Dirty Money: Feature selection using AdaBoost

J. van Turnhout (0312649) jturnhou@science.uva.nl,
N. Raiman (0336696) nraiman@science.uva.nl,
S. L. Pintea (6109969) s.l.pintea@student.uva.nl

February 3, 2010

Abstract

In the month January of 2010 a project on the classification of the fitness of money was proposed. During this month we have tested and implemented various techniques to handle the problem of money classification. The results from our experiment show promising development comparing to the current state of research.

In the below sections we have described the approaches that we have tried and the results obtained for each one of them.

1 Introduction

The goal of this project is to research several techniques that have a potential to distinguish between clean bills and dirty bills. In order to achieve this goal we had to think of what are the representative features of dirt that can be found on money bills and what techniques can be applied to model this.

The main techniques used in this project are: *PCA*, *Haar-like features*, *Convolution with predefined kernels*, ***edge-detection* and *intensity* distributions**. All of these techniques have been analysed on the image data using a machine learning technique called Adaptive Boosting, also refered to in literature as AdaBoost [2].

In section 2, related work on the project is discussed. In section 3, the implementation of the *AdaBoost* algorithm applied on the different techniques *Haar-like features*, *convolution with kernels*, *PCA*, *SVM*, *AdaBoost*, *edge detection* and *intensity* are discussed. We discuss our experiments and their results in section 4. Finally, we conclude in section 5 and propose some topics for future research.

what about telling first about what is done in previous projects. Then tell the idea of implementing a few techiniques to extract weak classifiers on regions of the bills and then combine them using Adaboost. So first we talk about previous work, then explain Adaboost and then talk about PCA, HaarConv and Intens&Edge

2 Background

In Europe a lot of different factories produce euro bills using different ink and paper. This makes the task of detecting dirty bills hard because looking at the global features will not work perfectly due to these small differences in printing. In Holland, on average bills return 2 to 3 times a year to the Dutch National Bank (DNB) with a total value of 1.1 bilion bills a year.

The current approach used in the DNB is to measure the reflection of the bill on a small area near the water mark. With this approach 30% of all bills are destroyed in order to destroy 95% of the dirty bills. In a first attempt to make this selection process more sophisticated global features (eigen-money) of clean and dirty bills are extracted using PCA analysis [3]. This approach improved on the results of the current method used by the DNB. The implementation was again improved by learning more local features of the water mark region (also refered to as white patch) of the bill [4]. This approach again improved the current results from DNB.

We think these results can be improved by learning weak classifeirs on

small areas (i.e. learn more local patterns) and combine those to a strong classifier using Adaboost. The methods used and experiments done are explained in the remainder of this deliverable

3 Methods

3.1 Adaboost

The *AdaBoost* algorithm will be used in combination with different techniques throughout this project, such as: *PCA*, *Haar-like features* or *edge* and *intensity* distributions over different regions.

AdaBoost is a machine learning technique which can be used in conjunction with various other learning algorithms. The idea is to have a (convex) set of weak classifiers (classifiers that perform at least better than random) and then minimize the total error over the training-set by finding the best classifier at each stage of the algorithm.

The theoretical basis of this algorithm is that given a set of models (or features), M, the algorithm will determine the subset of T models that are the best for distinguishing between the two classes (clean and dirty bills). Thus, *AdaBoost* will learn the most representative features of the two classes. The algorithm for determining the best models is shown in *Algorithm 1*. Another very important characteristic of this algorithm is the fact that it also specifies a method in which the models that were chosen as being the best, can be combined in order to give a strong classifier. The corresponding algorithm can be seen in *Algorithm 2*.

Note that the algorithm looks slightly different but is equivalent to the one described in [1].

Algorithm 1 AdaBoost learning features

function AdaBoostLearn(T, M, S)
T = nr. of hypothesis
M = Models
S = training-set, $\{(x_1, y_1), ...(x_n, y_n)\}$ with $x_i \epsilon X$ and $y_i \epsilon \{-1, 1\}$
$D_{1(i)} \leftarrow \frac{1}{n}$, with $i = 1, ..., n$
for $t = 1$ to T **do**
  $error_t \leftarrow 0$
  for $m \epsilon M$ **do**
    $h_j(x_i) \leftarrow predict(x_i)$ *%svm or gaussian distribution*
    $error_j \leftarrow \sum_{i=1}^{n} D_t(i)[y_i \neq h_j(x_i)]$
    if $error_j < error_t$ **then**
      $error_t \leftarrow error_j$
      $h_t \leftarrow h_j$
  $\alpha_t \leftarrow 0.5 \cdot \log \frac{1-error_t}{error_t}$
  for $i = 1$ to n **do**
    $D_{t+1}(i) \leftarrow \frac{D_t(i) \exp - \alpha_t \cdot y_i \cdot h_t(x_i)}{Z_t}$
return α, h

Algorithm 2 AdaBoost Prediction

function AdaBoostPredict(α, h, I)
α = weights
h = weak classifiers
I = image
p = prediction
for $t = 1$ to $length(\alpha)$ **do**
  $p \leftarrow p + \alpha_t h_t(I)$
return $sign(p)$

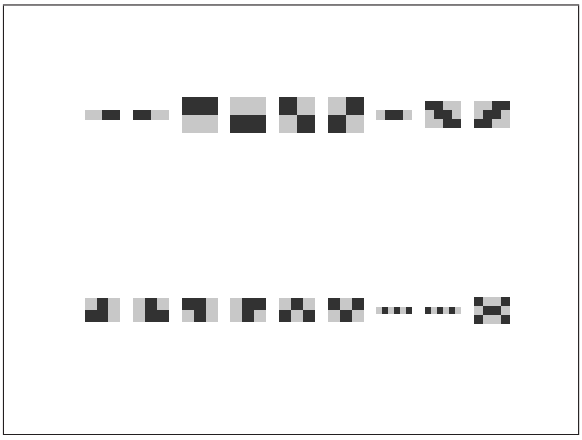

Figure 1: Patterns

3.2 Haar-like features and convolution with Haar-like kernels

The first idea that we have tried was to implement the *Viola & Jones* approach for object detection. The final cascade used in this paper was not implemented because the main idea of this cascade was not suitable for the purpose of this project. We have used the strong classifier computed in *AdaBoost* as the final output.

The first step of the algorithm is defining the patterns, that are mainly matrixes of different dimensions containing low and high values for intensity (-1 and 1) – Figure1 indicates a subset of the patterns used. The algorithm used in the

Figure 2:

$\sum_x \sum_y Image(y : y + h, x : x + w). * Pattern$

where: h – the heigh of the pattern
w – the width of the pattern

Viola & Jones article was designed to loop over all predefined patterns on each position in the image and convolve the specific region of the image with the patterns using the formula depicted in Figure2.

We have started by defining a large number of patterns of different sizes (≈100 patterns), and we have filtered out those that gave bad results when used in the *AdaBoost algorithm* for training the weak classifiers. The resulted values for each pattern and location in the image would, then, be used in *AdaBoost* to train an *SVM* classifier.

Taking into account the fact that the set of features generated by the algorithm for each pattern and each image, was extremely large, and the training took too much time. In order to solve this problem, we have decided to use just random locations at which to convolve the patterns with a region of the image (that would have the same size with the pattern). Figure 3 indicates what *AdaBoost* would choose as being the most representative 5 features for the front side and rear side of the bills.

The results obtained using *Haar features* were

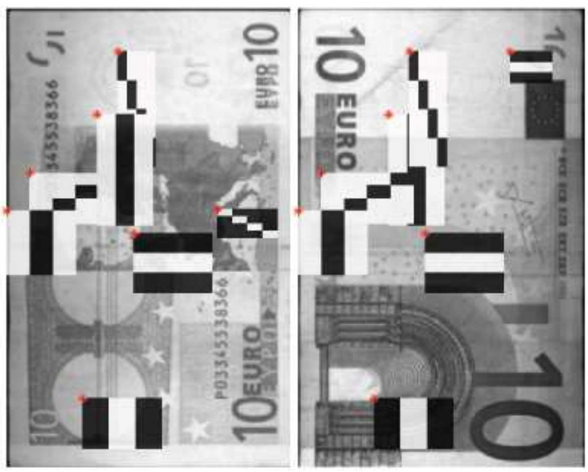

Figure 3: Haar Features for Front and Rear

not as good as we would have expected them to be. An explanation for this may be the fact that the patterns defined were not entirely able to model the dirt that can be found on the bills.

The second approach that we have tried was to define a set of patterns as before, and to segment each image into smaller regions that would be, then, convolved with the patterns.

We have tried using both no-overlapping segmentation of the images and 50% overlapping one. Due to the fact that many essential pixel values (like those that would indicate the presence of dirt) may happen to be positioned on the line that separates two neighboring regions, is possible that some information will be lost. Taking into account the number of regions generated for all images we can notice that there is a large amount of pixel values that might not be considered in the case of no-overlapping segmentation. As expected, the second method of dividing the images into regions gave better results compared to the first one.

The results obtained using this technique represented the input set of features for the *AdaBoost algorithm*. Figure 4 will help getting a better understanding of how the convolved regions of image and patterns would look like. In this image it is plotted the result obtained when convolving a bill with a simple pattern such as: **[1 -1;-1 1]** with an entire bill.

Thus, the resulted set of models would repre-

Figure 4: Convolution of an entire bill

sent the input features used in *AdaBoost*. For establishing the set of the best T features we have tried using both *SVM* and *Gaussian distribution*, and the results retrieved by the last one seemed slightly better than the ones obtained while using *SVM*. In the case in which *SVM* was used, a model was generated for each available feature. This model would give the best separation between the values corresponding to that feature for fit images and those for unfit images. In the case of the *Gaussian distribution*, the mean and covariance of features corresponding to fit images were computed, respectively the mean and covariance for the features corresponding to unfit images. In order to determine the predicted class of the input images we have tried two methods.

In the first method we would simply compute the *mean* and *covariance* for the values corresponding to fit, respectively unfit values corresponding to a specific feature. During testing, for each input value, we would compare the two numbers returned by *Gaussian density function* for fit, respectively unfit and we would choose the predicted class to be the maximum of the two.

The second method was using *Naive Bayes* in order to make use of the prior knowledge available. We know that in a real-life situation there are always more fit bills than unfit one and we would like to use this information to improve our classifiers. The predicted class was defined by using the *MAP* (maximum aposterior probability) estimation. The formula for computing the maximum aposterior probability for fit, respectively unfit class is given by the formulas depicted in Figure5.

It can be easily shown that the values of the parameters θ for a normal distribution (mean and covariance) that would maximize the probability of the classes are exactly the corresponding formulas indicated in Figure6.

The second technique used for determining the predicted class of the data given the parameters seemed to provides slightly better results in practice due to the fact that it also incorporates some prior knowledge of the data.

For this approach we have used two classifiers: one for the rear side and another for the front side of the bills and the predictions given by the two classifiers were combined into a third one using Naive Bayes.

In order to create the final model, we have tried two different methods:

- the first one was essentially just choosing the best model (the one that gives the minimum error) throughout all the repetitions and all the rounds in the validation;

- the second method was to use a system of voting – each time a feature was chosen by the *AdaBoost* algorithm, it would receive a vote. In the end the top T most voted features throughout all the validation rounds and all the repetitions would be chosen from the best model found. The corresponding weights would be generate by taking the mean of the features' α-weights computed in the *AdaBoost* algorithm;

As we would have expected, the second method for defining the best model, proved to give better results in this case so it was chosen to be applied.

Figure 5:

$$argmax_{\theta_{fit}} P(\theta_{fit} \mid X) = \frac{P(\theta_{fit}) * P(X|\theta_{fit})}{P(\theta_{fit}) * P(X|\theta_{fit}) + P(\theta_{unfit}) * P(X|\theta_{unfit})}$$

$$argmax_{\theta_{unfit}} P(\theta_{unfit} \mid X) = \frac{P(\theta_{unfit}) * P(X|\theta_{unfit})}{P(\theta_{fit}) * P(X|\theta_{fit}) + P(\theta_{unfit}) * P(X|\theta_{unfit})}$$

where:

- $P(\theta_{fit}), P(\theta_{unfit})$ – marginal probabilities of "fit" class, respectively "unfit" class
- $\theta_{fit}, \theta_{unfit}$ – the parameters of the classes (mean and covariance)
- $P(X \mid \theta_{fit}), P(X \mid \theta_{unfit})$ – the conditional probabilities of the two classes (representing normal distributions)

Figure 6:

$$\mu = \sum_i \frac{x_i}{|X|}$$
$$\sigma = \frac{(X-\mu)(X-\mu)^T}{|X|}$$

where X – the training data set
μ – the mean
σ – the covariance

3.3 PCA and SVM

In previous research, the use of SVM in combination with PCA already gave great error reduction on both the 5 and 10 euro bills. These results were conducted from performing PCA on the whitepatch area only, and propagate the projections in SVM. PCA on the whitepatch as well as the whole bill made it possible to reduce the data from a dimensionality equal to the number of pixels of the respective area to only a 30 components. This reduction ensures faster training and classification with SVM. The idea in this project is to identify more regions with similar discriminative features as the whitepatch. This is done by dividing the bill up in an arbitrary number of regions, and performing PCA for each of these regions in all the bills. The PCA segments then can be used to train SVM models for each segment. These SVM models in combination with their respective PCA segments in turn can be used as weak classifiers in AdaBoost.

3.4 Intensity and Edge distributions

Besides the more complex approaches we desided to implement also two more simple implementations, based on the intensities on the image and on the edges on the bill. The idea of using the intensity is based on the current approach used in the DNB (e.g. measureing the reflection of light on a small patch near the watermark **see section background**). The idea of using the edges is based on the assumption that dirty and old bills have significantly more wrinkels and folds then relative new bills. In the first simple implementation the **raw** intensity of the image of the bill is used. Herefor the bills are segmented in small regions (5 by 12) (**see figure ??**). Over these regions the average intensity is calculated. In this way it is possible to retrieve two gaussian distributions of the intensity of all the regions over all the images, one for clean bills and one for dirty bills. This provides a tool to compare the probability of a region on a bill belonging to a clean or dirty bill. In the edge approach the image is first filtered using the canny-edge detector. The canny edge filter works in 5 steps:

- Step 1: gaussian smoothing
- Step 2: extract the gradient of the image by convolving the image with 2 kernels, one to find the derivative in the X direction and one to find the derivative in the Y direction
- Step 3: determin the direction of the edge using results from step 2
- Step 4: apply nonmaximum suppression to find the real edge (selecting only the maximum edge points found)
- Step 5: finally the **hysterisis** is used to make the lines continues

The result of an image of a bill after applying canny edge filter can be seen in figure 7. After

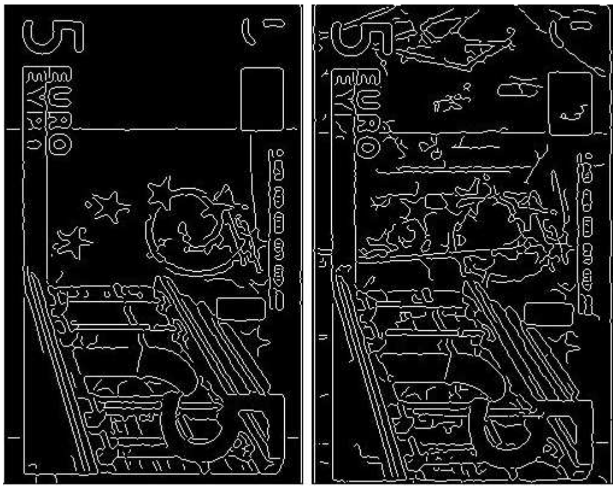

Figure 7: results canny edge filter. Left on a clean bill. Right on a dirty bill

applying the canny edge filter to the images the resulting images are segmented in small regions. Per region the sum of edge points is calculated. When doing this for all images we can again extract two gaussian distributions of edge points per region (one for clean and one for dirty bills).

4 Results

The available image set was split into a holdout set and the rest of the data was again split into a validation set and a part that was used for training the final model by *random sub-sampling validation* technique. The whole process of defining the validation set and applying random sub-sampling validation was repeated several times to ensure a correct estimation of the predictions. For each round of the validation, a model was trained using the *AdaBoost algorithm* described above. The obtained model was, then, tested by building the strong classifier and computing the corresponding values for: *true-positive estimation*, *true-negative estimation*, *false-positive estimation* and *false-negative estimation*.

4.1 Convolution Results

Due to the fact that the number of features generated for only 21 patterns and a segmentation for each image of 23 overlapping regions by 9 gives 4,347 features, the training part is relatively slow for this method. In order to create reliable model we have used 5 repetitions, 10 hypothesis (features to be chosen by AdaBoost), 10 trials in the random sampling validation method.

In Figure8 are indicated the most voted regions for the 10 Euro bills in *AdaBoost* throughout all trials and all repetitions. We can notice that for the front of the bills the regions selected are mainly positions on the water mark area, while for the rear side of the bills, there are some regions considered as being the most informative on the white patch, but also the area in the middle is selected as containing discriminative information.

Figure9 shows the most voted regions for the

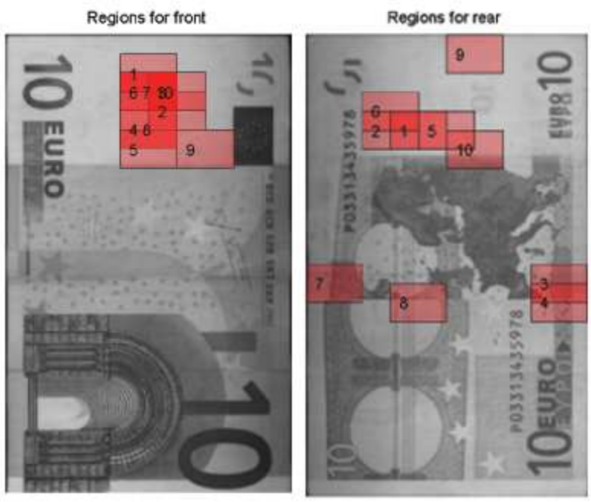

Figure 8: Best regions for 10 Euro bills

5 Euro bills in *AdaBoost* throughout all trials and all repetitions. For the 5 Euro bills the regions for front and rear that were chosen as being the most informative ones are again mainly around the water mark area, but unlike those for 10 Euro Bills, the regions for front side are the ones that are more spread throughout the image.

In order to determine the optimal number of

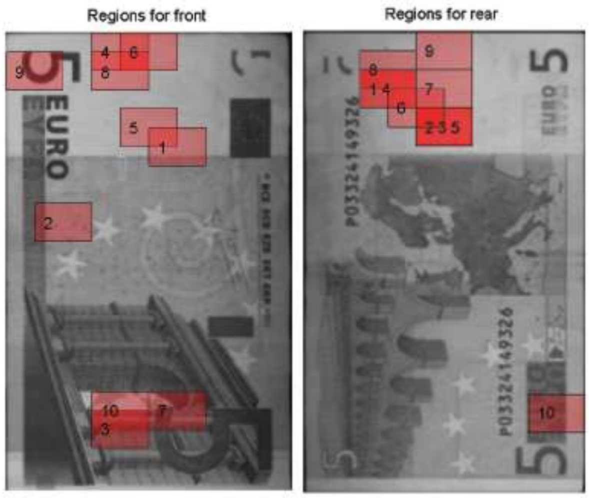

Figure 9: Best regions for 5 Euro bills

features that should be chosen in the end for the best model, we have plotted the error with respect to the number of features chosen. Figure10 and Figure11 show how the error evolves depending on the number of models for the 10 Euro bills, respectively for the 5 Euro bills.

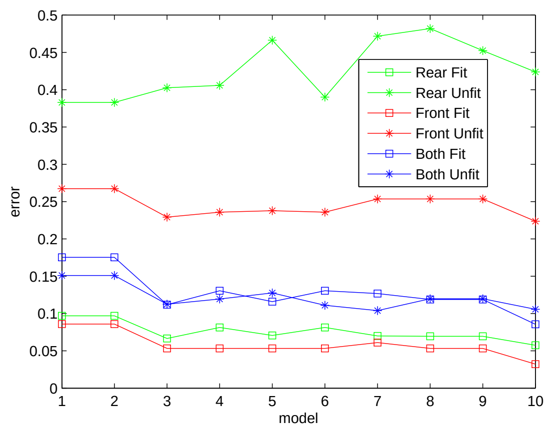

Figure 10: Error depending on the number of features - 10 Euro bills

4.2 PCA Results

RESULTS PCA

4.3 Intensity & Edges Results

RESULTS IE

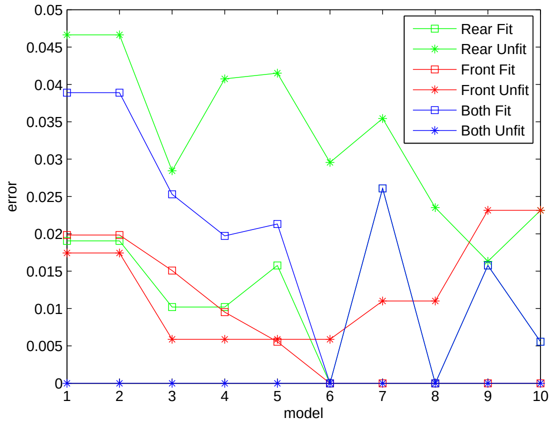

Figure 11: Error depending on the number of features - 5 Euro bills

4.4 Combined Results

Once all the strong classifiers for each of the methods above has been generated, we have combined their predictions using *Naive Bayes*.

The results for 10 Euro bills, respectively 5 Euro bills are shown in the Table1 and Table2.

Table 1: Results for 10 Euro bills

	Fit Error	*Unfit Error*
Haar	0.05	0.15
IE	0.1	0.025
PCA	0.083	0.05
Haar & IE	0.033	0.15
Haar & PCA	0.017	0.2
IE & PCA	0.017	0.075
Haar & IE & PCA	0.033	0.025

Table 2: Results for 5 Euro bills

	Fit Error	*Unfit Error*
Haar	0	0
IE	0.033	0
PCA	0.083	0.025
Haar & IE	0	0
Haar & PCA	0	0.025
IE & PCA	0.033	0.025
Haar & IE & PCA	0.033	0

5 Conclusion

From our experiments we have been able to learn the regions that contain the largest amount of information and help us distinguish between fit and unfit bills. These regions differ from one method to another, but in general the areas around the water-mark region and the middle of the bills have been proven to be the ones containing the most discriminative features.

As it can be noticed from the results in Section4.4, combining different techniques leads to an improvement in the performance of the final classifier.

Although the results obtained using these 3 methods are as good as we would have expected them to be, future work is possible and it should be mainly focused on finding a more powerful way of combining all the features used by all 3 techniques (Haar, PCA, Intensity & Edge) into a strong classifier.

We would also recommend that special attention should be payed to the validity of certain features of the images from the data set (intensity).

EDGE:: future work: Finding the intensity gradient of the image in stead of just canny edge detection and Non-maximum suppression
the first step of canny edge detection is to apply gaussian smoothing on the image. If the kernel used is too big and the borders are smoothed using zero padding this can eliminate very important data from the borders, which could hold very important distictive features for classifying clean and dirty/used bills.

References

[1] P. Viola & M. Jones:
Rapid Object Detection using a Boosted Cascade of Simple Features (CVPR 2001)

[2] AdaBoost:
http://en.wikipedia.org/wiki/AdaBoost

[3] G. Molenaar, A. Nusselder and K. Stefanov:
Eigen money approach

[4] J.M. Geusebroek:
Eigen money approach improved