ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.30, No-1-2, 2022

# Magnitude of Semicircle Tiles in Fourier-space - A Handcrafted Feature Descriptor for Word Recognition using Embedded Prototype Subspace Classifiers

Anders Hast[1,2]
[1] Department of Information Technology
Uppsala University
Centre for Image Analysis
SE-751 05 Uppsala, Sweden
anders.hast@it.uu.se
[2] The Swedish Institute for Children's Books
SE-113 22 Stockholm, Sweden

## ABSTRACT

The purpose of this paper is to in detail describe and analyse a Fourier based handcrafted descriptor for word recognition. Especially, it is discussed how the Variability in the results can be analysed and visualised. This efficiency of the descriptor is evaluated for the use with embedded prototype subspace classifiers for handwritten word recognition. Nonetheless, it can be used with any classifier for any purpose. An hierarchical composition of discrete semicircles in the Fourier-space is proposed and it will will be show how this compares to Gabor filters, which can be used to extract edges in an image. In comparison to Histogram of Oriented Gradients, the proposed feature descriptor performs better in this scenario. Compression using PCA turns out to be able to increase both the $F_1$-score as well as decreasing the Variability.

## Keywords

Discrete Fourier Transform, Gabor Filters, Subspaces, Embedded Prototypes, Clustering, $F_1$ score, Variability, Deep Learning, t-SNE.

## 1 INTRODUCTION

In recent times, *Embedded Prototype Subspace Classification* (EPSC) [HV21, HLV19, HL20, HV21] has proven to be able to classify datasets of various kinds, containing everything from single digits, characters to whole words, and even objects. Datasets used have been the MNIST dataset of handwritten digits [LCB10], E-MNIST containing letters [CATvS17], the Kuzushiji-MNIST dataset containing Japanese handwritten characters [CBK*18], the Fashion MNIST (F-MNIST) [XRV17] containing small images of clothes and accessories and a recently published dataset [HV21] based on the Esposalles dataset [RFS*13], where 30 different words were extracted to create an imbalanced dataset with a total of 16354 word images. This latter one will be used for the subsequent analysis in this paper.



The main contributions of this paper are as follows. First of all the Fourier based handcrafted feature descriptor (previously called mFFT) used in [HLV19, HL20, HV21] will for the first time be described and analysed in detail. Furthermore, an hierarchical composition of discrete semicircles in the Fourier-space is proposed and it will be shown how this compares to Gabor filters, which can be used to extract edges of different orientations and sizes in an image [MNR92]. This more elaborate feature descriptor will subsequently be called: *Magnitude of Semicircle Tiles in Fourier-space*, or MoSTiF for short.

Moreover, it will be shown how the EPSC can be optimised to use the proposed feature descriptor for fast matching. And last but not least, it will be shown that the when performing bootstrapping on a dataset, varying the size of the split between learning and validation partitions, the standard deviation (SD) of $F_1$ score can be a useful for understanding the behaviour of the classifier. The word *Variability* is often used as a synonym to SD, but here it will subsequently be used to denote the SD of the $F_1$ score in particular.

## 2 BACKGROUND

The main advantage of EPSC compared to many deep learning based methods [Sha18] for handwritten text recognition [KDJ18, DKMJ18, SF16] is that EPSC is shallow to its nature, with no hidden layers, and therefore does not require powerful GPU resources in the training process. In general, EPSC learns from the embedding of feature vectors, using dimensionality reduction techniques like t-SNE, UMAP or SOM [HV21] and then creates so-called subspaces from each cluster [KLR*77], which are a set of neurons specialised on identifying the class variation captured in that cluster. Obviously, EPSC does not always outperform the state-of-the-art deep learning approaches when it comes to accuracy. However, both learning and inference will generally be much faster due to its simplicity and compactness. Moreover, both the learning and classification processes are inherently easy to interpret [Kri19, CPC19], explain [ADRS*19, GSC*19, CPC19], and visualise.

The EPSC uses handcrafted features as input, while deep learning approaches such as Convolutional Neural Networks (CNN) have been efficiently used to extract learned features from images [SSTF*15, ZK15]. One drawback with learned features is the time consuming learning process. Since it has been noticed that CNN's produce Gabor-like features some efforts have been done to replace the CNN with Gabor filters [LCZ*18, JJL07]. It will be shown that this idea can be utilised by creating a hierarchical composition of discrete semicircles in the Fourier-space.

Several handcrafted features have been proposed in the literature, and some popular methods include Scale Invariant Feature Transform (SIFT) [Low04], Speeded Up Robust Features (SURF) [BETVG08] and Histograms of Oriented Gradients (HOG) [DT05], where some have been used in recognition systems [GDDM14]. Fourier based detectors can be constructed by taking the magnitude of the lowest frequency elements of signals [HV18, HSSK18]. Matuszewski et al. compute the magnitude from a few elements close the the centre of the shifted Fourier transform [MHWS17] while Buchholz and Jug [BJ21], create what they call a Fourier Domain Encoding (FDE) by computing normalised amplitude and phase of half the concentric Fourier rings. Herein, a mix of these two methods is proposed, by computing the magnitude of neighbouring Fourier semicircles. This is basically what was used previously for mFFT in [HLV19, HV21] together with HOG. However, here is proposed the extension of a hierarchy of image partitions making the feature vector more effective, which can be used without HOG. Since HOG has proven to be both simple and effective, the proposed descriptor will be compared to HOG as a reference.

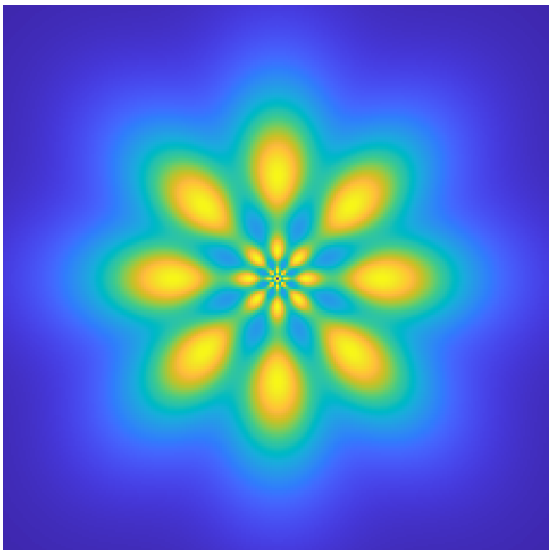


Figure 1: The Gabor filter banks in Fourier-space. The symmetric Gabor filter bank is distributed in four orientations and three frequency bands in this example.

## 3 THE MAGNITUDE OF SEMICIRCLE TILES IN FOURIER-SPACE

In this section the Magnitude of Semicircle Tiles in Fourier-space (MoSTiF) feature descriptor is introduced. The idea comes from the fact that CNN's create something similar to what Gabor filters does, which can be used to detect lines in different directions and frequencies. These are then combined in subsequent levels to more complicated features. In the EPSC this is done using the subspaces. Subspace classification is done by computing the norm of the projected feature vector to be classified into each subspace. The subspace of a certain class yielding the largest norm will tell what class the feature vector most likely belongs to. Subspace classification will be explained more in detail later.

As shown in Figure 1, a Gabor filter bank can be created by using Gaussians in a symmetric fashion. Each pairwise filter (placed diametrically with respect to the centre) corresponds to an orientation of the features. The further away from the centre the pairs are placed, the higher the frequency. Hence, only the innermost filters are of interest since they detect shape, while higher frequencies corresponds to very fine details or noise. Since subspaces are used to determine the more complicated features, elements are simply picked in a space filling [BHB16] semicircle and the magnitude of the complex Fourier value of the Discrete Fourier Transform (DFT) is computed as:

$$|\mathscr{F}[f(n)]| = \sqrt{\Re(\mathscr{F}[f(n)])^2 + \Im(\mathscr{F}[f(n)])^2}. \quad (1)$$

where $f(n)$ is the image $f$ at point $n$.

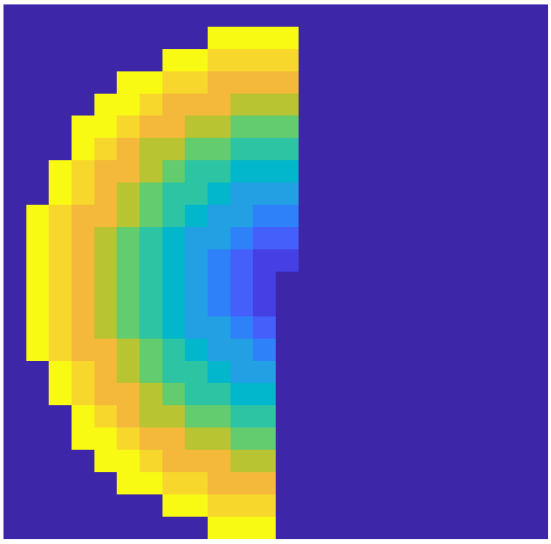


Figure 2: The different space filling semi-circles are shown in different colours.

Remember that the elements are diametrically distributed and therefore only half of the space filling circle is needed to construct a feature vector. Hence, a semicircle can be regarded as sampling the filter banks for a thin band of frequencies in all possible directions.

Figure 2 shows the possible space filling semi-circles for a $24 \times 24$ image patch. Note that the central pixel is not used since it corresponds to the DC content of the image, i.e. the overall brightness.

Computing one set of semi-circles on the whole image would not be efficient enough. Therefore the semicircles are also computed on different partitionings of the image in a hierarchical manner and then combined into a longer feature vector. Figure 3 shows three different combinations of partitions. The number above each sub-partition shows the number of semicircles used. The number to the right shows the total feature vector length.

By choosing the size $120 \times 120$ the image width and/or height can be evenly divided by $2, 3, 4, 5$ and 6. This gives 36 different possible block partitions of the input image. Each semicircle is concatenated in order to construct each part of the feature vector, which is subsequently normalised for each block in the partition. When all blocks and partitions are computed the final feature vector is also normalised.

## 3.1 Subspace Classification

Since it was first proposed by Watanabe et al. [WP73] in 1967, Subspaces have been used for classification in pattern recognition. This approach was later further developed by Kohonen and others [WLK*67, KLR*77, KO76, KRMV76, OK88]. In general, subspace classification can be regarded as a two layer neural network [HLV19, OK88, Laa07], where the weights are not learned using time consuming backpropagation. Instead weights are mathematically defined through Principal Component Analysis (PCA) [Laa07]. (Note that PCANet [CWW15] also set weights using PCA. However, this is done for features in a sliding window manner and is therefore fundamentally different from Subspace classification.) Last but not least, one important advantage is that the whole learning process can easily be visualised, since it is based on visualisation techniques (e.g. t-SNE), which makes it easy to both understand, interpret and explain, as compared to most of the state-of-the-art deep learning approaches, which are often regarded as black boxes.

Every image to be classified is represented by a feature vector $\mathbf{x}$ with $m$ real-valued elements $\mathbf{x}_j = \{x_1, z_2 ... x_m\}, \in \mathbb{R}$, such that the operations take place in a $m$-dimensional vector space $\mathbb{R}^m$. Any set of $n$ linearly independent basis vectors $\{\mathbf{u}_1, \mathbf{u}_2, ... \mathbf{u}_n\}$, where $\mathbf{u}_i = \{w_{1,j}, w_{2,j} ... w_{m,j}\}, w_{i,j} \in \mathbb{R}$, which can be combined into an $m \times n$ matrix $\mathbf{U} \in \mathbb{R}^{m \times n}$, span a subspace $\mathscr{L}_\mathbf{U}$

$$\mathscr{L}_\mathbf{U} = \{\mathbf{x} | \mathbf{x} = \sum_{i=1}^{n} \rho_i \mathbf{u}_i, \rho_i \in \mathbb{R}\} \tag{2}$$

where,

$$\rho_i = \mathbf{x}^T \mathbf{u}_i = \sum_{j=1}^{m} x_j w_{i,j} \tag{3}$$

Classification of a feature vector can be performed by projecting $\mathbf{x}$ onto each and every subspace $\mathscr{L}_{\mathbf{U}_k}$. The vector $\hat{\mathbf{x}}$ will in this way be a reconstruction of $\mathbf{x}$, using $n$ vectors in the subspace through

$$\hat{\mathbf{x}} = \sum_{i=1}^{n} (\mathbf{x}^T \mathbf{u}_i) \mathbf{u}_i \tag{4}$$

$$= \sum_{i=1}^{n} \rho_i \mathbf{u}_i \tag{5}$$

$$= \mathbf{U}^T \mathbf{U} x^T \tag{6}$$

By normalising all the vectors in $\mathbf{U}$, the norm of the projected vector can be simplified as

$$||\hat{\mathbf{x}}||^2 = (\mathbf{U}x^T) \cdot (\mathbf{U}x^T) \tag{7}$$

$$= (\mathbf{U}x^T)^2 \tag{8}$$

$$= \sum_{i=1}^{n} {\rho_i}^2 \tag{9}$$

Therefore, the feature vector $\mathbf{x}$, which is most similar to the feature vectors that were used to construct the subspace in question $\mathscr{L}_{\mathbf{U}_k}$, will therefore have the largest norm $||\hat{\mathbf{x}}||^2$.

## 3.2 Parameters to learn

As mentioned earlier, subspaces do not require learning through backpropagation, since the learning itself is

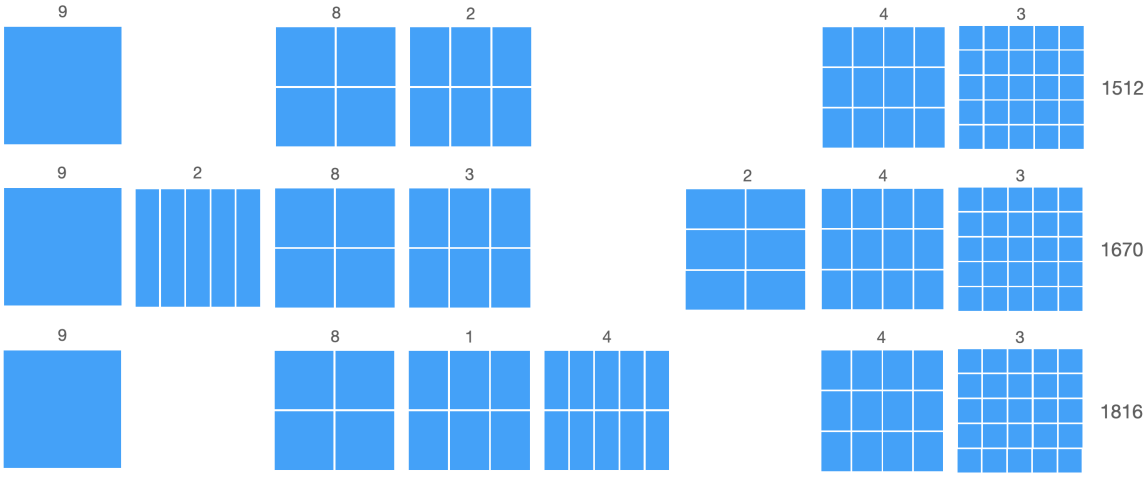



Figure 3: Three different feature vectors and their different partitionings are shown, one per row. The number of semicircles are denoted above each partition and the feature length is reported to the right.

done by the embedding obtained from some dimensionality reduction method such as t-SNE [MH08], UMAP [MH18] or SOM [Koh82]. Moreover, all the weights in the resulting neural network are set mathematically by PCA.

Nevertheless, bootstrapping can be used to evaluate and set parameters that are required for the overall performance. Such parameters are the feature detector itself, i.e. how many semicircles are to be used for different partitions. So far we have not devised a technique for doing that. Instead a Monte Carlo sampling approach was used to find the best parameters. The draw back is of course that this can be time consuming. Nonetheless, we think that the proposed partitionings shown in Figure 3 can be used for any dataset of handwritten words.

The subspace projection itself is done using only 6 dimensions, and it has been noted that this can be varied to improve performance when the size of the dataset to learn from is changed. Moreover, the effectiveness of PCA compression of the feature vector can also be evaluated as will be shown later herein.

### 3.3 Improved Embedding and Clustering

The idea of EPSC [HV21, HLV19] is to use some embedding technique to obtain prototypes for the construction of each subspace. In this work t-SNE [MH08], was used to reduce the number of dimensions of high dimensional data down to 2 dimensions. In this process, clusters are formed since t-SNE strives to move similar features (represented by their projected points) closer to each other and dissimilar points are kept further away from each other.

Previously, Hast et al. [HLV19] used kernel density estimation (KDE) [CHTT96] and watershed transform on the inverse image to find clusters in a two-dimensional image space, which is basically the same as performing

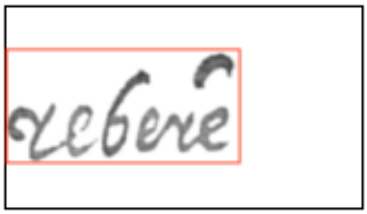


Figure 4: The placement of each word in the $90 \times 160$ rectangular bounding box. The background is removed but the word is not binarised. And the red rectangle shows how the word is cut out and then resampled to $120 \times 120$.

the Mean-Shift [CM02, FH75]. However, it was shown that other algorithms that requires specifying the exact number of clusters, such as K-means [HW79] could also be used [HV21]. Moreover, it was shown that instead of using a certain bandwidth for clustering, better performance was achieved by computing $k$ clusters, striving for these clusters to contain a certain predefined number of features $n_f$. Herein it was chosen to use a fixed $k$ instead. In fact $k = 2$ was chosen as it gave the best overall classification for this dataset (these initial experiments are not reported herein, since there are reasons to believe that $k$ depends on the data at hand and possibly also the amount of data). Furthermore, the process was simplified by replacing the inverse watershed with a gradient ascend method working on each data point instead of each pixel in the KDE image. Both these changes speeded up the learning process noticeably.

Another improvement, which gave an overall higher $F_1$ score, is shown in Figure 4, where each word image is cut out (red bounding box) and extracted from the image and resampled to the size $120 \times 120$..

# 4 EVALUATION AND METHOD

Opitz and Burst [OB21] show that the best choice to compute the $F_1$ scores for imbalanced datasets, is the arithmetic mean over harmonic means. I.e. $F_1$ scores are computed for each class and then averaged via arithmetic mean, such that

$$\mathbb{F} = \frac{1}{n}\sum_{i=1}^{n} F_1 = \frac{1}{n}\sum_{i=1}^{n} \frac{2P_iR_i}{P_i+R_i} \tag{10}$$

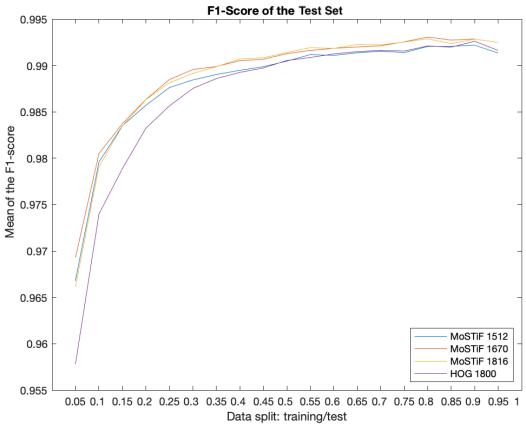
where $P_i$ and $R_i$ are precision and recall respectively for each class $i$.

It was chosen to compute the $F_1$ scores using the Bootstrapping method [Koh95, KW96], with stratification because the data is imbalanced. This means that the bootstrap sample is taken from the original set by using *sampling with replacement*, and that both the learning, test and validation sets are forced, as much as possible, to contain a certain percentage of each class. The validation set was kept the same throughout the experiments, using 50% of the available data. The remaining 50% was split into a set for training and a set for testing. The experiments where conducted 200 times for each data split, varying the permutations randomly, in order to be able to analyse the impact of what data are in each split, i.e the training and test set. By varying the percentage used for training, the change in $F_1$ scores could be analysed and parameters could be learned and set accordingly, as previously discussed. Moreover the SD of the $F_1$ score for each run was computed and is being called *Variability* in the subsequent presentation of the results. In any case, varying the split gives a chance to analyse the impact on the overall performance of the feature vectors in combination with EPSC.

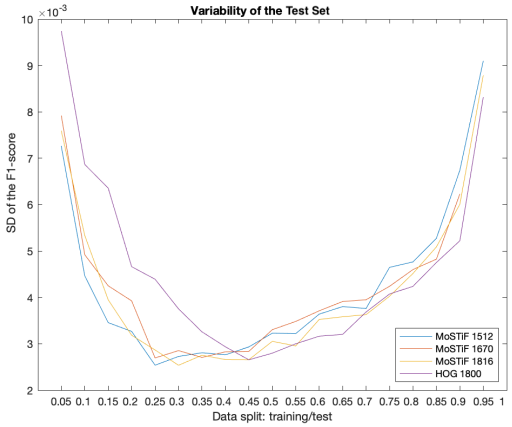
# 5 RESULTS

Figure 5 shows the $F_1$ score 5a and Variability 5b, respectively for the training set. It can be noted that HOG performs much better when the training set becomes larger. The U-shape of the Variability can be explained by the fact that when few data is used for training the classification will heavily depend on how well those data represents the test set as a whole. Similarly, when the test set is small, the classification will heavily depend on whether if it is difficult or not. In any case, the MoSTiF outperforms HOG and generally have lesser Variability.

Finally the Figure 6 and shows the $F_1$-score 6a and Variability 6b, respectively for the validation set. This time MoSTiF generally outperforms HOG and the U-shape is not visible in the Variability graph since the validation set is kept fixed. Using all learning data gives a validation $F_1$ score of 0.995 and a Variability of $9.116 \cdot 10^{-4}$.




(a) $F_1$-score

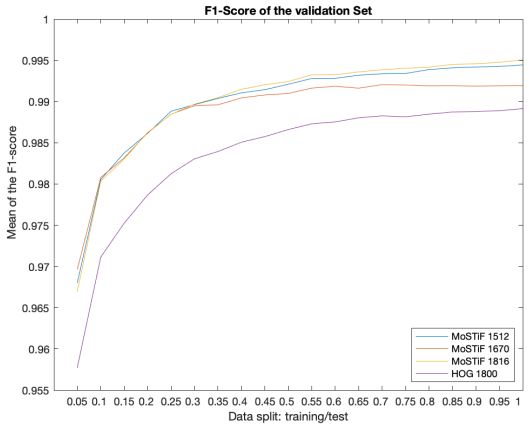



(b) Variability

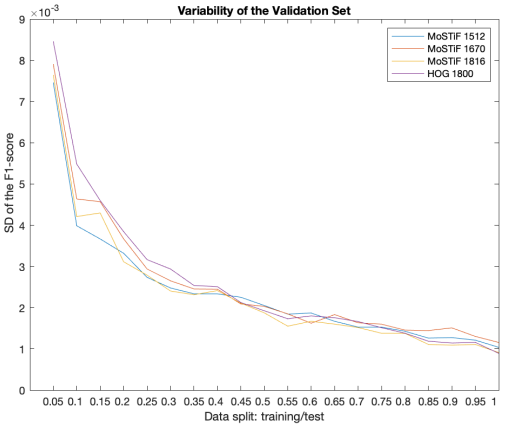
Figure 5: $F_1$-score and Variability (y-axis) for the test set, for 200 random runs by varying the split of data into a training and test set with varying sizes.

## 5.1 PCA Compression

In this section we analyse how to make faster training and classification by training on a fixed number of principal components obtained from the features in the training dataset, instead of using the full length of the training features [HM18]. This lossy compression is achieved by applying PCA on the matrix of features in order to reduce the dimensionality into a smaller number of principal components. The first Principal Component will capture the maximum variance in the features. The second will find variance that is incremental to the first, while still being orthogonal to the first. This process is repeated to find all the principal components. In fact, PCA can only produce as many principal components as there are features in the training dataset. Since each successive principal component captures the variance that is left after its preceding component, the components will be less discriminative and at some point they can be discarded without lowering

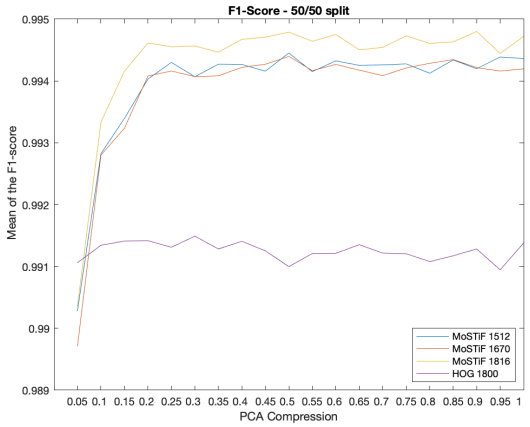



(a) $F_1$-score

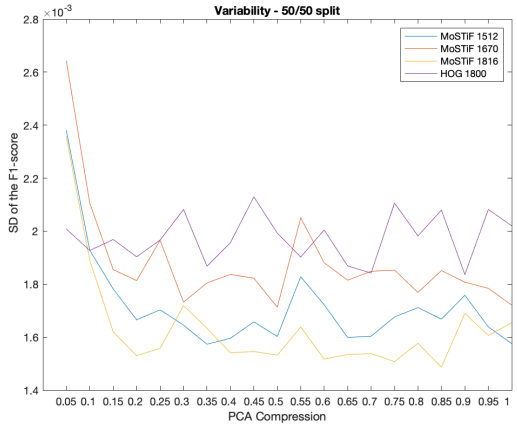



(b) Variability

Figure 6: $F_1$-score and Variability for the validation set, for 200 random runs by varying the split of data into a training and test set with varying sizes.




(a) $F_1$-score (y-axis) for different amounts of compression (x-axis).




(b) variability (y-axis) for different amounts of compression (x-axis).

Figure 7: $F_1$-score and Variability for different amounts of compression for 200 random 50-50 splits of the data.

the $F_1$-score. In fact it might even make it higher since this procedure will denoise the data, by capturing the main signal in the data and hereby omitting the noise.

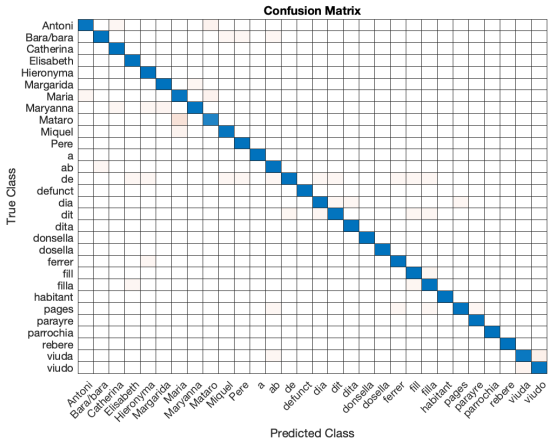
In practice, a number of the first components from the matrix obtained by PCA are kept and the rest are discarded. This matrix is subsequently multiplied to all data (training, validation and test) in order to reduce the dimensionality. This can actually be seen as a neural net applied to the feature vectors, which extracts the most important features in the data, since matrix multiplication is exactly what neurons do. The resulting features are then normalised before being used for training and inference.

Figure 7 shows both the $F_1$-score 7a and the Variability 7b for different amounts of compression for 200 random 50-50 splits of the data, i.e. while keeping the same validation set as before, all the remaining data is used for the learning. Interestingly all MoSTiF features perform slightly better when compressed down to 50% of its original size than using the full length. The MoSTiF 1816 even performs about just as well (99.99%) for only 20% of its original length, i.e. 334 elements long, and the Variability is even becoming lower.

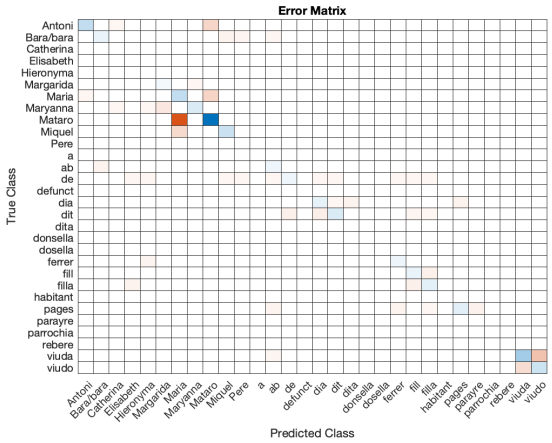
## 6 DISCUSSION

Figure 8 shows the confusion matrix in 8a, which is not so informative when the accuracy is rather high overall. Here the learning set is varied (50% of all available data) while the test set is kept fixed (the remaining 50%). A better view of where the classification do go wrong can be achieved by showing the errors instead, which can be done by computing a new diagonal, taking 1 minus the old diagonal, as shown in 8b.

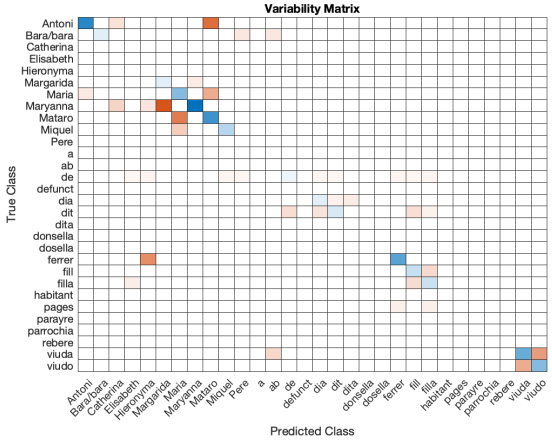
Another way to show where the classification goes wrong is to look at the Variability. This can be done by computing the SD of the $F_1$-score from the confusion matrices for all runs as shown in 8c. Note, that if there are outliers that always are misclassified, then they will




(a) Confusion matrix for HOG with 50% learning data. Mean over 200 runs.

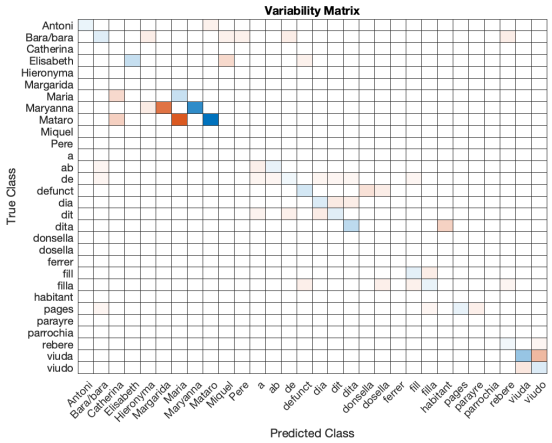



(b) Error matrix for HOG with 50% learning data. Mean over 200 runs.

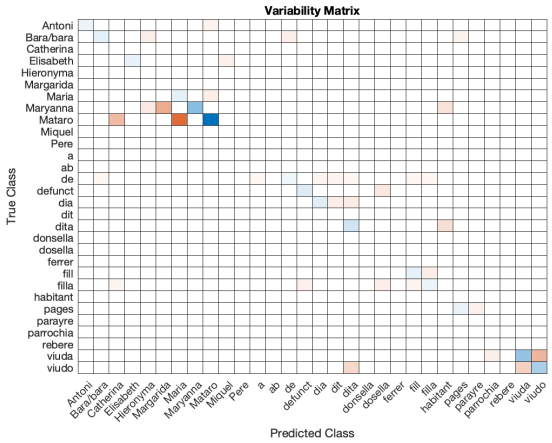



(c) Variability matrix for HOG with 50% learning data. Standard deviation over 200 runs.

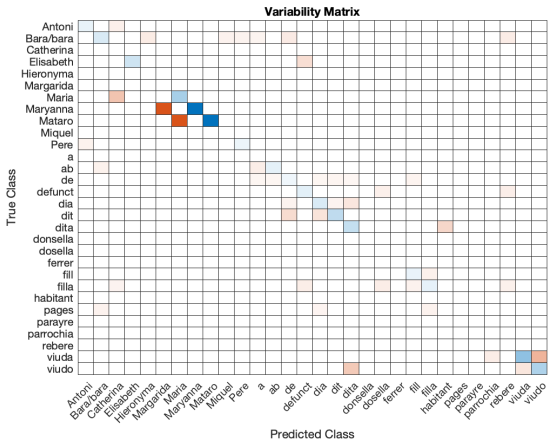
Figure 8: Confusion, Error and Variability matrices. When confusion is low, both the Error and variability matrices tells more about when classifications goes wrong, as the fluctuations become apparent.




(a) Variability matrix for MoSTiF 1512 with 50% learning data. Standard deviation over 200 runs.




(b) Variability matrix for MoSTiF 1670 with 50% learning data. Standard deviation over 200 runs.




(c) Variability matrix for MoSTiF 1816 with 50% learning data. Standard deviation over 200 runs.

Figure 9: Comparison of Variability Matrices for the three different MoSTiF features. Words with high variability are more dependent on the set for learning.

appear in the error matrix but not in the variability matrix, since they do not vary. Hence, the Variability matrix will only show which classes that depends on the actual set for learning, and therefore it will point out which classes that would need more learning data to become more stable.

Figure 9 shows the variability matrices for the three different MoSTiF feature vectors proposed. While comparing the three, one can note that some words are varying regardless of feature vector being used, which indicates that more learning data is necessary for those words. Some vary for only one of the vectors at a time, which indicates that an ensamble of classifiers, using different feature vectors, could be used to more correctly classify those words.

# 7 CONCLUSION

The MoSTiF feature descriptor turns out to be a better choice than HOG for EPSC and word recognition. Of the three different partitionings examined, the MoSTIF 1816 generally gives better $F_1$-score, lower Variability and performs better than the others when being compressed. However, since only one dataset was tested, it is not said that the results generalise to any kind of data. Nevertheless, it works very well for the task of word recognition together with EPSC. Furthermore, by using only about 2 subspaces per class, performing only 6 projections per subspace, and being able to compress the features down to only 20% of its original size, the computational cost for inference is indeed very low.

# 8 ACKNOWLEDGMENTS

This work has been partially supported by the Riksbankens Jubileumsfond (Dnr NHS14-2068:1). The computations were performed on resources provided by SNIC through UPPMAX under project SNIC 2020/15-177. The author wish to thank Raphaela Heil and Ekta Vats for fruitful discussions in the development of the ideas presented. The dataset used is publicly available at `https://andershast.com/datasets/`

# 9 REFERENCES

[ADRS*19] Arrieta A. B., D'iaz-Rodr'iguez N., Ser J. D., Bennetot A., Tabik S., Barbado A., Garc'ia S., Gil-L'opez S., Molina D., Benjamins R., Chatila R., Herrera F.: Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *ArXiv abs/1910.10045* (2019).

[BETVG08] Baya H., Essa A., Tuytelaarsb T., Van Goola L.: Speeded-up robust features (surf). *Computer vision and image understanding 110*, 3 (2008), 346–359.

[BHB16] Barrera T., Hast A., Bengtsson E.: A chronological and mathematical overview of digital circle generation algorithms : Introducing efficient 4- and 8-connected circles. *International Journal of Computer Mathematics 93*, 8 (2016), 1241–1253.

[BJ21] Buchholz T., Jug F.: Fourier image transformer. *CoRR abs/2104.02555* (2021).

[CATvS17] Cohen G., Afshar S., Tapson J., van Schaik A.: EMNIST: an extension of MNIST to handwritten letters. *CoRR abs/1702.05373* (2017).

[CBK*18] Clanuwat T., Bober-Irizar M., Kitamoto A., Lamb A., Yamamoto K., Ha D.: Deep learning for classical japanese literature. *CoRR abs/1812.01718* (2018).

[CHTT96] Carbon M., Hallin M., Tat Tran L.: Kernel density estimation for random fields: the l 1 theory. *Journal of nonparametric Statistics 6*, 2-3 (1996), 157–170.

[CM02] Comaniciu D., Meer P.: Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24*, 5 (May 2002), 603–619.

[CPC19] Carvalho D. V., Pereira E. M., Cardoso J. S.: Machine learning interpretability: A survey on methods and metrics. *Electronics 8*, 8 (Jul 2019), 832.

[CWW15] Chen C., Wang D.-H., Wang H.: Scene character recognition using pcanet. In *Proceedings of the 7th International Conference on Internet Multimedia Computing and Service* (New York, NY, USA, 2015), ICIMCS '15, Association for Computing Machinery.

[DKMJ18] Dutta K., Krishnan P., Mathew M., Jawahar C.: Improving cnn-rnn hybrid networks for handwriting recognition. In *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)* (2018), IEEE, pp. 80–85.

[DT05] Dalal N., Triggs B.: Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on* (2005), vol. 1, IEEE, pp. 886–893.

[FH75] Fukunaga K., Hostetler L.: The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory 21*, 1 (January 1975), 32–40.

[GDDM14] Girshick R., Donahue J., Darrell T., Malik J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2014), pp. 580–587.

[GSC*19] Gunning D., Stefik M., Choi J., Miller T., Stumpf S., Yang G.-Z.: Xai—explainable artificial intelligence. *Science Robotics 4*, 37 (2019).

[HL20] Hast A., Lind M.: Ensembles and cascading of embedded prototype subspace classifiers. *Journal of WSCG 28*, 1/2 (2020), 89–95.

[HLV19] Hast A., Lind M., Vats E.: Embedded prototype subspace classification : A subspace learning framework. In *The 18th International Conference on Computer Analysis of Images and Patterns (CAIP)* (2019), Lecture Notes in Computer Science, pp. 581–592.

[HM18] Hernandez W., Mendez A.: Application of principal component analysis to image compression. In *Statistics*, GÃ¶ksel T., (Ed.). IntechOpen, Rijeka, 2018, ch. 7.

[HSSK18] Hast A., Sablina V. A., Sintorn I.-M., Kylberg G.: A fast fourier based feature descriptor and a cascade nearest neighbour search with an efficient matching pipeline for mosaicing of microscopy images. *Pattern Recognition and Image Analysis 28*, 2 (2018), 261–272.

[HV18] Hast A., Vats E.: Radial line fourier descriptor for historical handwritten text representation. *Journal of WSCG 26*, 1 (2018), 31–40.

[HV21] Hast A., Vats E.: Word recognition using embedded prototype subspace classifiers on a new imbalanced dataset. *Journal of WSCG 29*, 1-2 (2021), 39–47.

[HW79] Hartigan J. A., Wong M. A.: Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics) 28*, 1 (1979), 100–108.

[JJL07] Ji P., Jin L., Li X.: Vision-based vehicle type classification using partial gabor filter bank. In *2007 IEEE International Conference on Automation and Logistics* (2007), pp. 1037–1040.

[KDJ18] Krishnan P., Dutta K., Jawahar C.: Word spotting and recognition using deep embedding. In *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)* (2018), IEEE, pp. 1–6.

[KLR*77] Kohonen T., Lehtiö P., Rovamo J., Hyvärinen J., Bry K., Vainio L.: A principle of neural associative memory. *Neuroscience 2*, 6 (1977), 1065 – 1076.

[KO76] Kohonen T., Oja E.: Fast adaptive formation of orthogonalizing filters and associative memory in recurrent networks of neuron-like elements. *Biological Cybernetics 21*, 2 (Jun 1976), 85–95.

[Koh82] Kohonen T.: Self-organized formation of topologically correct feature maps. *Biological Cybernetics 43*, 1 (Jan. 1982), 59–69.

[Koh95] Kohavi R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2* (San Francisco, CA, USA, 1995), IJCAI'95, Morgan Kaufmann Publishers Inc., pp. 1137–1143.

[Kri19] Krishnan M.: Against interpretability: a critical examination of the interpretability problem in machine learning. *Philosophy & Technology* (2019).

[KRMV76] Kohonen T., Reuhkala E., Mäkisara K., Vainio L.: Associative recall of images. *Biological Cybernetics 22*, 3 (Sep 1976), 159–168.

[KW96] Kohavi R., Wolpert D.: Bias plus variance decomposition for zero-one loss functions. In *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning* (San Francisco, CA, USA, 1996), ICML'96, Morgan Kaufmann Publishers Inc., pp. 275–283.

[Laa07] Laaksonen J.: *Subspace classifiers in recognition of handwritten digits*. G4 monografiaväitöskirja, Helsinki University of Technology, 1997-05-07.

[LCB10] LeCun Y., Cortes C., Burges C.: Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist 2* (2010).

[LCZ*18] Luan S., Chen C., Zhang B., Han J., Liu J.: Gabor convolutional networks. *IEEE Transactions on Image Processing 27*, 9 (2018), 4357–4366.

[Low04] Lowe D. G.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60*, 2 (2004), 91–110.

[MH08] Maaten L. v. d., Hinton G.: Visualizing data using t-sne. *Journal of machine learning research 9*, Nov (2008), 2579–2605.

[MH18] McInnes L., Healy J.: UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv e-prints* (Feb. 2018).

[MHWS17] Matuszewski D. J., Hast A., Wählby C., Sintorn I.-M.: A short feature vector for image matching: The log-polar magnitude feature descriptor. *PLOS ONE 12*, 11 (11 2017), 1–21.

[MNR92] Mehrotra R., Namuduri K., Ranganathan N.: Gabor filter-based edge detection. *Pattern Recognition 25*, 12 (1992), 1479–1494.

[OB21] Opitz J., Burst S.: Macro f1 and macro f1, 2021.

[OK88] Oja E., Kohonen T.: The subspace learning algorithm as a formalism for pattern recognition and neural networks. In *IEEE 1988 International Conference on Neural Networks* (July 1988), vol. 1, pp. 277–284.

[RFS*13] Romero V., Fornés A., Serrano N., Sánchez J. A., Toselli A. H., Frinken V., Vidal E., Lladós J.: The ESPOSALLES database: An ancient marriage license corpus for off-line handwriting recognition. *Pattern Recognition 46*, 6 (2013), 1658–1669.

[SF16] Sudholt S., Fink G. A.: Phocnet: A deep convolutional neural network for word spotting in handwritten documents. In *ICFHR* (2016), IEEE Computer Society, pp. 277–282.

[Sha18] Shapshak P.: Artificial intelligence and brain. *Bioinformation 14*, 1 (2018), 38.

[SSTF*15] Simo-Serra E., Trulls E., Ferraz L., Kokkinos I., Fua P., Moreno-Noguer F.: Discriminative learning of deep convolutional feature point descriptors. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)* (USA, 2015), ICCV '15, IEEE Computer Society, pp. 118–126.

[WLK*67] Watanabe W., Lambert P. F., Kulikowski C. A., Buxto J. L., Walker R.: Evaluation and selection of variables in pattern recognition. In *Computer and Information Sciences* (1967), Tou J., (Ed.), vol. 2, New York: Academic Press, pp. 91–122.

[WP73] Watanabe S., Pakvasa N.: Subspace method in pattern recognition. In *1st Int. J. Conference on Pattern Recognition, Washington DC* (1973), pp. 25–32.

[XRV17] Xiao H., Rasul K., Vollgraf R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR abs/1708.07747* (2017).

[ZK15] Zagoruyko S., Komodakis N.: Learning to compare image patches via convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2015).