university of groningen / faculty of mathematics and natural sciences / artificial intelligence

Non-linear combination of features for writer identification

Bachelor's Project Thesis

Lisette Boeijenk, s2499428, e.l.boeijenk@student.rug.nl,
Supervisor: M.Sc. S. He

Abstract: Writer identification is very important in pattern recognition. Previous works have shown that combinations of different features result in a higher performance than the individual feature involved in the combinations. Traditionally, features are combined linearly, averaging the distance measurements for writer identification. However, non-linear combination of the feature vectors has not yet been studied thoroughly. In this thesis, a non-linear combination of different features is studied in which the dimensions of features are first reduced using the principal component analysis (PCA) method and then different features are non-linearly combined using a kernel function. The proposed non-linear combined method is evaluated using five different both textural and allographic features on four data sets. Three classical kernel functions are applied to the features, such as the gaussian, sigmoid and polynomial kernels. Experimental results show that the PCA reduced features result in higher performances for higher dimension features and that the non-linear combination of PCA reduced features using kernel functions result in lower performance except for the combination of Local Binary Pattern and Chain Code. Non-linear combination is thus not preferred over linear combination.

1 Introduction

Writer identification has been studied intensively in the last two decades, which has led to the introduction of many feature extraction methods [1]. The problem that arises with these feature extraction methods is how they can be combined to achieve a higher performance for automatic writer identification.

Given a query document that consists of a digitalized handwritten text, the task of writer identification is to find the document in a reference database written by the same author. The digitalized handwritten texts have either been scanned or photographed or are recordings of the texts being written. Writer identification has applications in forensic science [2, 3] and historic documents analysis [4, 5]. Experts in these fields might have as task the identification of the authorship or authenticity of a query document. The query document might be forensic related, such as a ransom note or a threatening letter, but also historic related, such as a signature.[6]

For forensic science writer identification is also useful as an identification method of suspects, since handwriting is a behavioural biometric modality. The advantage of behavioural biometrics (e.g., voice, signature, handwriting) over physiological biometrics (e.g., DNA, fingerprint, retina) is that behavioural biometrics do not require cooperation of the subjects and are less invasive. Yet, up to this day the accuracy for identification is quite lower for behavioural biometrics than physiological biometrics. For the application of writer identification in forensic science, a performance of 100% is needed on a hit list of 100 writers from a database of less than 100000 writers, for having 100 suspects can be managed in a police investigation [7]. This target performance is not yet feasible by the current systems.

This thesis focusses on the offline and text-independent writer identification problem. The methods (or features) used for writer identification can be grouped in two categories, textural (or statistical) and allographic (or grapheme-based) features. Textural features are based on writing slant, curvature, and pen grip, while allographic features are based on character shapes that are characteristic for the writer. Character shapes (or allographs or graphemes) do not need to be complete letters

of the alphabet, but can be smaller.

Previous studies have shown that the performance of writer identification increases when two different features are combined [5, 8, 9] since different features capture different kinds of information from the handwritten sample. Especially the combination of features from different feature groups, such as textural and allographic features has proven to be quite successful [4, 10]. When combining features, most researchers take a weighted average of the distances produced by the different features, leading to new distance measures between the writers [4, 5, 8–10]. The method of combining features using (weighted) averages is called linear combination in this thesis.

The purpose of this thesis was to look into a method of combining the features at an earlier stage, at the vector-level by applying a kernel to two features, thus receiving a new, higher dimensional vector that describes the handwritten sample. In the thesis this method is called non-linear combination. A kernel is a mathematical function that projects vectors into a new, higher dimensional plane using the inner products. Different kernels were used to examine the effects on the performance and to see if non-linear combination would result in higher performances than linear combination. The kernels are discussed more extensively in Section 6.

When a kernel function is applied to two vectors of size N, the new vector is quite high in dimensions namely $N * N$, and thus would take a long time to evaluate. One way to solve this problem is to use the PCA method to reduce the feature dimensions before combining. A combination of kernelization with PCA already exists and is called "kernel PCA" [11, 12]. In kernel PCA, the vectors are first kernelized to a higher dimensional matrix and then the principal eigenvectors of the kernel matrix are computed instead of the covariance matrix, which reduces the computations needed. By contrast, in this thesis a new method was tested, which was to first apply PCA to the features to reduce them in dimensionality without losing too much information and performance and *then* kernelizing the reduced features. This method was chosen on the theory that the PCA would linearly find the smallest dimension needed to describe the dataset by a feature. Applying a kernel function on two of these reduced features would then result in a non-linear optimization of the strong points of the two features, resulting in a higher performance.

In the next section, a survey is made on offline, text-independent writer identification research, the combinations between features, and the application of kernels and dimensionality reduction on the features. Section 3 describes the data sets used in this thesis. The features that were used are described in sections 4. Sections 6 and 5 explain the kernels used and the method for dimensionality reduction. The implementation of the features, kernelization and dimensionality reduction for writer identification is described in section 7. The experimental results are given and discussed in section 8 and the thesis is concluded in section 9.

2 Related work

Writer identification is part of the field of automatic handwriting recognition [3]. An overview of automatic handwriting recognition until 1989, including writer identification, can be found in [3].

Within the field of writer identification there is the distinction between offline and online methods. Offline methods perform the identification on digitized images of handwriting, while online methods perform the identification at the time of writing, for which some sort of visualization over time is needed, for example filming the writer as he writes or by recording the writing on a tablet.

In addition, there is a distinction between text-dependent and text-independent methods. For text-dependent methods the content of the text is predefined, as text-dependent methods relate the handwritten characters or words to the known transcript. For text-independent methods there are no textual constraints since these methods do not require any information about the characters or words in the handwritten text. [13]

Several features have been proposed in the literature for offline and text-independent writer identification. The work in [4] introduced three features and compared them to other features, in addition linear feature combinations were tested where the final distance was calculated as the simple or weighted average of the separate distances. Combining features from different categories increased the performance in [4]. The conclusion that combining features results in higher performances than

the individual features involved in the combinations was drawn in an earlier research as well [10]. In [10], the linear combination of features was also calculated by computing an average of the distance functions. The conclusion was that this combination method "diminishes the risk of a biased solution, while capturing the most of the achievable increases in writer identification and verification performance."

The Quill and QuillHinge features were proposed in [5]. Quill finds the probability distribution between the direction and the width of the ink. QuillHinge is a mix between the Quill and Hinge features. Linear combinations of Quill and QuillHinge with Hinge[4, 7] and Fraglets[14] were made, once more based on averages, and the conclusion was that combining the features gives a small improvement in performance of writer identification both on historical and modern handwriting data sets.

The study in [1] surveyed writer identification methods on different languages and scripts, not only on Roman scripts but also on Chinese, Arabic and Indic scripts, which concluded that text-dependent features have a higher performance than text-independent features and that a new trend has begun that shifts the focus from predominantly Roman scripts towards non-Roman scripts and even multi-lingual scripts. The same trend can be found in [8], where the Junclets feature was proposed and tested on English data sets, a Chinese and a mixed Chinese/English data set. Junclets or linear combinations of features with Junclets outperformed Quill, Hinge and QuillHinge, especially on the data set in which Chinese subjects wrote a paragraph in English.

In [2] a different method for feature combination was proposed, where not the average of the distance measurements was taken, but the feature vectors themselves were combined into a new vector. A new technique is described in [2] for offline writer identification using a codebook of connected-component contours (CO^3). The new, combined, feature vector was created by adjoining the two features without feature-group weighing. Optimization tests only gave slight improvements. The combination of the new CO^3 feature with a Hinge-based feature resulted in better performances.

In [15] an experimental research was conducted on diverse feature selection and extraction methods, all of which resulted in reduced feature sets. PCA was used to linearly combine features into new feature sets. The performance of PCA was marginally higher than the baseline.

The study in [12] proposed kernel PCA and tested this kernel PCA with different kernels (polynomial, sigmoid, gaussian), which resulted in fine performance on pattern recognition. The proposal proposed in [12] was to use kernel methods on classical algorithms, creating non-linear variants.

3 Data sets

For this thesis the experiments were conducted on four data sets: Firemaker [16], IAM [17], CERUG-EN, and CERUG-CN [8]. Example pages from these data sets can be found in figure 2.1.

The Firemaker data set consists of handwritten documents from 250 Dutch subjects. In this thesis page 1 vs page 4 was used, following the works [4, 8]. For page 1 the subjects were asked to copy a text in Dutch, for page 4 the subjects were asked to describe a cartoon in their own words. Both pages contain primarily lower-case letters.

The IAM data set consists of handwritten English documents of varying content. A modified subset was used which contains two documents of varying length per writer from 650 writers [4]. Again both pages contain primarily lower-case letters.

The CERUG data set contains English and Chinese handwritten documents from 105 Chinese subjects and can be divided into two subsets: the CERUG-EN and CERUG-CN data sets.

On the CERUG-EN data set page 3, for which the subjects were asked to copy two paragraphs in Enghlish, was split in two subpages, resulting in two pages of each one paragraph.

On the CERUG-CN data set page 1 vs page 2 was used. For page 1 the subjects were asked to copy two paragraphs in Chinese, for page 2 the subjects were asked to write about topics they liked in Chinese.

For this thesis no separation was made in the form of a test set and training set, all the handwritten samples of each data set were put together in one folder, creating a testing environment that is more likely to occur in the real world and is more challenging.

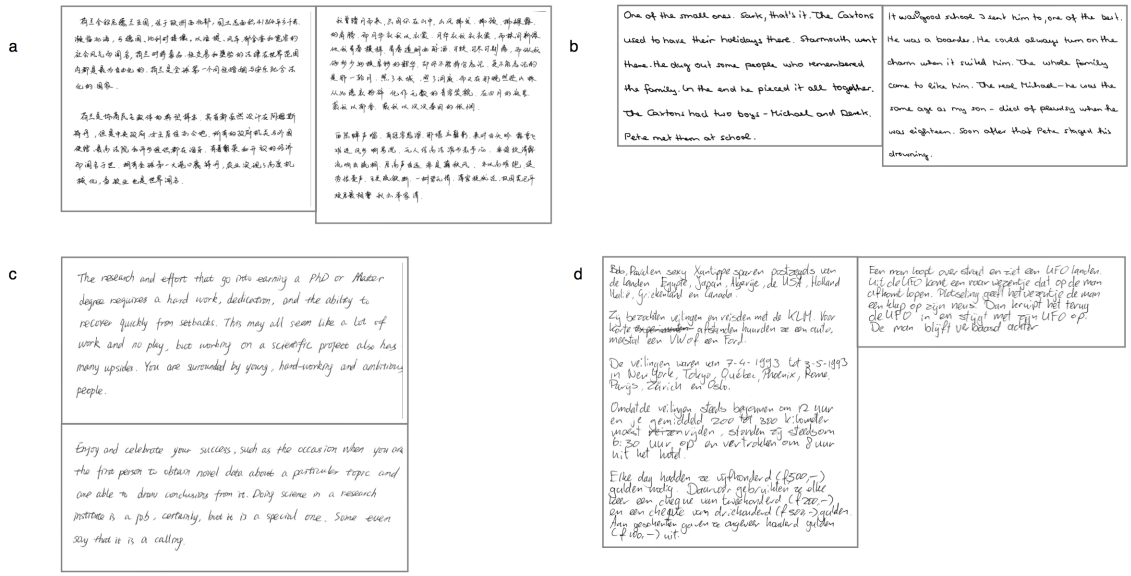

Figure 2.1: Example pages from (a) CERUG-CN, (b) IAM, (c) CERUG-EN, and (d) Firemaker.

Table 4.1: Features used in this thesis and their corresponding dimensionality and level. ("Abbr" is short for abbreviation and "Dim" is short for dimensionality.)

Feature	Abbr	Dim	Level
Local Binary Pattern	LBP	254	Textural
Chain Code	CC	9	Textural
Hinge	Hinge	253	Textural
Connected Component Contours	CO^3	900	Allograph
Junclets	Junclets	900	Allograph

4 Feature extraction

To identify a writer based on handwriting, an informative description of the handwriting needs to be constructed. This description takes the form of a vector and is calculated by a *feature extraction* algorithm. As mentioned in section 1 there are two main feature categories, the texture-level features and the allograph-level features. This section covers the five feature extraction methods used in this thesis. Each of the features used in this thesis was programmed following the guidelines in accompanying papers. A summary of the dimensionality and level of the features can be found in table 4.1.

4.1 Texture-level features

Texture level features describe the texture of the images of the handwritten samples. These textures are calculated both from the pixels and the probability of distinctive relationships between the pixels. Three texture level features were considered in this thesis: the local binary pattern, the chain code and the hinge feature. These three features were chosen for their increasing complexity and ability to capture textural information, respectively. In the following subsections these three texture-level features will be reviewed.

4.1.1 Local binary pattern

The local binary pattern (LBP) is a feature that is not proposed specifically for writer identification, but is a general feature used for classification in the field of computer vision. It was found to be quite effective on texture classification [18]. The algorithm of LBP used for this thesis was based on the algorithm described in [18]. The handwriting images are first transformed to gray-scale before the LBP feature is extracted. The LBP feature takes the 8 neighbours of each pixel and and thresholds the gray-scale values of these neighbours to the gray-scale value of the central pixel, returning a 0 for values lower and a 1 for values equal to or higher than the center value. The resulting binary values are weighted by powers of two calculated from the neighbouring position and summed over, giving the LBP code, see figure 4.1 for an illustration. A his-

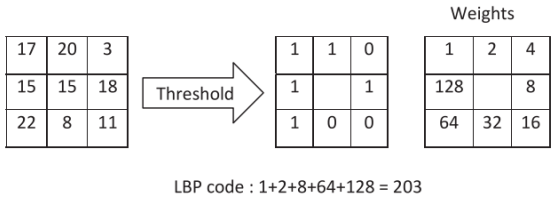

Figure 4.1: An example of the LBP code calculation as presented in [18].

togram is built of the LBP codes for the pixels containing 254 bins and is normalized. This means that the LBP feature has a dimensionality of 254.

The exact details on how to implement the LBP feature used in this thesis can be found in [18].

4.1.2 Chain Code

The algorithm used for the Chain Code feature in this thesis was based on [9]. The handwriting images are first transformed into gray-scale images and then binarized using Otsu's global thresholding algorithm [19], after which the connected components are extracted using 8-connectivity. For each connected component the angle with the next connected component is calculated, separated in the 8 directions of 3-by-3 neighbours as illustrated in figure 4.2. A 9th direction code is added to handle other directions than the 3-by-3 neighbours, for example where the end of the connected components meets the beginning, which would mean that both have the same position. A histogram is constructed of the resulting chain codes (i.e. numbers between 1-9) and is normalized to form the final feature representation. This results in the Chain Code feature with the dimensionality 9.

More details about the implementation of the Chain Code feature can be found in [9].

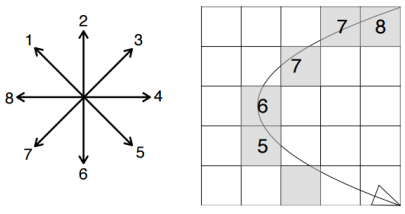

Figure 4.2: An illustration of Chain Code (the number on the pixels) on the contours of the ink trace (the gray pixels).

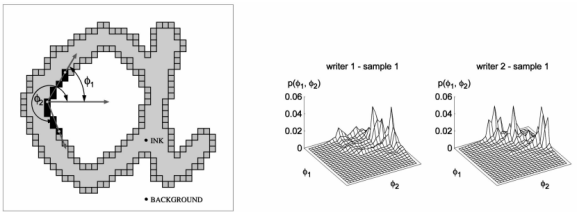

Figure 4.3: An illustration of Hinge feature extraction as presented in [4].

4.1.3 Hinge

The Hinge feature used for this thesis was based on the algorithm described in [4]. The Hinge feature is quite similar to the chain code feature. The handwriting images are transformed into gray-scale, binarized using Otsu's algorithm [19] and the 8-connectivity component contours are extracted. For each of the connected components, the angle with the "next" and "previous" components is calculated, giving a ϕ_1 and ϕ_2. The "next" and "previous" components are separated from the center pixel by edge fragments of n pixels long. An example of this feature is illustrated in figure 4.3 using an edge fragment of 7. For this thesis an edge fragment of 7 pixels was used. A histogram in the form of a 2D matrix of the resulting angles ϕ_1 and ϕ_2 is created, where only angle combinations are considered where $\phi_2 > \phi_1$. The histogram is then normalized and transformed into a 1D array. The resulting Hinge feature has a dimensionality of 253.

More details on the Hinge feature can be found in [4].

4.2 Allographic-level features

Allographic-level features use graphemes, which are fragments of the handwriting, to build the feature representation. For allographic-level features the assumption is that writers can be seen as stochastic generators of these graphemes [4]. By this assumption a writer can be described by computing a probability distribution on grapheme usage for that writer. This computation is done in three steps. First the handwriting connected component is segmented, so as to extract the graphemes from the image. This is often done using heuristics, i.e. using the minima in the lower contours of the handwritten words to separate the word. After segmentation, a codebook is generated from the graphemes using a clustering method. This is necessary since

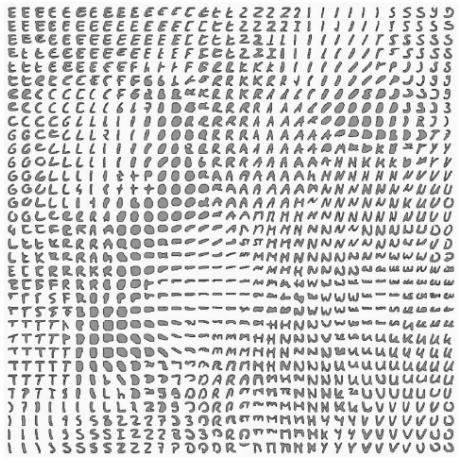

Figure 4.4: Illustration of codebook of 33x33 CO^3, using Kohonen SOFM on 26k samples, as presented in [2].

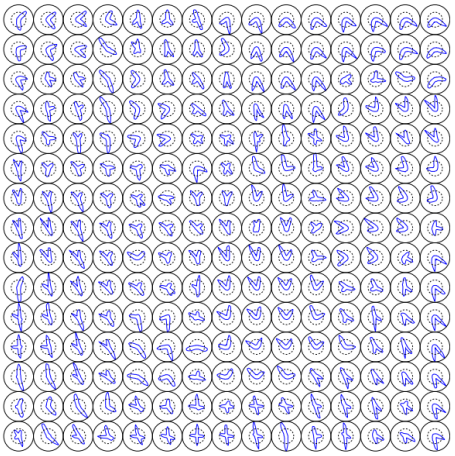

Figure 4.5: Example of codebook on 225 junctions using Kohonen 2D as illustrated in [8].

after segmentation the amount of graphemes obtained is too large to work with. By clustering, a shape codebook is created in the form of a reduced list of graphemes. Examples of such codebooks are illustrated in figure 4.4 and 4.5. Different clustering methods can be used, depending on the task at hand. Another variable is the size of the codebook, different sizes result in different performances, so this variable needs optimization. In the last step a probability distribution is calculated by segmenting the handwriting samples and matching the resulting graphemes to the graphemes in the codebook. The resulting distribution is the characteristic of a writer.

In this thesis two allographic-level features were used for testing: Connected Component Contours and Junclets. These features will be viewed further in this section.

4.2.1 Connected component contours

The first allographic-level feature considered in this thesis uses a codebook of connected component contours and is called CO^3. CO^3 is based on the algorithm described in [2]. The handwritten images are preprocessed to obtain the connected component contours. First the images are transformed into gray-scale, then they are blurred for smoothing and binarized using the mid-point gray value. Subsequently, the connected components are extracted and the contours are computed using Moore's algorithm. The contours consist of (x,y) coordinates. These coordinates are clustered using the Kohonen self-organizing feature map (SOMF) [20], resulting in the codebook, as illustrated in figure 4.4.

The writer specific feature vector is calculated as follows. For each writer the connected component contours are extracted and using the Euclidean distance matches are made between the writer's contours and the codebook. A histogram is made of the amount of occurrences of the writer's contours that are in the codebook as well, the histogram is finally normalized. This results in a feature with the dimensionality of 900.

More details about the implementation of CO^3 can be found in [2].

4.2.2 Junclets

The second allographic-level feature used in this thesis is Junclets, which was first introduced in [8]. The algorithm used in this thesis is also based on [8]. For Junclets the handwriting images need to be preprocessed by binarizing the image, for this thesis Otsu's thresholding algorithm was used [19]. Junclets is based on the assumption that the manner in which strokes cross (or junctions) is characteristic for a writer. There are three types of junctions, L, Y and X-junctions. These three types of junctions are detected in the handwritten documents, without the need of segmentation. The generated codebook contains Kohonen 2D clustered junctions and the feature characterizing the individual writers is a probability distribution of the writer's junctions compared to the codebook. The resulting feature Junclets has the dimensionality of 900.

The exact details on Junclets can be found in [8].

5 Dimensionality reduction

There are a lot of methods for reducing dimensionality [11]. One of the most well-known of these dimensionality reduction methods is principal component analysis (PCA). The goal of PCA is to find a projection best representing the data by choosing directions that maximise the variance in the chosen training data and projecting the data onto a new coordinate system created using these directions.

The implementation of PCA in this thesis was done as follows. First N random samples of the data set were chosen. On these random samples the variance is calculated. This variance is found by a covariance matrix (see equation 5.1), of which the eigenvectors with the largest eigenvalues correspond to correlations in the data. In this thesis, the number of principal components are denoted by the free parameter $ndim$. The $ndim$ principal eigenvalues of the random samples can then be used to map all the data from the data set in this new projection of reduced dimensionality.

$$\begin{aligned} C(X) &= XX^T \\ &= [x_1 - \bar{x} \cdots x_y - \bar{x}] \begin{bmatrix} (x_1 - \bar{x})^T \\ \vdots \\ (x_y - \bar{x})^T \end{bmatrix} \end{aligned} \tag{5.1}$$

where x is the data set and $\bar{x}$ is the mean value of the data set.

6 Kernelization

A kernel function is the inner dot product between two vectors in a feature space [21]. The goal of kernel functions is to find non-linear relationships between two vectors.

In this thesis, three well-known kernel functions were evaluated. The Gaussian, Sigmoid and Polynomial kernel functions.
Gaussian kernel function:

$$K_G(x_i, z_j) = \exp(-\frac{||x_i - z_j||^2}{2\sigma^2}) \tag{6.1}$$

Sigmoid kernel function:

$$K_S(x_i, z_j) = \tanh(\kappa(x_i \cdot z_j) + \gamma) \tag{6.2}$$

Polynomial kernel function:

$$K_P(x_i, z_j) = (\kappa(x_i \cdot z_j) + \gamma)^\lambda \tag{6.3}$$

where x_i and z_j are two vectors. Each kernel function has besides the two vectors to be kernelized other parameters in order to fit the kernel function to the data. The Gaussian kernel has only one kernel for which the best value needs to be found for the data, σ. The Sigmoid kernel is a little more complex with two parameters in need of optimization, the κ and γ values. Polynomial is the most complex kernel function as it has three parameters, κ, γ and $lambda$ that need to be optimized. After optimization the resulting, kernelized vector is of size $length(x_i) * length(z_j)$.

7 Writer identification

For every query document q the distance was calculated to all the other documents in the data set. The documents were then sorted on distance measurements in ascending order. The first n documents comprised the *hit list*. The hit list is counted as correct if it contains a document written by the same writer. This gives the $top - n$ performance. For this thesis the typical hit list sizes of 1 and 10 were used.

In this thesis tree different distance algorithms were tested, χ^2, Euclidean and Manhattan, which were selected from more distance algorithms in an early stage of the research, as they gave the highest performances.
χ^2 distance:

$$\chi^2(p, q) = \frac{1}{2} * \sum_{i=1}^{k} \frac{(p_i - q_i)^2}{p_i + q_i} \tag{7.1}$$

Euclidean distance:

$$E(p, q) = \sqrt{\sum_{i=1}^{k} (p_i - q_i)^2} \tag{7.2}$$

Manhattan distance:

$$M(p, q) = \sum_{i=1}^{k} |p_i - q_i| \tag{7.3}$$

where p and q are two feature vectors between which the distance is calculated, k is the length

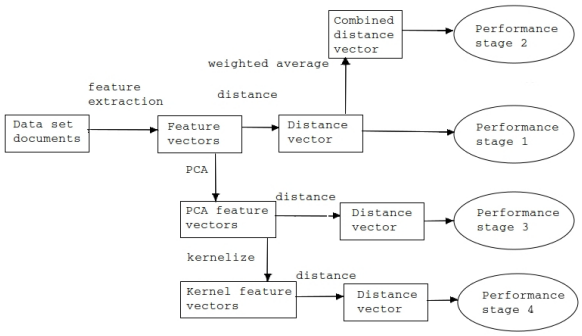

Figure 7.1: Pipeline of the methods used in this thesis.

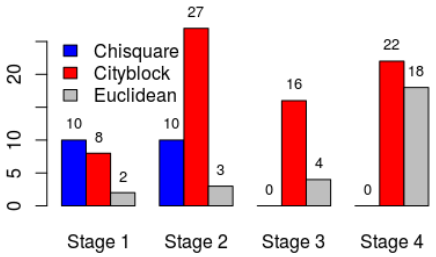

Figure 8.1: Histogram of best distance functions per stage. Stage 1 is the single feature, stage 2 is the linear combination, stage 3 is the PCA reduced feature and stage 4 is the kernelized reduced feature.

of the feature vector p and i gives the i^{th} element of the feature vectors.

In this paper first the feature vectors were extracted using the five features from section 4 on the four data sets described in section 3. To have a baseline to compare to, the performance of the feature vectors themselves was calculated using the three different distance algorithms at stage one.

At the second stage the feature vectors were combined linearly as done in previous papers. A weighted average was taken of the distance measurements of two features on the same data set, using the same distance algorithm. This average was computed as follows:

$$Av(\delta_1, \delta_2) = \lambda * \delta_1 + (1 - \lambda) * \delta_2 \tag{7.4}$$

where δ_1 and δ_2 are two distance measurements. More information about the distance functions can be found in [22].

The performance was then calculated using the new distance. This linear combination was done for all data sets, using all distance algorithms on all possible dual combinations between the five features (i.e. ten feature combinations). The best performances per combination per data set were used as comparison for the proposed non-linear combination.

At the third stage the feature vectors were reduced in dimensionality by applying PCA as described in section 5. For each feature on each data set the optimal combination of number of writers used and the new dimensionality was found. The best performances per feature per data set was compared to the same feature on the same data set of the single features in stage one.

At the final and fourth stage the reduced feature vectors were combined using the three different kernel functions described in section 6. The best parameter settings were found for each kernel and the PCA parameters that gave the best performance were used. The best performances of the combinations on the data sets were used as a comparison to the linear combination method.

Figure 7.1 illustrates the methods used.

8 Experimental results

8.1 Stage 1: single features

After the feature vectors were generated their performance was calculated on the four data sets using three different distance functions. Figure 8.1 shows for each stage the number of times a certain distance function (Chisquare, Cityblock or Euclidean) gave higher performances than the other distance functions. As can be checked, the total amount of stages 1 and 3 is 20 (each contained single features, so 5 features times 4 data sets) and the total amount of stages 2 and 4 is 40 (which contained combinations of features, 5 features in combinations of 2 is 10 combinations on 4 data sets).

Figure 8.1 illustrates for stage 1 that the Chisquare distance function often gave the highest performance, with the Cityblock distance function as close second. The Cityblock distance function is only preferred in two instances by the Junclets feature.

The resulting performances of the features can be found in table 8.1. The Firemaker and IAM data sets gave the best performances using the Chisquare distance function, the CERUG-CN data set had the best performances when applying the Cityblock feature and the CERUG-EN data set

Table 8.1: The performance of the different features in stage 1.

Feature	Firemaker		IAM		CERUG-EN		CERUG-CN	
	Top-1	Top-10	Top-1	Top-10	Top-1	Top-10	Top-1	Top-10
LBP	52.4	82.0	65.0	85.6	47.6	69.5	14.3	39.0
CC	35.4	70.8	32.9	67.2	38.6	78.6	6.7	16.7
Hinge	84.8	**95.8**	85.8	95.1	41.4	81.4	91.0	95.7
CO^3	61.2	78.4	77.8	91.3	76.2	95.7	83.3	94.3
Junclets	**86.0**	95.0	**88.2**	**96.3**	**93.8**	**97.6**	**93.3**	**98.1**

is in the middle with as much preference for the Chisquare as the Cityblock feature. The Euclidean feature was preferred least, which can also be seen in figure 8.1. The Junclets feature gives overall the highest performance and the Chain Code feature has the worst performance. The allograph level features perform better on CERUG-EN data set then the textural level features. On all other data sets the Hinge feature performs best after the Junclets feature. On the CERUG-CN data set the LBP and CC perform very poorly.

8.2 Stage 2: linear combinations

First the best λ values were found for the parameter for linear combination. Figure 8.2 can be seen as a histogram in the form of a line plot to show the trend. The y-axis, Count, shows the amount of times a certain value for λ gave higher performances than all the other values. This count is over all the data sets and all the feature combinations of two features. To find the best value for λ the range of 0.0 to 1.0 was tested using intervals of 0.05.

Figure 8.2 shows that the linear combination is not much of a combination. The most used and most preferred λ's were either close to 0.0 or close to $1, 0$, which means that the distance measurement of one feature has almost all the influence on the combined distance measurement. So the linear combinations are combinations in which one feature is very dominant and the other feature has very little influence.

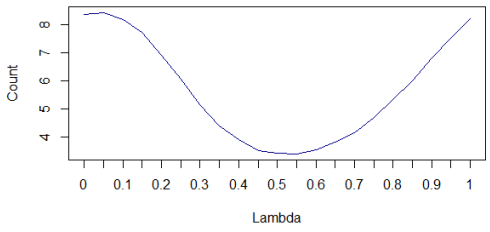

Figure 8.2: Usage of best λ values in linear combination (stage 2), smoothed using Gaussian filter.

Table 8.2: The performance of the linear combinations between the features, stage 2.

Feature combinations	Firemaker		IAM		CERUG-EN		CERUG-CN	
	Top-1	Top-10	Top-1	Top-10	Top-1	Top-10	Top-1	Top-10
LBP+CC	54.4	82.4	65.5	85.5	61.4	92.4	14.3	39.0
LBP+Hinge	85.6	96.0	86.6	95.3	65.2	94.8	92.4	98.1
LBP+CO^3	70.0	90.2	84.8	93.7	88.6	98.1	88.1	97.6
LBP+Junclets	86.6	95.4	88.5	96.2	96.2	**98.6**	**96.7**	**99.0**
CC+Hinge	85.4	95.4	85.8	95.1	67.1	92.4	91.4	96.2
CC+CO^3	65.4	88.4	82.6	93.2	87.6	97.1	83.8	94.3
CC+Junclets	86.6	94.6	88.5	96.2	**97.6**	98.1	93.8	98.1
Hinge+CO^3	86.2	96.0	91.2	96.6	81.9	96.7	94.8	98.6
Hinge+Junclets	**89.0**	**96.2**	91.1	**96.8**	94.3	97.6	96.2	98.1
CO^3+Junclets	87.2	95.6	**91.4**	**96.8**	94.3	97.6	95.2	98.1

Figure 8.1 shows that for the linear combinations (Stage 2) the Cityblock distance is preferred as distance function as it most often results in the best performances, Euclidean is again favoured least.

Table 8.2 shows the final performances of the combinations. For each combination on each data set the distance function and λ value was used that gave the highest performances, so these performances found in the table are the highest that can be reached using the method in this thesis. For the linear combination as could already be seen in figure 8.1 the preference has shifted. Every data set preferred Cityblock now instead of Chisquare for the highest performances. The IAM and CERUG-CN data sets never used the Euclidean distance to get a high performance. Linear combinations with Hinge, Junclets or CO^3 had a very strong preference for Cityblock over the other to distance functions. Linear combinations with LBP and CC preferred Cityblock as well, but also used Chisquare to get high results.

As can be seen from table 8.2, combinations with Junclets are the most powerful. Interestingly, where LBP was one of the weaker features in stage 1, especially on the CERUG data sets, in combination with Junclets it gives the best performance. For CERUG-EN this is peculiar, since the single features of stage 1 showed that the allographic level features performed much better than the texture level features, the expectation would be that the combination of allographic features would result in a higher performance than the combination with a texture level feature.

Most combinations indeed result in a higher performance together than each feature separately (table 8.1) as reported in previous research. Not all combinations give higher performances together than separately, for example the combination of Chain Code with Hinge and with Junclets on the Firemaker set gives a slightly lower Top-10 perfor-

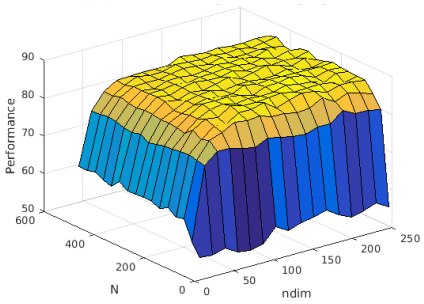

Figure 8.3: Effect of N and $ndims$ parameters of PCA on Top-1 performance of Hinge on the Firemaker data set using the Cityblock distance function.

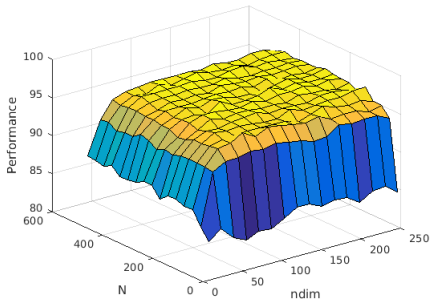

Figure 8.5: Effect of N and $ndims$ parameters of PCA on Top-10 performance of Hinge on the Firemaker data set using the Cityblock distance function.

mance than Hinge and Junclets separately.

8.3 Stage 3: PCA reduced feature vectors

Figures 8.3 and 8.5 show the influence of the N and $ndims$ parameters of PCA on the top-1 and top-10 performance on the Hinge feature respectively. The same trend was seen for most features on most data sets using both the Cityblock and Euclidean distance functions. The figures show that from a certain point on in the N and $ndims$ parameters a ceiling is reached and the performance does not vary much.

Figure 8.1 shows that the Cityblock distance function gives higher performances most of the time. The Chisquare distance function is not used at all since after PCA the feature vectors can contain negative numbers and the requirement for Chisquare is that the input data x, $x \geq 0$ [22].

Figure 8.4 shows the reduction in feature dimensionality for the best performances. The features with the highest dimensions (i.e. CO^3 and Junclets) have the highest reduction in feature size, up to 92%. The features with average dimensions (i.e. LBP and Hinge) are reduced less in size for optimal performances with lower reductions for Firemaker and IAM than the CERUG data sets. The CC feature was reduced in information by quite an amount of all data sets except for IAM, which is interesting since the CC feature only has a dimensionality of 9, which cannot contain much information, so any reduction results in a big loss in information.

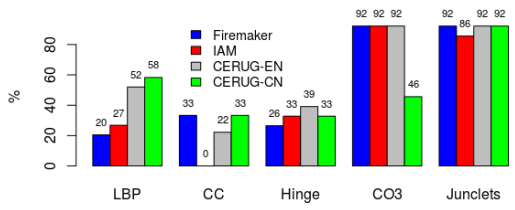

Figure 8.4: Percentage of reduction in dimension by PCA.

Table 8.3 shows the final performances of the PCA reduced features. For each reduced feature on each data set the distance function (i.e. Cityblock or Euclidean) in combination with the values for N and $ndim$ were used that gave the highest performances, so these performances are the highest that can be reached using the method in this thesis. The best N and $ndim$ values were very variable and dependent on the feature used in combination with the data set. The Junclets and CO^3 features did lie close in the best values for the different data sets. The best distance function was mainly influenced by the feature used, the CO^3 feature was the only one with a preference for the Euclidean distance and Junclets also preferred Euclidean once, on the CERUG-EN data set. Interestingly, that was the only set for which CO^3 had a higher performance using Cityblock.

Table 8.3 shows the final performances of the PCA reduced feature vectors, or stage 3. The results of the application of PCA are promising for the proposed method in this thesis, for most features have a slightly higher performance on the data sets than without the application of PCA in stage 1 (table 8.1). The performance of the Chain Code feature does decrease, this can be expected since the Chain Code feature already has a low dimension of 9, so reducing it even further means losing valuable information, resulting in a lower per-

Table 8.3: The performance of the different features after PCA in stage 3, "Dim" is the new dimensionality of the feature on the data set.

Feature	Firemaker			IAM			CERUG-EN			CERUG-CN		
	Top-1	Top-10	Dim	Top-1	Top-10	Dim	Top-1	Top-10	Dim	Top-1	Top-10	Dim
LBP	56.6	82.6	202	60.3	84.2	186	49.5	80.0	122	17.1	61.0	106
CC	34.0	71.2	6	27.8	65.0	9	35.7	78.1	7	5.7	16.7	6
Hinge	87.2	96.2	186	86.9	95.1	170	41.4	81.9	154	87.1	95.7	170
CO^3	61.4	85.6	70	72.5	88.9	70	64.8	89.0	70	80.5	94.8	490
Junclets	**90.6**	**97.8**	70	**89.6**	**97.0**	130	**93.8**	**98.1**	70	**94.3**	**98.1**	70

formance. The PCA reduction has the best effect on Junclets and increases the performance on each data set.
Comparing tables 8.2 and 8.3 show that the PCA reduced features of Junclets and Hinge perform better on the Firemaker data set than in linear combinations with other features. For the IAM and CERUG data sets it occurs less often that the PCA reduced features perform better single than the linear combination of the normal features.

8.4 Stage 4: kernelized feature vectors

For the kernels the optimal parameter values need to be found first. For the gaussian kernel there is one parameter that needs optimizing, σ, for the sigmoid kernel there are two parameters in need of optimization, κ and γ. The polynomial kernel lastly has three parameters to optimize, κ, γ and λ.

8.4.1 Gaussian kernel

Figure 8.6 shows an example of the influence of the σ parameter on the performance of a Gaussian kernel combination. As can be seen there is a clear preference for a certain value, for $\sigma = 0.001$.

Figure 8.7 shows the amount of times a certain value for the σ parameter was preferred over the other values for σ by giving a higher performance. The intervals used to find the best values for σ were set such that different place values of decimal numbers were tested, from tens to hundred thousandths. Figure 8.7 shows that most other combinations of features by contrast have the best performance with $\sigma = 0.01$.

8.4.2 Sigmoid kernel

Figure 8.8 shows an example of the influence of the κ and γ parameters on the performance of a sigmoid kernel combination. As can be seen there is a high variability and not a clear preferable combination of parameters or even a preference for a single parameters.

Figure 8.10 shows the amount of times a certain value for the κ or γ parameter was preferred over the other values for κ or γ by giving a higher performance. The intervals used to find the best values for κ were set such that different place values of decimal numbers were tested, from thousands to millions. The values for γ were chosen to see what kind of value would give higher performances, a positive or negative number, a whole number of fractions. Figure 8.10 shows that there is not a preferable

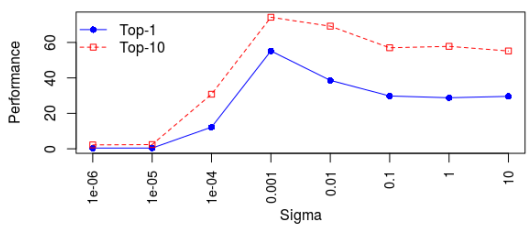

Figure 8.6: Influence of σ parameter on gaussian kernel applied on LBP+Hinge and the Firemaker data set.

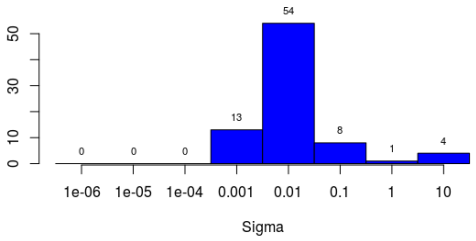

Figure 8.7: Histogram of σ resulting in best performances of gaussian kernel. The peak lies at $\sigma = 0.01$, which gave the best performances 54 times.

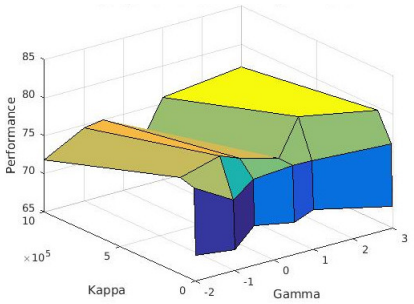

Figure 8.8: Top-1 influence of κ and γ parameters on sigmoid kernel applied on LBP+Hinge and the Firemaker data set.

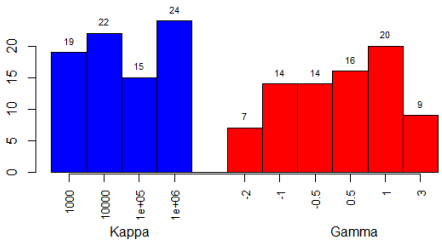

Figure 8.10: Histogram of κ and γ resulting in best performances of sigmoid kernel.

value resulting in the highest performances as well. The best parameter combination depends on the features that are combined and the data set used.

8.4.3 Polynomial kernel

Parameter optimization tests on polynomial showed that the second parameter, γ has no influence on the performance. Figure 8.9 shows an example of the influence of the κ and λ parameters on the performance of a polynomial kernel combination. For this combination it is clear that $\kappa = 0.5$ results in higher performances but that the λ results in variable performances with no clear preference. The same trend can be seen in figure 8.11

Figure 8.11 shows the amount of times a certain value for the κ or λ parameter was preferred over the other values for κ or λ by giving a higher performance. γ is not shown here since as said before, it has no influence on the performance, so no certain value was preferred over other values. The intervals used to find the best values for κ were set at 0.5 and 1 because earlier tests showed that there was a different in performance only between higher than 1 or lower than 0.5, the exact values did not matter, they stayed the same above or below 1 and 0.5. The values for λ were chosen such that different polynomial functions were created, the negative values for fractions under one, fractions for (square)roots and positive numbers for different orders of polynomial functions. Figure 8.11 shows that there is a variability in λ for the values that result in the highest performances. The best value for λ depends on the features that are combined and the data set used. κ however does show a difference in preference, with almost $\frac{2}{3}$ preferring $\kappa = 0.5$ and $\frac{1}{3}$ preferring $\kappa = 1.0$ to get the highest performances.

8.4.4 Resulting performances

Figures A.1 and A.2 (see Appendix A) show the best performances of the three kernels on the feature combinations on four data sets. The gaussian kernel results in much lower performances for the Firemaker and IAM data sets. In the CERUG-EN and CERUG-CN the performance of the gaussian kernel lies closer to the other two kernels and is even higher than the other two for the combination of Hinge and CO^3 on CERUG-EN and higher than polynomial in the Hinge+Junclets combination on

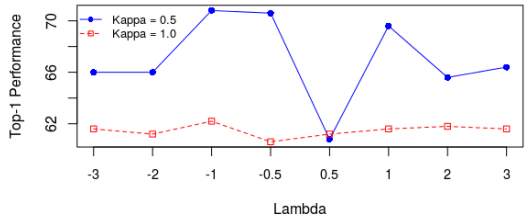

Figure 8.9: Top-1 influence of κ and λ parameters on polynomial kernel applied on LBP+Hinge and the Firemaker data set.

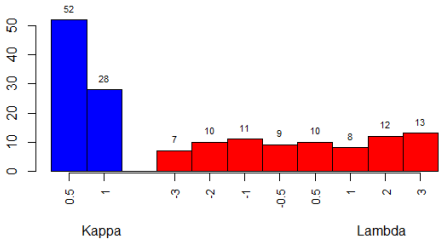

Figure 8.11: Histogram of κ and λ resulting in best performances of polynomial kernel.

the CERUG-CN data set. Nonetheless, the polynomial and sigmoid kernels are overall clearly better suited for these features and data sets. The polynomial and sigmoid kernels also follow the same trends for the data sets, indicating the strength of the different feature combinations on those data sets.

Figure 8.1 shows that there is not much of a difference in the usage of the Cityblock or Euclidean distance functions for the non-linear combination. Again the Chisquare distance is not used since the kernels can result in feature vectors in which not all elements are positive numbers.

Table 8.4 shows the best performances of the non-linear combinations using kernels. For each combination on each data set the parameters for PCA were used that gave the highest performance for the features on that particular data set. The distance function, kernel function and parameters accompanying the kernel were used that gave the highest performances, so these final performances in the table are the highest performances that can be reached using the non-linear combination method in this thesis. The Firemaker and CERUG-CE data sets both prefer the Cityblock distance, howeve rthe IAM and CERUG-CN data sets prefer the Euclidean distance. Combinations with CC, Hinge or Junclets have a preference for one distance function, however combinations with LBP and CO^3 have no preference.

As shows in figures A.1 and A.2 the Gaussian kernel has the highest performances only twice, on the CERUG-CN data set and each time in combination with the CO^3 feature. For all other data sets and features there was a preference for the Sigmoid kernel over the Polynomial kernel except for the CC feature, which did not have a preference for either.

Table 8.4 shows that there are no longer combinations with high peaks in performance, the performances on each data set lie quite close together. The best combinations on the data sets are combinations with Junclets.

When comparing tables 8.1 and 8.4 it can be seen that for Firemaker, IAM and CERUG-EN the non-linear combination of the features often result in higher performances than the single features, though not always in combinations with Junclets. Even so, for the CERUG-CN data set the non-linear combination of the features often results in lower performances than the Hinge, CO^3 and Junclets features themselves.

Comparing tables 8.3 and 8.4 show that non-linear combinations with Hinge and Junclets result in lower performances than the PCA reduced Hinge and Junclets themselves, only the other features sometimes benefit from the non-linear combination, but overall the non-linear combination of the features results in lower performances than the PCA reduced features separately.

8.5 Non-linear combination vs linear combination

Comparing tables 8.2 and 8.4 it can be seen that overall the non-linear combination of the features has quite lower performances than the linear combination of the features. Only the combination of LBP with CC benefits from the non-linear combination and results in much higher performances using the non-linear combination than the linear combination. However, the other feature combinations suffer from the non-linear combination in comparison to the linear combination, especially on the CERUG data sets where the difference in performance can be up to 50%. For the IAM and CERUG-CN data sets the same combination results in the highest performance for both the linear combination and the non-linear combination, yet, for IAM the top-1 and top-10 performance of the non-linear combination is 3.8% and 2% lower, respectively. For CERUG-CN the top-1 and top-10 performance of the non-linear combination is 8.1% and 2.3% lower, respectively. For the Firemaker and CERUG-EN data sets a shift has occurred in best combinations for the non-linear and linear combinations, though both are still combinations with Junclets.

Table 8.4: The performance of the non-linear combinations between the reduced features, stage 4.

Feature	Firemaker		IAM		CERUG-EN		CERUG-CN	
combinations	Top-1	Top-10	Top-1	Top-10	Top-1	Top-10	Top-1	Top-10
LBP+CC	73.8	89.4	70.6	87.0	66.7	91.9	39.5	61.0
LBP+Hinge	81.6	90.8	85.0	92.2	79.0	91.4	83.3	93.8
LBP+CO^3	64.2	75.8	72.2	84.6	71.0	87.6	55.7	78.6
LBP+Junclets	85.0	90.4	84.6	91.5	**93.3**	**96.7**	**88.6**	**96.7**
CC+Hinge	80.8	90.8	82.2	92.5	64.8	81.9	43.8	68.1
CC+CO^3	62.8	85.8	70.1	83.3	64.3	87.6	27.1	38.1
CC+Junclets	85.6	**94.8**	85.6	**94.8**	83.8	93.3	78.6	91.0
Hinge+CO^3	65.6	82.0	76.8	87.2	60.5	73.8	47.6	77.6
Hinge+Junclets	85.6	93.4	85.3	92.5	76.7	86.7	79.0	90.0
CO^3+Junclets	**87.6**	**94.8**	**87.6**	**94.8**	86.2	95.7	65.7	88.6

Table 8.5: Writer identification performances on the Firemaker and IAM data sets.

Approach	Firemaker			IAM		
	Writers	Top-1	Top-10	Writers	Top-1	Top-10
Bulacu and Schomaker [4]	250	83	95	650	89	97
Siddiqi and Vincent [6]	-	-	-	650	89	97
A.A. Brink et al. [5]	251	86	97	657	97	98
He and Schomaker [23]	250	90.4	98.2	650	93.2	97.2
Y. Hannad et al. [18]	-	-	-	657	89.54	96.77
S. He et al. [8]	250	89.8	96.0	650	91.1	97.2
Proposed	250	87.6	94.8	650	87.6	94.8

Table 8.6: Writer identification performances on the CERUG data sets.

Approach	CERUG-EN			CERUG-CN		
	Writers	Top-1	Top-10	Writers	Top-1	Top-10
S. He et al. [8]	105	89.5	97.6	105	94.2	97.1
Proposed	105	93.3	96.7	105	88.6	96.7

8.6 Comparison to previous studies

Tables 8.5 and 8.6 show the best performances reached in other research. For Firemaker the in this thesis proposed method does not have a poor performance compared to the other research, especially the top-1 performance is not that bad. Though, this difference in performance is also due to the features used in previous research. For the IAM and CERUG-CN data sets the proposed method has quite a lower performance than previous research. On the CERUG-EN data set the top-1 performance of the non-linear combination is quite higher than the performance reached in [8], however, the top-10 performance is lower.

9 Conclusions

In this thesis a new method for combining features for writer identification was introduced. Instead of linearly combining the features by taking the (weighted) average of the distances between different features the feature vectors were first reduced in dimension using PCA and then non-linearly combined by applying a kernel function. The results show that this proposed non-linear combination method achieved much worse performance than the linear combination. All three of the tested kernel functions gave much lower performances than the linear combination. This gives the conclusion that non-linear combination is not preferred over linear combination and that for future research it is advised to use linear combinations.

Nonetheless, what this study showed was that reducing the dimensionality of features of medium to high dimensionality using for instance PCA is very promising. Not only are the features reduced a lot in dimensionality, the resulting performances are actually better than the features without PCA applied. So when in future research a new feature is introduced that has a medium to high dimensionality it is advisable to try dimensionality reduction as part of the feature to extract the most relevant information, which also makes the feature less time consuming to work with.

For future research it could therefore be interesting to explore the application of dimensionality reduction on different features using different methods and to see how this influences the performance. It could also be interesting to look at linear combinations of the reduced features and see how the reduction influences the linear combinations.

References

[1] C. Halder, S. M. Obaidullah, and K. Roy, "Offline writer identification and verification—a state-of-the-art," in *Information Systems Design and Intelligent Applications: Proceedings of Third International Conference INDIA 2016, Volume 3* (C. S. Satapathy, K. J. Mandal, K. S. Udgata, and V. Bhateja, eds.), (New Delhi), pp. 153–163, Springer India, 2016.

[2] L. Schomaker and M. Bulacu, "Automatic writer identification using connected-component contours and edge-based features of uppercase western script," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, pp. 787–798, June 2004.

[3] R. Plamondon and S. N. Srihari, "Online and off-line handwriting recognition: a comprehensive survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 63–84, Jan 2000.

[4] M. Bulacu and L. Schomaker, "Text-independent writer identification and verification using textural and allographic features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, pp. 701–717, April 2007.

[5] A. Brink, J. Smit, M. Bulacu, and L. Schomaker, "Writer identification using directional ink-trace width measurements," *Pattern Recognition*, vol. 45, no. 1, pp. 162 – 171, 2012.

[6] I. Siddiqi and N. Vincent, "Text independent writer recognition using redundant writing patterns with contour-based orientation and curvature features," *Pattern Recognition*, vol. 43, no. 11, pp. 3853 – 3865, 2010.

[7] M. Bulacu, L. Schomaker, and L. Vuurpijl, "Writer identification using edge-based directional features," in *Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on*, pp. 937–941, Aug 2003.

[8] S. He, M. Wiering, and L. Schomaker, "Junction detection in handwritten documents and its application to writer identification," *Pattern Recognition*, vol. 48, no. 12, pp. 4036 – 4048, 2015.

[9] I. Siddiqi and N. Vincent, "A set of chain code based features for writer recognition," in *2009 10th International Conference on Document Analysis and Recognition*, pp. 981–985, July 2009.

[10] M. Bulacu and L. Schomaker, "Combining Multiple Features for Text-Independent Writer Identification and Verification," in *Tenth International Workshop on Frontiers in Handwriting Recognition* (G. Lorette, ed.), (La Baule (France)), Université de Rennes 1, Suvisoft, Oct. 2006.

[11] L. Van Der Maaten, E. Postma, and J. Van den Herik, "Dimensionality reduction: a comparative," *J Mach Learn Res*, vol. 10, pp. 66–71, 2009.

[12] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[13] A. Schlapbach, *Writer Identification and Verification*. Amsterdam, The Netherlands, The Netherlands: IOS Press, 2008.

[14] L. Schomaker, M. Bulacu, and K. Franke, "Automatic writer identification using fragmented connected-component contours," in *Frontiers in Handwriting Recognition, 2004. IWFHR-9 2004. Ninth International Workshop on*, pp. 185–190, Oct 2004.

[15] A. Schlapbach, V. Kilchherr, and H. Bunke, "Improving writer identification by means of feature selection and extraction," in *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*, pp. 131–135 Vol. 1, Aug 2005.

[16] L. Schomaker and L. Vuurpijl, "iuf firemaker: A benchmark data set for writer identification," tech. rep., Nijmegen Institute for Cognition and Information, University of Nijmegen, The Netherlands, 11 2000.

[17] U.-V. Marti and H. Bunke, "The iam-database: an english sentence database for offline handwriting recognition," *International Journal on Document Analysis and Recognition*, vol. 5, no. 1, pp. 39–46, 2002.

[18] Y. Hannad, I. Siddiqi, and M. E. Y. E. Kettani, "Writer identification using texture descriptors of handwritten fragments," *Expert Systems with Applications*, vol. 47, pp. 14 – 22, 2016.

[19] N. Otsu, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, no. 285-296, pp. 23–27, 1975.

[20] T. Kohonen, *Self-organization and associative memory*, vol. 8. Springer Verlag, 2 ed., 1988.

[21] J. Shawe-Taylor and N. Cristianini, *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[22] B. McCune and J.Grace, *Analysis of Ecological Communities*. MjM Software Design, Gleneden Beach, Oregon, 2002.

[23] S. He and L. Schomaker, "Delta-n hinge: rotation-invariant features for writer identification," in *2014 22nd International Conference on Pattern Recognition (ICPR)*, pp. 2023–2028, IEEE, 2014.

A Appendix

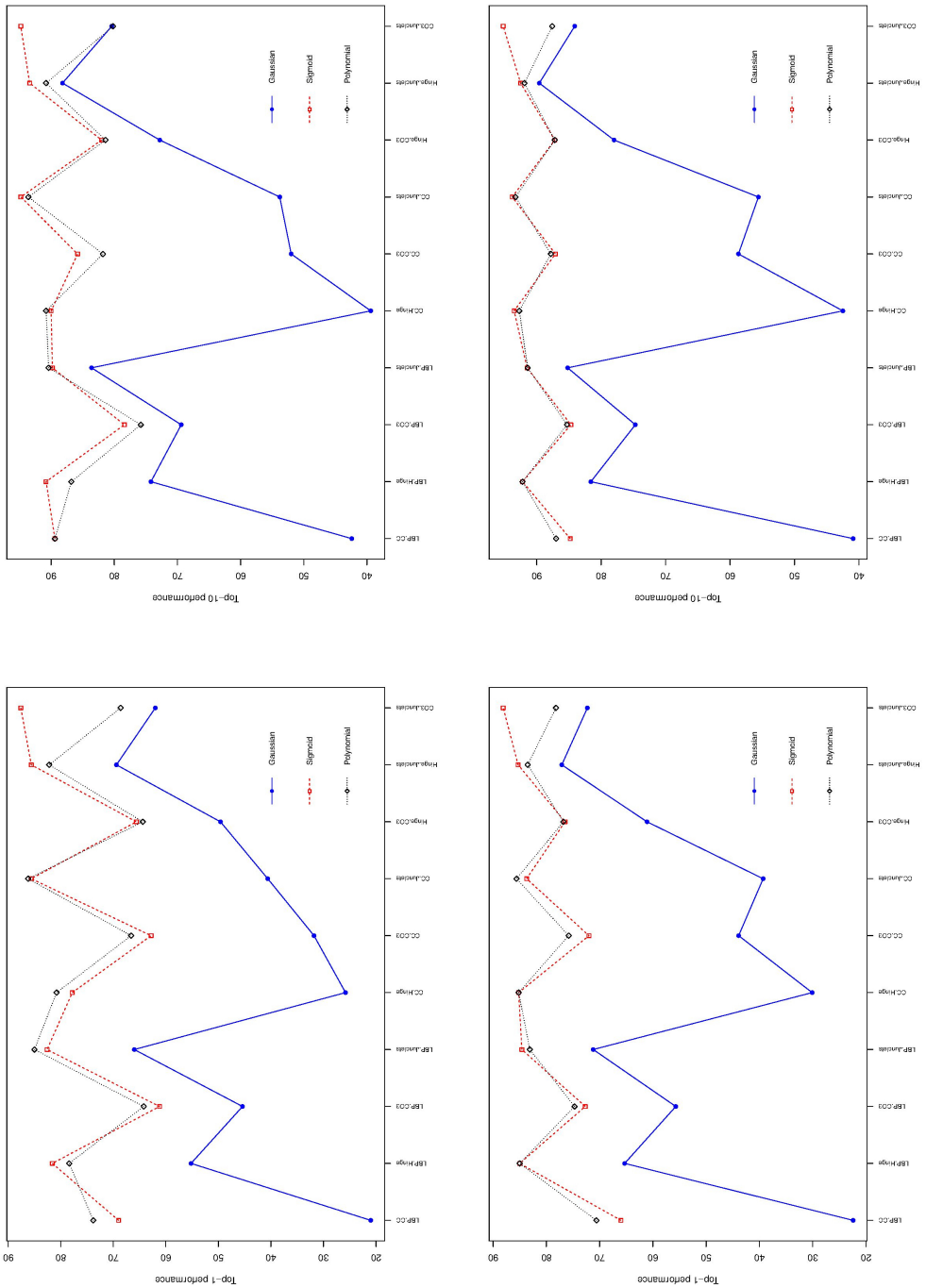

Figure A.1: Top-1 and top-10 performances of the gaussian, sigmoid and polynomial kernels on the different feature combinations and data sets.

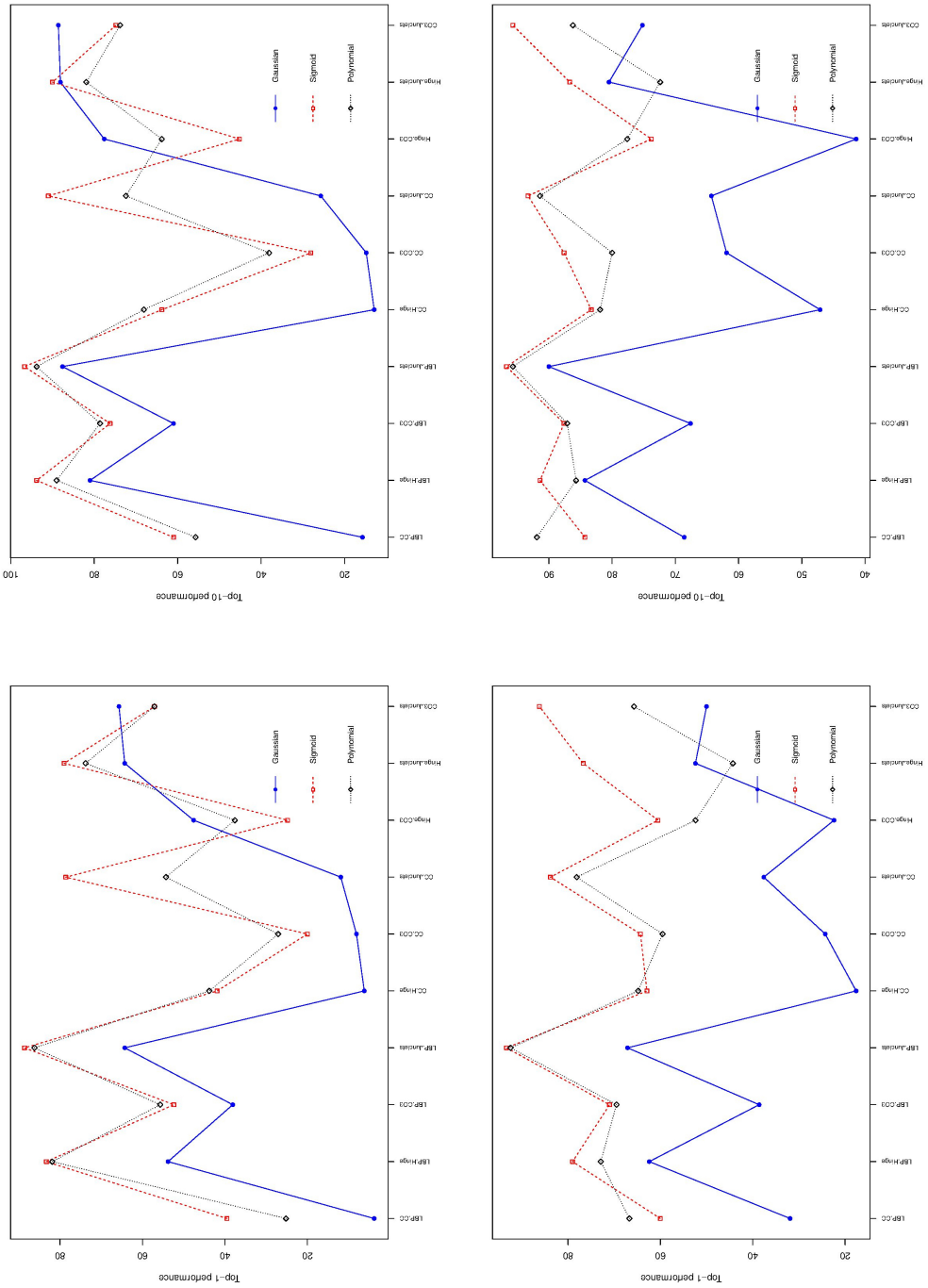

Figure A.2: Top-1 and top-10 performances of the gaussian, sigmoid and polynomial kernels on the different feature combinations and data sets.