裁剪系数, 即增大训练弱分类器用的样本数目从而有可能训练得到加权错误率小于 0.5 的弱分类器. 如果当采用全部样本进行训练时, 错误率仍然超过 0.5, 则停止迭代, 保留了原始 Adaboost 算法的分类能力. 算法的流程如图 3 所示, 对原始 Adaboost 算法的步 3(a)~3(c) 做了更改, 其余部分见图 1. DWTAdaboost 算法, 既采用了 SWTAdaboost 算法提高训练速度的性质, 同时保留了采用全部样本进行训练的原始 Adaboost 的分类能力, 避免采用 SWTAdaboost 算法裁剪系数选择不当的时候过早终止的问题.

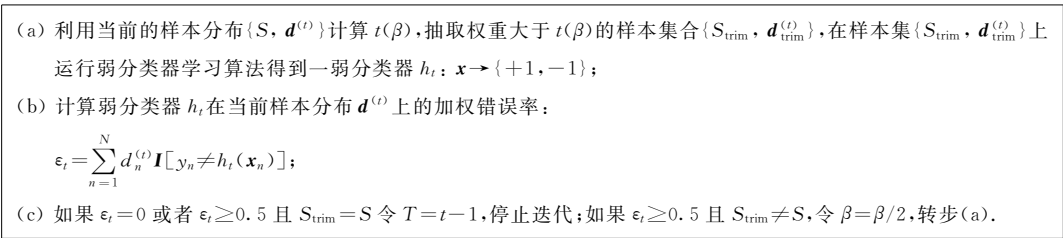


图 3 DWTAdaboost 算法流程

5 实验结果

为了比较 Adaboost、SWTAdaboost 和 DWTAdaboost 这 3 种算法, 我们在一个较大的数据集上做了测试. 测试数据集是在人脸检测领域中广泛使用的 MIT 人脸数据库^①. 数据集里面共有 6977 个训练样本, 24045 个测试样本, 每个样本包含一个 19×19 的人脸或者非人脸图像, 我们直接利用归一化的图像像素作为特征, 特征的维数总共是 361 维. 训练时, 弱分类器采用一个节点的树状分类器即 Stump 分类器. 测试所用的机器为 P4 3.0GHz CPU, 1GB 内存, 采用的 MATLAB 版本为 7.0.1.

实验 1. SWTAdaboost 和 DWTAdaboost 算法的训练加速性能. 令裁剪系数为 0.1, 3 种算法的实验结果如图 4、图 5 所示. 图 4 表明了 3 种算法在训练集和测试集上的错误率变化情况, 由该图可以看出, SWTAdaboost 和 DWTAdaboost 算法与原始的 Adaboost 算法相比, 在训练集和测试集合上的错误率基本相同. 图 5 表明了 SWTAdaboost 和 DWTAdaboost 算法在每次迭代中采用的训练样本的比例, 由该图可以看出经过 60 次迭代后, SWTAdaboost 算法和 DWTAdaboost 算法用的样本数目少于全部样本数目的 20%, 3 个算法在同一台机器上经过 200 次迭代所用的时间分别为 400s、75s、77s, 大约有 5 倍的加速. 由于裁剪系数的取值

较小, SWTAdaboost 算法无提前退出情况, 从而与 DWTAdaboost 算法的运行状况完全相同.

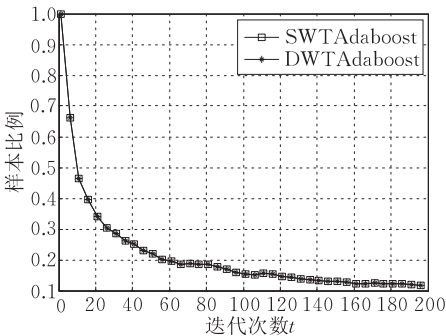


图 5 裁剪系数为 0.1 时 SWTAdaboost 与 DWTAdaboost 算法采用的样本比例

实验 2. 不同的裁剪系数对 SWTAdaboost 的影响. 裁剪系数取 0.2、0.3、0.4 时算法在训练集和测试集上的错误率如图 6 所示, 每次迭代使用的样本的比例变化情况如图 7 所示. 由这两个图可以看出, 当裁剪系数取 0.4 时, SWTAdaboost 算法迭代 6 步后训练得到的弱分类器在全部样本上的加权错误率大于 0.5, 从而导致算法过早地结束, 达不到最佳性能. 裁剪系数为 0.2 和 0.3 时 SWTAdaboost 迭代 200 次需要的时间分别为 58s、38s, 说明裁剪系数较大时算法的训练速度更快, 但同时我们必须控制裁剪系数的大小以防止出现提前退出的情况.

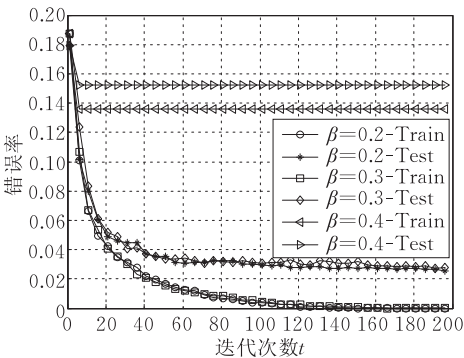


图 6 裁剪系数变化时 SWTAdaboost 算法在训练集和测试集上的错误率

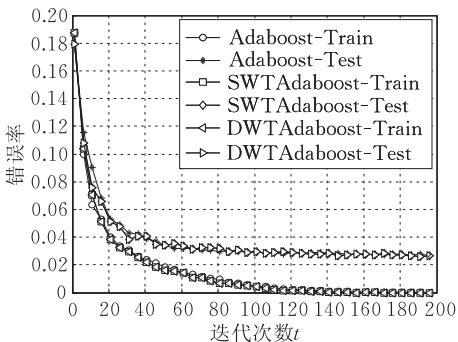


图 4 裁剪系数为 0.1 时 3 种算法在训练集和测试集上的错误率

① CBCL Face Database #1, MIT Center For Biological and Computation Learning, <http://www.ai.mit.edu/projects/cbcl>

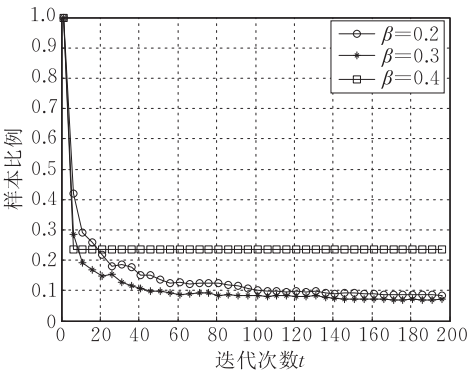


图 7 裁剪系数变化时 SWTAdaboost 算法每次迭代采用的样本比例

实验 3. 不同的初始裁剪系数对 DWTAdaboost 的影响. 初始裁剪系数取 0.2、0.3、0.4 时算法在训练集和测试集上的错误率如图 8 所示, 每次迭代使用的样本的比例变化情况如图 9 所示. 由这两个图可以看出, 当裁剪系数取 0.4 导致 DWTAdaboost 算法找到的弱分类器的加权错误率大于 0.5 时, DWTAdaboost 算法可以动态减小裁剪系数, 重新进行训练从而使得算法达到与采用所有样本时同样的性能. 当初始裁剪系数取 0.2、0.3、0.4 时, 算法

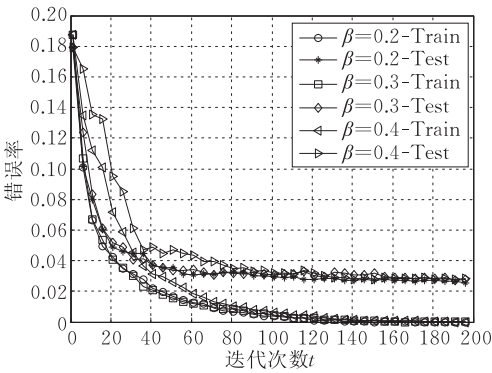


图 8 初始裁剪系数变化时 DWTAdaboost 算法在训练集和测试集上的错误率

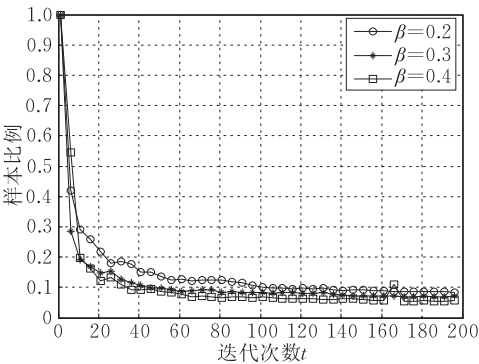


图 9 初始裁剪系数变化时 DWTAdaboost 算法每次迭代采用的样本比例

经过 200 次迭代的时间为 51s、38s、37s, 所以 DWTAdaboost 算法保证了训练速度的提高, 同时使得算法不过早终止.

实验 4. DWTAdaboost 算法使用不同的初始裁剪系数时训练时间和标准的 Adaboost 算法训练时间的比值, 如图 10 所示. 由该图可以看出, 初始裁剪系数越大, 训练速度越快, 到一定值之后, 由于某些迭代次数需要反复训练, 反而会增大训练时间, 所以在实际使用时仍然需要控制初始裁剪系数的大小.

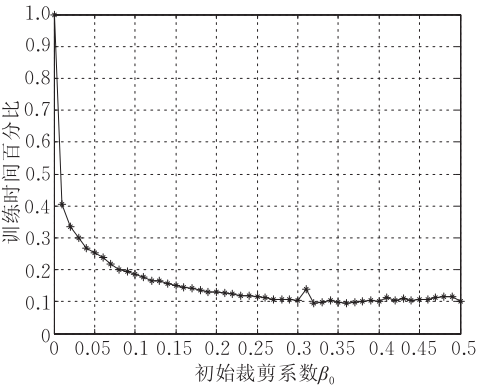


图 10 初始裁剪系数变化时 DWTAdaboost 算法每次训练所用时间和原始 Adaboost 算法训练所用时间的比值

由以上的实验结果可知, SWTAdaboost 算法和 DWTAdaboost 算法与原来的 Adaboost 算法相比, 在训练大规模数据时可以大大提高训练速度. SWTAdaboost 算法受裁剪系数的影响较大, 当裁剪系数过大会导致算法过早结束, 达不到最佳的性能; 对于不同的训练问题, 裁剪系数的影响也不同, 实际使用时需要反复进行实验, 选取最佳的裁剪系数, 这在一定程度上影响了该方法的应用. DWTAdaboost 算法保留了 SWTAdaboost 算法训练速度较快的优点, 当裁剪系数选择不当的时候可以自动调节裁剪系数, 达到与原始 Adaboost 算法基本相同的性能, 实际应用时更加方便.

6 结 论

本文提出了基于动态权重裁剪的 Adaboost 算法, 它继承了静态权重裁剪 Adaboost 算法较快的训练速度, 同时保证了算法可以达到与原始 Adaboost 算法基本相同的性能. 实验结果证明该算法在处理大规模数据时可以大大提高训练速度, 具有较大的实用价值.

参 考 文 献

- [1] Valiant L G. A theory of the learnable. *Communications of ACM*, 1984, 27(11): 1134-1142.
- [2] Kearns M, Valiant L G. Cryptographic limitations on learning Boolean formulae and finite automata. *Journal of the ACM*, 1994, 41(1): 67-95
- [3] Schapire R E. The strength of weak learnability. *Machine Learning*, 1990, 5(2): 197-227
- [4] Freund Y. Boosting a weak algorithm by majority. *Information and Computation*, 1995, 121(2): 256-285
- [5] Freund Y, Schapire R E. A decision-theoretic generation of online learning and an application to boosting. *Journal of Computer and System Science*, 1997, 55(1): 119-139
- [6] Schapire R E, Singer Y. BoosTexter: A boosting-based system for text categorization. *Machine Learning*, 2000, 39(2): 135-168
- [7] Viola P, Jones M. Robust real-time face detection. *International Journal of Computer Vision*, 2004, 57(2): 137-154

- [8] Wu J X, Rehg J M, Mullin D M. Learning a rare event detection cascade by direct feature selection//Sebastian Thrun, Lawrence K Saul, Bernhard Schölkopf. *Advances in Neural Information Processing Systems 16*. Cambridge, MA: MIT Press, 2004: 1523-1570
- [9] Wu J X, Brubaker S C, Mullin D M, Rehg J M. Fast asymmetric learning for cascade face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008, 30(3): 369-382
- [10] Domingo C, Gavaldà R, Watanabe O. Scaling up a boosted-based learner via adaptive sampling. *Data Mining and Knowledge Discovery*, 2002, 6(2): 131-152
- [11] Bradley J K, Schapire R. Filterboost: Regression and classification on large datasets//Platt J C, Koller D, Singer Y, Roweis S. *Advances in Neural Information Processing Systems 20*. Cambridge, MA: MIT Press, 2008: 185-192
- [12] Friedman J, Hastie T, Tibshirani R. Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 2000, 28(2): 337-407



JIA Hui-Xing, born in 1981, Ph. D. candidate. His research interests include pattern recognition, computer vision and image processing.

ZHANG Yu-Jin, born in 1954, professor, Ph. D. supervisor. His research interests are mainly in the area of image engineering that includes image processing, image analysis and image understanding, as well as their applications.

Background

Adaboost algorithm has demonstrates its success in face detection. Since the diversity of face pose, a lot of samples are needed to train a detector. Traditional training algorithm of Adaboost is very slow, costing about two weeks to train one detector. In order to train many detectors to evaluate the performance of different parameters, a faster training algorithm is needed.

There are two strategies to increase the training speed: feature selection and sample selection. The method in this paper belongs to the latter. Since the weight of each sample changes at each round of Adaboost, all weak classifiers are needed to retrain. The training time is determined by the number of sample, so it can be reduced by discarding some small weight samples. Based on the observation, Friedman

proposed static weight trimming Adaboost (SWTAdaboost), which increase the training speed greatly.

However, the parameter choose for SWTAdaboost is hard for different data sets. This paper presents dynamic weight trimming Adaboost (DWTAdaboost). At each iteration, dynamic weight trimming discards most of the samples with small weight and keeps only the samples with large weight to train the weak classifier. Then it checks the performance of the weak classifier on all the samples, if the weighted error is too high, it will increase the number of training samples and retrain the weak classifier. DWTAdaboost increases the training speed greatly compared to original Adaboost algorithm while keeps the classification power of the original Adaboost compared to SWTAdaboost.