Analysis of a Likelihood Ratio Classifier for Histogram Features

Melissa Tijink

Abstract— **In this paper an extended research is done on a new likelihood ratio (LLR) classifier as proposed by [1]. We analyzed the assumption that histograms, representing biometric features, can be modeled as draws from a multinomial distribution and the parameters of this distribution can be modeled as the Dirichlet probability density. We extracted the estimated parameter vector of the Dirichlet density from real data by training it with the LLR classifier and used this parameter vector to create new synthetic data. This synthetic data was used to train and test the classifier. In contrary to the results of training and testing with real data, the results with synthetic data were nearly perfect. This lead to the conclusion that the assumption of a multinomial distribution was not correct. Besides this, we compared the LLR classifier with another trained classifier: the support vector machine (SVM). In order to do 1:1 comparisons with the SVM, we concatonated the histograms. The results showed that the LLR classifier performed better, however after a simple experiment we concluded that concatenating the histograms does not result in a well functioning SVM.**

I. INTRODUCTION

In biometrics, biometric samples, e.g. fingerprints or facial images, are compared in order to establish whether they originate from the same individual or not. This is done by means of a so-called classifier. Good biometric features in these samples are discriminative for every individual. Histograms of image discriptors such as Binarized Statistical Image Features (BSIF) are an example of such features and can be compared, [12]. A graphical representation of this process can be found in figure 1. In [1] a new classifier based on the likelihood ratio is presented. The classifier was derived for 1:1 comparison and were tested on histograms of Binarized Statistical Image Features (BSIF), [1], [12]. The likelihood ratio classifier is optimal in Neyman Person sense, as explained in [1], since it reaches the maximum True Match Rate (TMR) for every given False Match Rate (FMR). The TMR and FMR will be explained later in this paper.

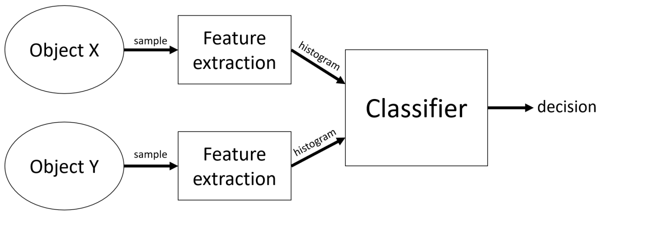

Fig. 1. Graphical representation of the process of comparing two objects

The performance of this newly developed classifier was compared to that of the chi-square classifier and the results presented in [1] were promising. We will now present two research goals that will be discussed in this paper.

The new classifier is based on the assumption that the histograms can be modeled as draws from a multinomial distribution and the parameters of this distribution can be modeled as draws from a Dirichlet probability density, [1]. However, observations suggest that this assumption may not be correct and the data is not (fully) multinomially distributed. It was found that the classifier performed better on synthetic data that was based on the Dirichlet probability density and multinomial distribution than on measured statistics of real data. The aim is to validate these model assumptions by deriving synthetic data from real data and train and test the classifier with this data. These recognition results can then be compared with the results of the real data.

The other research goal in this paper will be to compare the likelihood ratio classifier in [1] with another trained classifier. The likelihood ratio classifier in [1] is compared with the chi-square classifier. The likelihood ratio classifier is a trained classifier, unlike the chi-square. In [1] it is suggested to compare the classifier with other trained classifiers as the Support Vector Machine (SVM), [4]. For fair comparison the conventional ways in which the SVM is used to compare classes, one-vs-all and one-vs-one [9], are not suited. The solution presented in this paper is to concatenate the histograms of two datasamples, which will be explained in section IV-A.3.b.

This document starts with a short (mathematical) summary of the LLR classifier from [1]. Since not all readers might be familiar with the field of classifiers we will explain some terms and concepts in order to understand the results later on in this document. Then in section III the model assumptions of [1] will be checked. The original results will be examined, the method, results and conclusions will also be in this section. In section IV the comparison with the SVM classifier will be discussed. Starting with an explanation of the SVM, followed by the method and results. A discussion of the results follows, ending with a conclusion about this research goal. The paper ends with a summarizing conclusion and recommendations in section V.

II. THEORY

In this section a (mathematical) summary of [1] and the theory of several concepts related to this research will be explained. If the reader is already familiar with the concepts, he can continue at section III.

A. Likelihood ratio classifier

As mentioned in the Introduction (section I), the newly proposed classifier in [1] is the Likelihood ratio classifier for histogram features. We will give a short summary of the mathematical reasoning that results into this new classifier by [1].

Let us compare two histograms, $\sum_{i=1}^{n} x_i = X$ and $\sum_{i=1}^{n} y_i = Y$, the question is whether $\mathbf{x}$ and $\mathbf{y}$ are from the same 'person' (source). Two assumptions made in [1] are that histogram $\mathbf{x}$ is a realisation of random vector $\underline{\mathbf{x}}$ and that a probability $p_i \geq 0$ is associated to every bin i from $\mathbf{x}$, such that $\sum_{i=1}^{n} p_i = 1$. From here an interesting and important assumption follows, that we will investigate further in section III, namely that the outcomes that follow from observations are statistically independently allocated in the histogram. Thus, the realisation of $\mathbf{x}$ out of $\underline{\mathbf{x}}$ is the multinomial distribution with parameters $\mathbf{p}$, as shown in equation (1), [1].

$$P\{\underline{\mathbf{x}} = \mathbf{x}|\mathbf{p}\} = \text{Mult}(\mathbf{x}|\mathbf{p}) \tag{1}$$

When we then write the likelihood ratio to compare the two histograms, assuming that when $\mathbf{x}$ and $\mathbf{y}$ are from different persons (individuals), they are statistically independent, we get equation 2. In this equation S denotes the condition that $\mathbf{x}$ and $\mathbf{y}$ come from the same source.

$$\text{lr}(\mathbf{x}, \mathbf{y}) = \frac{P\{\underline{\mathbf{x}} = \mathbf{x}, \underline{\mathbf{y}} = \mathbf{y}|S\}}{P\{\underline{\mathbf{x}} = \mathbf{x}\}P\{\underline{\mathbf{y}} = \mathbf{y}\}} \tag{2}$$

In [1] it is argumented that a prior probability density $f_{\underline{\mathbf{p}}}(\mathbf{p})$ can be used to model the parameters of the multinomial distribution. Using this prior probability density, equation (2) can be rewritten to equation (3). In [1] it is then reasoned that the Dirichlet density can be used as the prior probability density. The expression for the Dirichlet density is as equation (4). In this equation the α is the parameter vector and $\text{B}(\alpha)$ represents the quotient of gamma functions that are part of the Dirichlet density, [1]. The complete description of $\text{B}(\alpha)$ and the gamma function can be found in the Appendix equation (11) and equation (12) respectively. Then equation (3) and equation (4) are combined, which results in equation (5), which is the likelihood ratio, [1].

$$\text{lr}(\mathbf{x}, \mathbf{y}) = \frac{\int_{\mathbf{p}} \text{Mult}(\mathbf{x}|\mathbf{p})\text{Mult}(\mathbf{y}|\mathbf{p}) f_{\underline{\mathbf{p}}} \text{d}\mathbf{p}}{\int_{\mathbf{p}} \text{Mult}(\mathbf{x}|\mathbf{p}) f_{\underline{\mathbf{p}}} \text{d}\mathbf{p} \int_{\mathbf{q}} \text{Mult}(\mathbf{y}|\mathbf{q}) f_{\underline{\mathbf{q}}} \text{d}\mathbf{q}} \tag{3}$$

$$\text{Dir}(\mathbf{p}|\boldsymbol{\alpha}) = \frac{1}{\text{B}(\boldsymbol{\alpha})} \prod_{i=1}^{n} p_i^{a_i - 1} \tag{4}$$

$$\text{lr}(\mathbf{x}, \mathbf{y}|\boldsymbol{\alpha}) = \text{B}(\boldsymbol{\alpha}) \frac{\text{B}(\mathbf{x} + \mathbf{y} + \boldsymbol{\alpha})}{\text{B}(\mathbf{x} + \boldsymbol{\alpha})\text{B}(\mathbf{y} + \boldsymbol{\alpha})} \tag{5}$$

Using a training set, a maximum likelihood estimator can be developed that computes the estimate $\hat{\alpha}$ for α. In [1], this is done by the Nelder-Mead downhill method [11], which needs an initial estimate for α, $\hat{\boldsymbol{\alpha}}_{\text{init}}$. This $\hat{\boldsymbol{\alpha}}_{\text{init}}$ is based on the expected value and variance of the representative set of bin probabilities, [1].

B. Data representation

In order to assess the performance of a classifier there are several terms and graphical plots. Since the reader might not be familiar with the world of classifiers, we will explain briefly the ones we use later in this report.

Classifiers have the task to categorize datasamples, to name a few examples: it is "Person A" or it is "not Person A", it is black or it is white, these samples are from the same source or not, etc. Afterwards it will be measured how many of these samples were categorized correctly. There will also be situations were the classifier misjudges, which results in false positives and false negatives, which, in biometric context, we call false matches and false non-matches. In the context of this paper we call the decision of a classifier that two samples are from the same individual when they are not, is a false match. The decision that two samples are from different individuals when they are in fact from the same person, is a false nonmatch.

The comparator's decision is based on a score, that the comparator assigns to each sample. When the result of a sample has a score that is above a certain threshold t it is decided that this sample belongs to class "A", when it has a score below this threshold it is decided that this sample belongs to class "not A". To get a full idea of the performance the false matches and false non-matches will be calculated for a varying threshold. This gives the False Match Rate (FMR), equation (6) and False Non Match Rate (FNMR), equation (7). In these equations s is the score and t is the threshold. The categories of datasamples in the context of this paper are 'mated' (two samples originating from the same individual) and 'nonmated' (two samples do not originate from different individuals). We can also calculate the True Match Rate (TMR), which gives an indication of the proportion of matches that are actually decided to be a match, equation (8).

$$\text{FMR}(t) = \frac{|\{s|s \geq t, \text{nonmated}|\}}{|\{s|\text{nonmated}\}|} \tag{6}$$

$$\text{FNMR}(t) = \frac{|\{s|s < t, \text{mated}|\}}{|\{s|\text{mated}\}|} \tag{7}$$

$$\text{TMR}(t) = 1 - \text{FNMR}(t) \tag{8}$$

A value that represents the perfomance of the classifier is the Equal Error Rate (EER). When the FMR and FNMR are plotted against the varying threshold, they will intersect at some point. This means that there is a threshold value for which the FMR and FNMR are equal. The error rate at this point is called the EER. The proportion of false matches is equal to the proportion of false non-matches. The lower this value, the better the system performs.

When we plot the TMR against the FMR we get Receiver Operating Characteristic Curve (ROC curve). It originates from radar analysis and it gives insight in the probability of detection vs. the probability of false alarm. It can, for example, be used in a cost/benefit analysis, [3].

III. Model assumptions validation

LLR as classifier is based on the assumption that the data has a multinomial distribution, as mentioned in section II-A. This is an important assumption and it can be questioned whether this assumption is valid. The multinomial distribution is a generalization of the binomial distribution and it can for example model the probability of counts for rolling a dice with k sides n times. It is important that the n trials are independent. As described in section II-A in [1] the assumption is made that the allocation of outcomes of observations into the histogram is statistically independent.
The LLR classifier was tested by the authors of [1] with two datasets: real data and synthetic data. Real data means histograms that are derived from actual pictures, in [1] the Smartphone Biometric Dataset from [13] is used. From this dataset 75 subjects are used for training and 71 for testing. Each subjects has 15 samples. From training followed an estimated parameter vector $\hat{\alpha}$, which was used for testing, [1].
The synthetic data is created using the multinomial random data function (from Matlab), which is based on the multinomial distribution. This required a probability vector as input, the synthethic data uses the Dirichlet distribution as probability vector. The parameter vector for this Dirichlet distribution is based on the reversed number of bins of the histograms. The synthetic data is created to be multinomially distributed, [1]. It appeared that classifier performed better when trained with synthetic data than with the real data. In figure 2 the ROC curve of the real data is visible, in figure 3 the ROC curve of the synthetic data is visible. Although we cannot equally compare them, since the underlying Dirichlet distributions are not alike at all, it is interesting to see that the classifier for synthetic data performs clearly better than the one for real data. This indicates that real data might not completely multinomially distributed, since we know that the synthetic data is multinomially distributed. In order to investigate this, we will create synthetic data that we can compare with the real data to see whether the assumption that data is multinomially distributed.

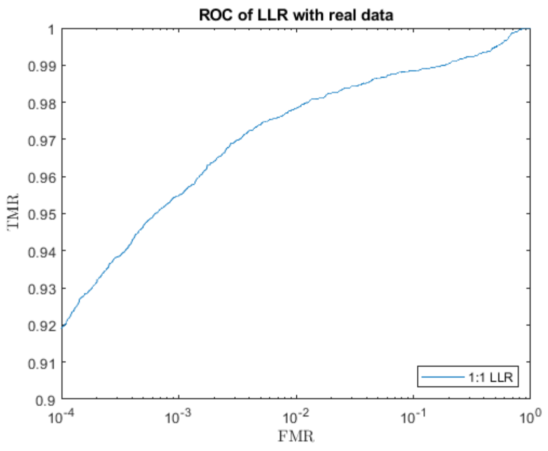

Fig. 2. ROC curve after training and testing the LLR classifier with real data.

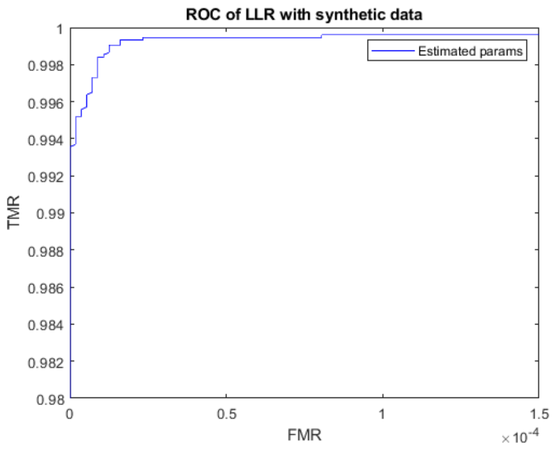

Fig. 3. ROC after training and testing the LLR classifier with synthetic data.

A. Method

In order to test whether the assumption was correct that the histograms of real data are multinomially distributed, we will do the following:

- Train the classifier with real data samples. This results in an $\hat{\alpha}$ for the supposedly Dirichlet probability density of the real data.
- Use this $\hat{\alpha}$ to create synthetic data. Create two datasets: one for training and one for testing. These will be of the same size as the ones used with real data (75 subjects for testing, 71 for training, consisting of 15 samples per subject). Creating new data is done by creating a probability vector out of the Dirichlet distribution with parameter vector $\hat{\alpha}$. Then a multinomial distribution with the probability vector as parameter vector is used to create the histograms.
- Train the classifier with the new synthetic training data.
- Test with synthetic test data.

The existing code from [1] for creating, training and testing of synthetic data and real data has been modified to create new code that fulfills the task as described above.

If the real data is indeed multinomially distributed, the results after testing with synthetic data should not differ too much from the results of the real data, since it is based on the estimated parameter vector from the Dirichlet distribution. However, when the results differ, we have reasons to assume that the initial assumptions were wrong.

B. Results

For the real data we used the Smartphone Biometric Dataset from [13], as used in [1]. This consists of in total 146 subject with each 15 samples. The dataset for training consists of 75 subjects, 71 subjects were used for the testing dataset. The same number of subjects and samples were created as synthetic data. The plot of the Error rates is visible in figure 4 and the EER is calculated to be 0. The TMR at FMR = 0.0001 is calculated to be 1.

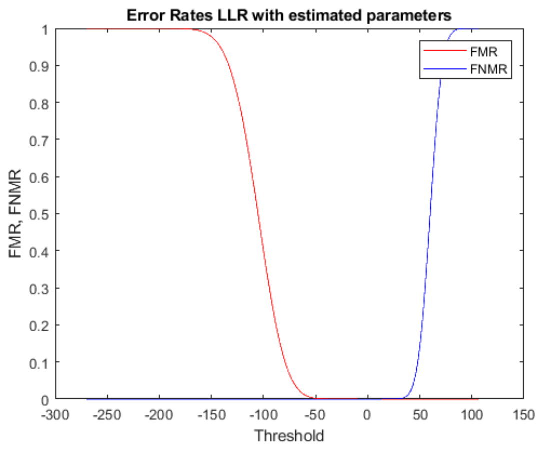

Fig. 4. Error rates of the LLR with the estimated parameter vector that followed from the new synthetic data.

C. Discussion/Conclusion

It is interesting to see that this system performs perfectly, the TMR jumps almost instantly to 1 and the EER reaches to zero. It vastly outperforms the LLR with only real data. This explains indeed that the original real data is not a perfect multinomial distribution. However, the classifier on real data still performs seemingly well, as the results from [1] showed. This means that the real data is not a perfect multinomial distribution, but it presumably does resemble one.

IV. Comparison with SVM

In [1] it is mentioned that the classifier the LLR classifier is compared to, is an untrained classifier (namely chi square). Since the LLR is a trained classifier, it is interesting to see its performance compared to a trained classifier. As an example the support vector machine (SVM), which is often used, is chosen.

A. Theory of the SVM

In this section the SVM and related topics will be explained. When the reader is already familiar with the concept of SVM he can go to section IV-A.3.b.

1) SVM explained: Support vector machines are two-class classifiers [4]. The SVM is trained with a set of labeled data. Labeled means that it is indicated to which category the data belongs, in the context of this paper these categories will be *mated* and *nonmated*. As the name of the classifier already unveils, the SVM uses vectors. It represents the data as vectors, these vectors are mapped into a high-dimensional space. The SVM then tries to find the seperating hyperplane with the largest margin towards the datapoints of the two categories, [4].

To intuitively understand the SVM an example will help. As shown in figure 5, there are two different categories (or classes), black and white circles. The hyperplane (indicated with 'H3') is placed where it has the largest margin towards the classes. H1 and H2 could not have been the hyperplanes, since H1 does not separate the classes and H2 does not have not the maximum margin.

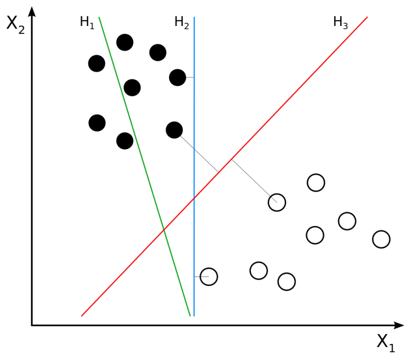

Fig. 5. H1 does not separate the classes. H2 does, but only with a small margin. H3 separates them with the maximal margin. Source: adapted from [5]

The hyperplane in figure 5 was relatively easy to find. This is because the classes were linearly seperable. Using the SVM it is also possible to find hyperplanes for non linearly seperable data, by transforming them to a higher dimension, [4]. This is called the kernel trick. An example of two classes that are not linearly seperable is visible in figure 6. When transforming the data to $z = x^2 + y^2$, you will get figure 7. Here it is possible to seperate the classes. The hyperplane in the original space is visible in figure 8.

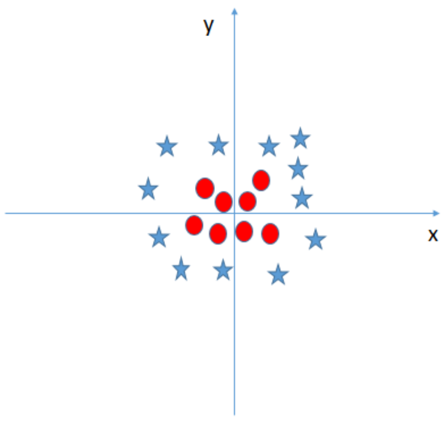

Fig. 6. Example of two classes that are not linearly seperable. Source: adapted from [6].

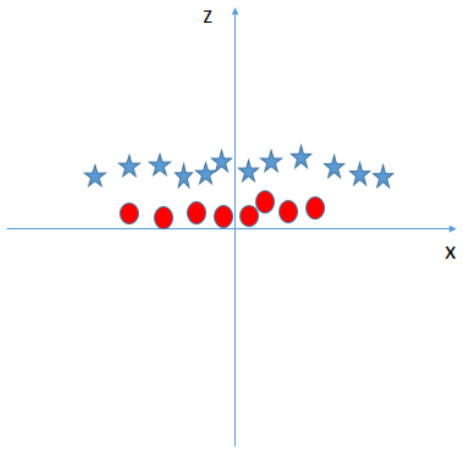

Fig. 7. Data of figure 6 transformed to $z = x^2 + y^2$.Source: adapted from [6].

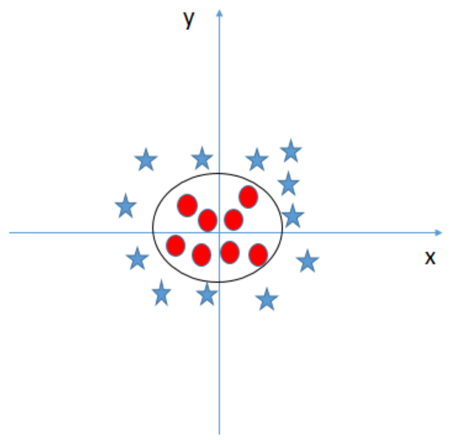

Fig. 8. Data of figure 6 seperated by a non-linear hyperplane. Source: adapted from [6].

2) Radial Basis Function Kernel: As explained a kernel helps to seperate the classes. The radial basis function (RBF) kernel is suited for non-linear classification, such as figure 6. The RBF kernel can be described as equation (9), where **x** and **x'** are two samples and γ is a parameter which will explained below, $|\mathbf{x} - \mathbf{x}'|^2$ is the squared Euclidean distance.

$$K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2) \tag{9}$$

The γ parameter influences how much influence one data sample has, it can be seen as the inverse of the standard deviation of the kernel (the radial basis function is a Gaussian function). It determines the 'width' of the function, [8]. Since it is the inverse, a small γ belongs to a Gaussian function with a large variance. In this situation two vectors that are far apart will be considered similar. A large value for γ belongs to a small variance, so two vectors will be considered similar if they are close to each other.

A parameter that has not been mentioned yet is the C parameter. The C parameter is of influence in the minization problem of the SVM, it is not part of the RBF kernel, [7] The C parameter determines how accurate the training will be, which is a trade off to the complexity of the decision function. It is possible that the two classes are not completely separable, thus there are training errors. The C parameter controls the influence of these errors (or each individual support vector). Setting C too low will result in accepting too much and the training will not be accurate anymore. Setting C too high will result in 'overfitting' and a complex decision function. [4].

3) Comparison: The conventional use of the SVM is not applicable in our research (comparing it to the LLR). To explain this, the conventional uses will be described and the problem will be illustrated using an example. Then a solution will be proposed.

a) Conventional use: With classification it is often the idea to identify an object. Imagine a situation where you have data of features of n objects. You then get new data of one of these object and you would like to identify which object it is. As mentioned before, SVM is a two-class identifier. It can seperate two categories. Now we have n *different* object. Usually this is solved in either a One VS All construction or a One VS One construction. Using the One VS All construction you have to make an SVM for each object, the classes are then *Object A* and *Not Object A*. Then some type of scoring system has to be used to determine to which class your undetermined object belongs. The one VS one method uses pairs, you create an SVM for all possible pairs of classes. This results in $(n \times (n-1)/2)$ SVMs. Again some kind of scoring system has to be used. The disadvantage of either approach is that the SVMs are specifically trained for the individuals in the datasets. They cannot be used to recognize unseen individuals, as is required in realistic biometric applications. [9]

b) Proposed solution: In [1] the classification context was different, it was a comparison. Are the two histograms from the same person? This is a different question than: who is this person? In this context it is possible to add new individuals to the dataset. The question if the histograms origin from the same person, is a binary question: the answer is either yes or no. This means we can make two classes: 'same persons' and 'not the same persons', to which we will refer to as 'mated' and 'non-mated' respectively. This means the data of two persons has to be combined in order to be presented as one sample. The idea is to concatenate the two histograms. This results in a long vector with either the data of two histograms from the same person or from different persons.

The expectation is that these classes are separable. The data from one person is expected to be similar and data from different persons should be different. When the histograms would only exist of one bin and we combine the samples, we can plot them on a two dimensional plane. If the data would have a mean around zero and a (expected) high covariance for mated data, it would look like figure 9. The nonmated data then has a covariance of zero, which is the expectation, and would look like figure 10.

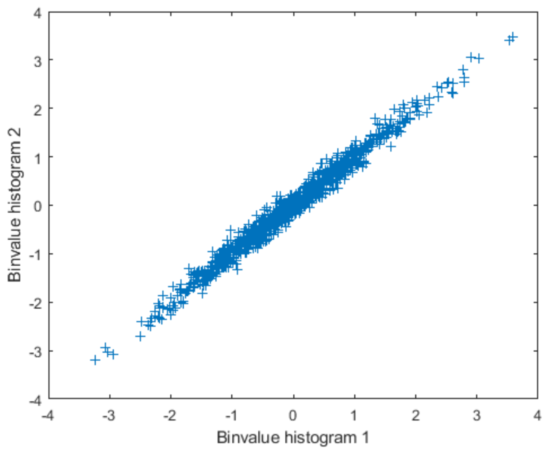

Fig. 9. Datasamples with a high covariance

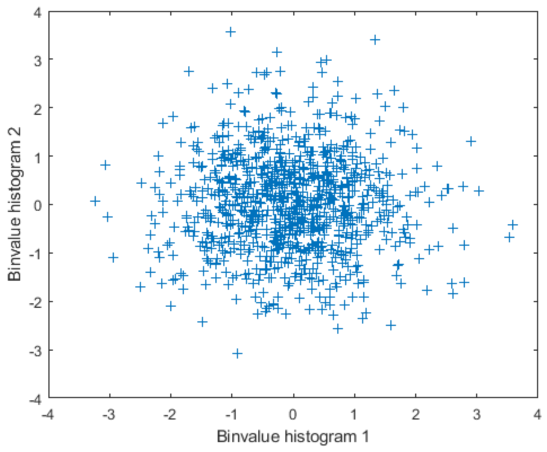

Fig. 10. Datasamples with a covariance equal to zero

B. Method

The SVM is created in Matlab, using the `fitcsvm` function from the statistics and machine learning toolbox. The complete code consists of six steps, which we will explain chronologically.

1) Step 1 Data: The data that is used for training and testing are histograms from the Smartphone Biometric Dataset, as used in [1]. The code from [1] is reused to load the data. There are 15 samples per subject and in total there are 146 subjects. 75 of the subjects are used for training, the 71 remaining subjects are used for testing. Every sample is a histogram with 256 bins. Besides the data of all the subjects, there is also an Identification vector created, that indicates to which subject the data belongs. This ID vector is usefull later on in this experiment.

2) Step 2 Normalize: The feature vectors have to be normalized, this way they contribute equally. This is done by dividing each histogram by its sum.

3) Step 3 Training: To evenly train the SVM, an equal amount of mated and nonmated datasamples have to be used. This is done the following way:

1) Calculate the total number of samples and make all possible combinations of sample(number)s using `Combinations` in Matlab.
2) Indicate whether the samples belong to the same subject using the ID tags.
3) Seperate the mated and nonmated combinations.
4) Randomly select the same amount of combinations from the nonmated combinations as the mated combinations.

The data has to be prepared for the training. As mentioned in section IV-A.3 the (selected) histograms will be concatenated. There is also a vector created containing the labels, mated or nonmated.

To train the SVM the `fitcsvm` function from Matlab is used. As described in section IV-A.2 the RBF kernel is suited for our problem. The γ parameter can be found in the Matlab function parameter `Kernelscale`, which equals $\frac{1}{\sqrt{\gamma}}$. The C parameter is called the `BoxConstraint`. The `KernelScale` parameter and `Boxconstraint` parameter can be optimized, using the input argument `OptimizeHyperparameters`. According to the description of Matlab it attempts to minimize the cross-validation loss (error) for `fitcsvm` by varying the parameters, [10]. The default is Bayesian optimization and it searches for the BoxConstraint and KernelScale log-scaled in the range of [1e-3, 1e3]. The number of function evaluations is 30. After optimization Matlab gives the best observed values for the parameters and the best estimated values (according to the models), [10].

4) Step 4 Testing: The SVM is trained, so now we can test it, using the testsamples. As mentioned in step 1 (IV-B.1) we have 71 test subjects with each 15 samples. We create again combinations of mated and nonmated subjects, it is possible to use all combinations for testing. Again the data is concatenated and labels are created. Using the `predict` function from Matlab the classification scores can be obtained. The SVM classification score for classifying observation X is the signed distance from X to the decision boundary ranging from -∞ to +∞. A positive score for a class indicates that X is predicted to be in that class. A negative score indicates otherwise, [2].

5) Step 5 Analysing the results: Since we know which combinations are mated and which are nonmated, we can seperate the scores accordingly. We will use the scores for the mated class, which means positive scores for X to be in that class and a negative score for X to be in the nonmated class. Ideally all the scores are either positive or negative. To see if this is the case, the normalized histograms of the scores will be used.

6) Step 6 Presenting the results: The last step is to present the results so they can be analyzed and discussed. The FMR and FNMR will be plotted against a varying threshold value and ROC curve will be plotted. The FMNR can be calculated as in equation (7), the FMR as in equation (6) and the TMR as in equation (8). The s is the score, the t in all three equations is the threshold for the scores. The FNMR, FMR and TMR are calculated for 1000 values of the threshold, ranging linearly from the minimal score to the maximum score.

C. Results

After optimization the following parameter values were observed best and used for the results:

- KernelScale = 0.032356
- BoxConstraint = 2.7696

As mentioned in the method (IV-B.5) two normalized histograms are created of the scores for the mated and nonmated testsamples. These histograms are visible in figure 11. It is already visible that there is no clear separation in scores for the mated and nonmated samples, which whould have meant a clear seperation of the peaks of the histograms.

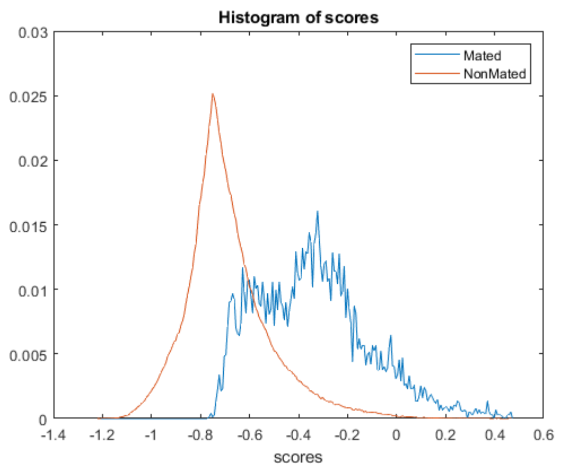

Fig. 11. Histograms of the scores after testing. In blue the histogram for the mated data, in red the histogram for the nonmated data.

TABLE I
RESULTS OF USING THE SMARTPHONE BIOMETRICS DATASET (FROM [13]) AS USED IN [1]

	EER	TMR $\times 10^{-2}$ at FMR =0.0001
LLR	0.25×10^{-2}	85.45
SVM	0.198	1.68

Besides the histograms, the plot of error rates and the ROC plot are avaiable, visible in the Appendix figure 13 and figure 14 respectively. The EER of this system is about 0.198. The EER of the LLR for 1:1 histogram comparison is 0.25×10^{-2} for the Smartphone Biometric Dataset, [1]. In [1] the value for the TMR at FMR=0.0001 is calculated too. For LLR for 1:1 histogram comparison this value equals 85.45×10^{-2} for the Smartphone Biometric Dataset, [1]. For the SVM this value is about 1.68 $\times10^{-2}$. The results are also visible in table I

D. Discussion

It is surprising to see that the SVM performs so poorly compared to the LLR classifier. The overlap in the histograms (figure 11) indicates that the two classes were not easily separable. The cause could be that the concatenated data cannot be seperated well by an SVM.

A simple test to confirm this is to concatenate two features that consist of only one value, instead of histograms of 256 bins. The space where the new vector is mapped on is then in 2D. For the 'mated' data we create 1000 vectors using the multivariate normal distribution. To simulate the idea that they are similar, a high covariance will be used:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} 1 & 0.99 \\ 0.99 & 1 \end{bmatrix}\right)$$

The nonmated data, again 1000 vectors, is created using the multivariate normal distribution. To simulate that the data is from a different source, a covariance of zero will be used:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$$

The scatter plot of the mated data looks like figure 9, the one of the nonmated data looks like figure 10. Similar wise testdata is created. The training and testing of the SVM is done similarly to the method described in section IV-B. This gives the histograms visible in figure 12. The EER is equal to 0.111 and the TMR at FMR = 0.0001 equals about 71×10^{-2}. The plot of the error rates and the ROC curve are available in the appendix figure 15 and figure 16.

We can compare the results of this SVM with a simple classifier based on a dissimilarity measure. We calculate the score according to equation (10), where s is the score and $\mathbf{x}$ and $\mathbf{y}$ are the features, using the same testdata as was used to produce figure 12 . The histogram of these scores can be found in the appendix figure 10. The peaks of the scores of the mated and nonmated data are comparable with the ones that the SVM produced (figure 12). This means that the performance of the SVM is not worse than that of a simple classifier.

$$s = -(\mathbf{x} - \mathbf{y})^2 \tag{10}$$

Another point of discussion worth mentioning is that the LLR tests were done with shortened histograms, the last 76 bins were removed and only the 180 first bins were used in testing and training the LLR classifier, [1]. In the test with the SVM the original histograms of 256 bins were used. However, looking at the results presented in figure 15 and figure 16 it seems like the LLR would have outperformed the SVM even with fewer bins.

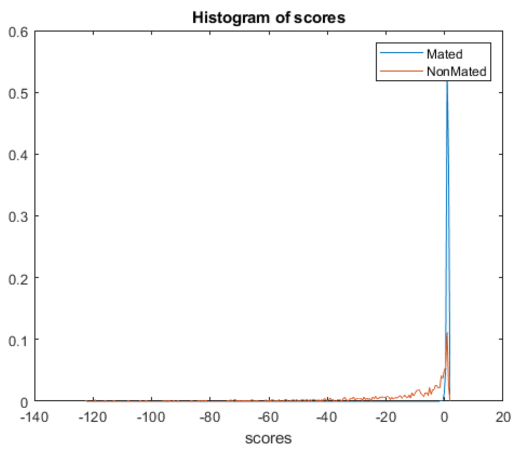

Fig. 12. Histograms of the scores after testing of the 2D data. In blue is the histogram of the mated data, in red the histogram for the nonmated data.

E. Conclusion

The SVM using the `fitcsvm` function from Matlab does not perform better than the LLR classifier from [1]. The LLR classifier appears to be considerably better than the SVM, looking at the values of the EER and the TMR at FMR = 0.0001 (presented in tabel I. The poor performance of the SVM could be the fact that the two histograms were concatenated and that the two categories (mated and nonmated) were not seperable this way. This was investigated by testing it with 2D vectors, described in section IV-D. It appeared that the performance of the SVM on 2D is not worse than a simple classifier based on a dissimilarity measure. This all leads to the conclusion that SVM is not suitable for 1:1 histogram comparison and that the concatenating of the histograms is not the cause of poor performance.

V. CONCLUSIONS

In this paper we discussed two research goals related to the LLR classifier proposed by [1].

We checked the model assumption of [1] that histograms can be modeled as draws from a multinomial distribution and the parameters of this multinomial distributions as the Dirichlet probability density. We did this by creating synthetic data which had for sure these characteristics. The Dirichlet probability density is determined by a parameter vector. This parameter vector was retrieved from the estimated Dirichlet parameter vector of real data from the Smartphone Biometric Dataset (from [13]) as used in [1]. When the histograms of the real data would have been multinomially distributed the performance of the classifier trained with the synthetic data should be comparable with the performance of the classifier trained with real data. However, it appeared that the LLR classifier performed nearly perfect on this newly created synthetic data. This lead to the conclusion that the assumption that the histograms of real data are multinomially distributed is not correct. However the performance of the classifier on real data is sufficiently good and the near perfect performance on the synthetic data means that the estimated parameter vector of the real data approximates the multinomial distribution. The question why the statistics of these histograms are not independent, could be answered by future research.

Furthermore, the performance of the LLR classifier as proposed by [1] was compared with another trained classifier: the support vector machine. To enable 1:1 comparisons for the SVM we had to concatenate histograms of two samples and present it as one sample. The RBF kernel was used for the SVM with optimized parameters γ and C. The results after training with the same dataset as the LLR classifier are presented in table I. It can be seen that the LLR classifier outperforms the SVM. The assumption that the histograms could be concatenated might have been wrong, this was tested by concatenating histograms of only one bin. The mated samples were simulated by draws from a multivariate random distribution with a high covariance, the nonmated samples by draws from the same distribution but with zero covariance. The results of this test are shown in figure 15, where it is clear that the two peaks are not seperable. However, when the scores are calculated using a simple classifier based on a dissimilarity measure, they do not differ too much from the scores of the SVM (figure 17). This leads to the conclusion that the LLR classifier performs better than the SVM on 1:1 comparison, however SVM is not the ideal solution to perform 1:1 comparisons on histograms. Since this implementation of the SVM was not suitable to compare with the LLR classifier, further research can be done by comparing the LLR classifier with other trained classifiers.

REFERENCES

[1] R. Veldhuis, K. Raja and R. Ramachandra. ”A Likelihood Ratio Classifier for Histogram Features.”

[2] *Predict labels using support vector machine (SVM) classifier - MATLAB predict- MathWorks Benelux.* Available on: `https://nl.mathworks.com/help/stats/classreg.learning.classif.compactclassificationsvm.predict.html#d120e596412` Last checked on: 3-3-2019

[3] *Detector Performance Analysis Using ROC Curves- MATLAB Simulink- MathWorks Benelux.* Available on: `https://nl.mathworks.com/help/phased/examples/detector-performance-analysis-using-roc-curves.html` Last checked on: 3-3-2019

[4] C. Cortes and V. Vapnik. *Support-Vector Machines.* Machine Learning, 20, 273-297 (1995).

[5] *Support-vector machine.* Available on: `https://en.wikipedia.org/wiki/Support-vector_machine#/media/File:Svm_separating_hyperplanes_(SVG).svg`

[6] *Understanding Support Vector Machine algorithm from examples.* Available on: `https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/`

[7] H. Shunjie, C. Qubo and H. Meng. ”Parameter selection in SVM with RBF kernelfunction”. *Automation Congress 2012*, Puerto Vallarta, Mexico, 2012, pp. 1-4. Available on: `https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6321759&tag=1`

[8] G. F. Smits and E. M. Jordaan, ”Improved SVM regression using mixtures of kernels,” *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No.02CH37290)*, Honolulu, HI, USA, 2002, pp. 2785-2790 vol.3. doi: 10.1109/IJCNN.2002.1007589

[9] J. Milgram, M. Cheriet and R. Sabourin. “One Against One” or “One Against All”: Which One is Better for Handwriting Recognition with SVMs? *Tenth International Workshop on Frontiers in Handwriting Recognition*. Oct 2006, La Baule (France), Suvisoft, 2006. ¡inria-00103955¿

[10] ”Train binary support vector machine (SVM) classifier” *MATLAB fitcsvm - MathWorks Benelux*. Available on: `https://nl.mathworks.com/help/stats/fitcsvm.html#bt7oo83-5`

[11] J.A. Nelder and R. Mead. ”A simplex method for function minimization.” *The Computer Journal*, 7(4): 308-313, 1965.

[12] J. Kannala and E. Rahtu, ”BSIF: Binarized statistical image features,” *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, Tsukuba, 2012, pp. 1363-1366.

[13] K. B. Raja, R. Raghavendra, M. Stokkenes, and C. Busch. Smartphone authentication system using periocular biometrics. In *2014 International Conference of the Biometrics Special Interest Group (BIOSIG)*, pages 1–8, Sept 2014.

VI. Appendix

A. Equations

$$\mathrm{B}(\alpha) = \frac{\prod_{i=1}^{n} \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{n} \alpha_i)} \tag{11}$$

$$\Gamma(z) = \int_0^\infty x^{z-1} e^{-x} dx \tag{12}$$

B. Figures

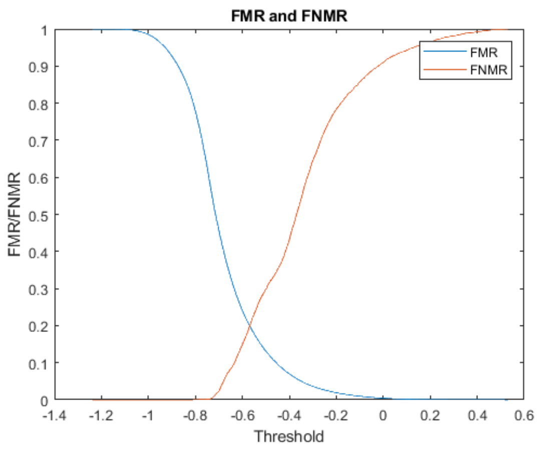

Fig. 13. In blue the FMR against the varying threshold for the score and in red the FNMR against the varying threshold for the score.

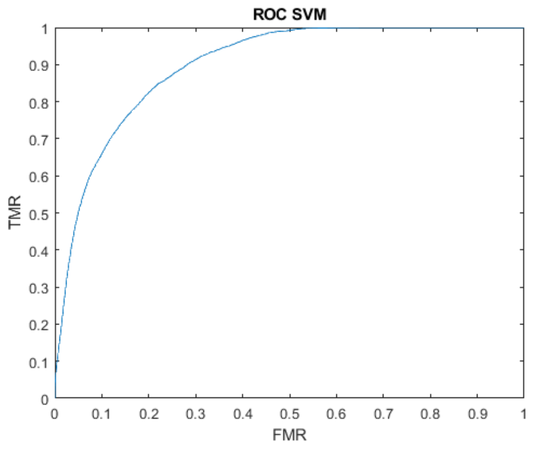

Fig. 14. ROC curve of the SVM

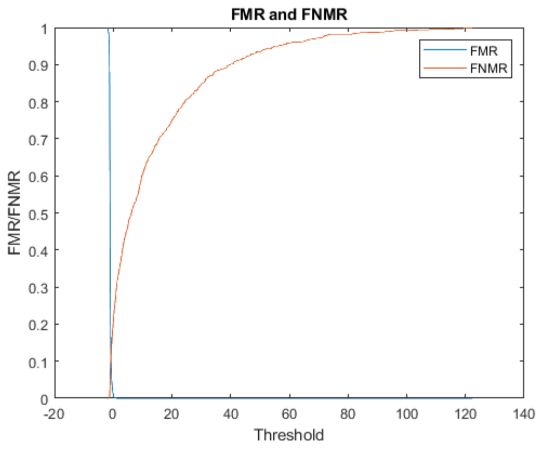

Fig. 15. In blue the FMR of the 2D testdata against the varying threshold for the score and in red the FNMR of the 2D testdata against the varying threshold for the score.

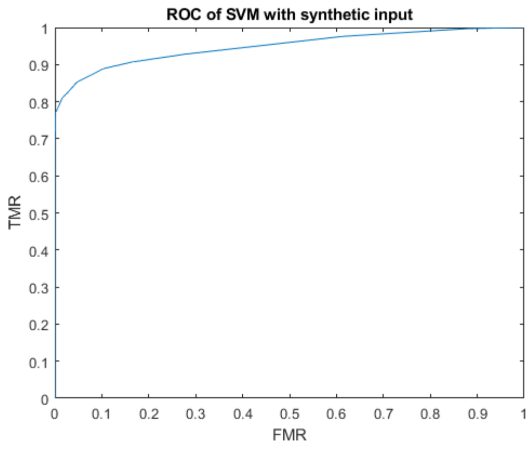

Fig. 16. TMR against the FMR of the 2D testdata.

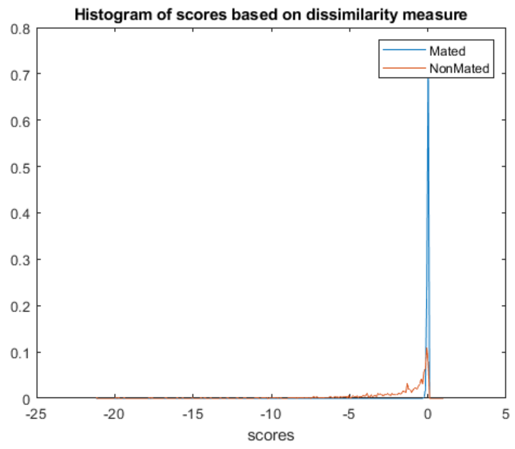

Fig. 17. Histograms of the scores of the dissimilarity classifier described in equation (10).