Ensembles and Cascading of Embedded Prototype Subspace Classifiers

Anders Hast and Mats Lind
Department of Information Technology
Uppsala University
SE-751 05 Uppsala, Sweden
anders.hast@it.uu.se;
mats.lind@it.uu.se

ABSTRACT

Deep learning approaches suffer from the so called interpretability problem and can therefore be very hard to visualise. Embedded Prototype Subspace Classifiers is one attempt in the field of explainable AI, which is both fast and efficient since it does not require repeated learning epochs and has no hidden layers. In this paper we investigate how ensembles and cascades of ensembles perform on some popular datasets. The focus is on handwritten data such as digits, letters and signs. It is shown how cascading can be efficiently implemented in order to both increase accuracy as well as speed up the classification.

Keywords
Subspaces, Ensembles, Cascading, Embedded Prototypes, Neural Networks, Deep Learning.

1 INTRODUCTION

Recently, deep learning based methods [Sha18], have shown tremendous performance in the area of handwritten text recognition [KDJ18, DKMJ18, SF16]. Nevertheless, there are a few problems and challenges that still need to be addressed. First of all these methods usually require powerful GPU resources in the training process, not at least because of the back propagation in numerous epochs, which makes them very time consuming. Secondly, the learning process of a deep neural network can be regarded as a black-box [CPC19]. Furthermore, the learning mechanism is not easy to comprehend nor to visualise, because of the many hidden layers. This is know as the interpretability problem [Kri19, CPC19] and in order to find a solution the field of explainable AI (XAI)[ADRS*19, GSC*19, CPC19] has emerged.

A new kind of learning process together with a well researched classification method [KLR*77] was recently proposed [HLV19] called *Embedded Prototype Subspace Classification* (EPSC), which go beyond the black-box deep neural network. It is a mathematically well-defined neural net based on subspaces (or manifolds) aimed at classification of handwritten letters, which is both easy to interpret, explain and visualise. This method is an especially interesting alternative when speed is crucial since it is not based on back propagation and therefore does not require an iterative training procedure. Even if it not always beats the state of the art, it comes close enough to be an interesting alternative due to its interpretability and compactness, having just one input layer and one output layer with no hidden layers.

In this paper we present how ensembles of EPSC can be set up in such a way that classification can be done in an efficient manner. Cascading is used to progressively exclude nodes in the resulting neural network that cannot contribute to the correct classification. The idea is to make classification faster without compromising accuracy.

2 BACKGROUND

The first application of subspaces in pattern recognition was proposed by Watanabe et al. [WP73] in 1967, and later further developed [WLK*67], by Kohonen and others [KLR*77, KO76, KRMV76, OK88]. The idea of the later approaches of subspace learning was to choose some group of prototypes for the construction of each subspace. This was done by searching for the k nearest neighbors in feature space, which is a rather time consuming process. In the following subsections a more efficient and accurate procedure is explained.

2.1 Embedded Prototype Subspace Classification

The idea of EPSC [HLV19] was to use t-distributed stochastic neighbour embedding (t-SNE) [MH08] to find representative clusters in 2D image space by applying kernel density estimation (KDE) [CHTT96] and a so called inverse watershed transform (IWS) [RM00, VS91], which is simply the watershed transform on an inverse image. Therefore, *valleys* between mountain tops are found in the image rather than the *drainage divide* that separates adjacent drainage basins, also known as watersheds.

The t-SNE is a machine learning technique that reduces the number of dimensions of high dimensional data to 2 or 3 dimensions. Clusters are formed since it strives to maintain a representation of similarities among features, so that similar features are represented as points closer to each other and dissimilar points further away from each other.

PCA [WEG87] is used both as an intelligent start guess for t-SNE in order to make the process deterministic, but also for generating each subspace [Laa07]. By computing the norm of the projected feature vector to be classified into each subspace, the process can be regarded as a two layer neural network [OK88, Laa07], where the weights are mathematically defined through PCA. Another important advantage is that the learning process can easily be visualised, which makes it easy to understand, interpret and explain compared to most state of the art deep learning approaches.

2.2 Clustering

The main idea behind clustering, or cluster analysis, is to group data that are in some sense similar into separate clusters. Usually some kind of distance function is used to group similar data points together [JMF99]. Clustering is a common technique for exploratory data mining and analysis, and is used in many fields, including machine learning, pattern recognition, image analysis, machine vision and its applications [JMF99].

The selection of an appropriate clustering approach is indispensable for the performance of a machine vision task [XW05]. Clustering can be done in many ways, and a variety clustering algorithms have been proposed in literature, such as k-means [HW79] and DBSCAN [EKSX96]. These algorithms require the user to set the number of clusters to be found, while others, like Mean-Shift [CM02, FH75] finds the number of clusters depending on the size of the Gaussian Kernel chosen.

Just as in [HLV19], it was chosen to perform clustering in image space in order to easily be able to visualise it as the learning process proceeds. Nonetheless, the method used is producing a similar result as Mean-Shift.

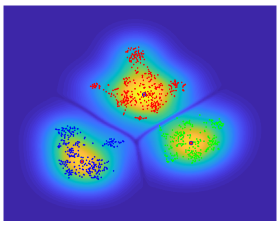
2.3 Kernel Density Estimation

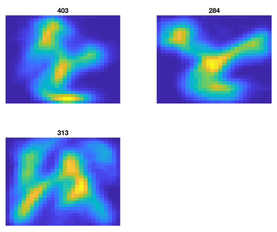
Kernel Density Estimation (KDE) [CHTT96] can be performed by splatting Guassian discs onto an image, for each point in the dataset. The size d of each disc can be set in a similar way as the kernel size for Mean-Shift. However, since KDE and IWS work in image space, the result can easily be visualised. It was chosen in [HLV19] to use the Silverman's rule of thumb as the base level for computing the bandwidth h of the clustering,

$$h = \left(\frac{4\sigma^5}{3n} \right)^{1/5} \quad (1)$$

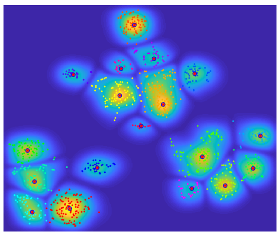
where σ is the standard deviation of n samples.

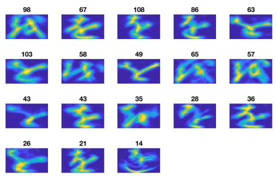
The KDE surface can be used in a visualisation as a visual aid to identify clusters, but is also used as the basis for the IWS. By multiplying h by a scale factor, it is possible to vary the number of clusters.


(a) Clusters, $h = 3.0$.



(b) Heatmaps for the 3 resulting clusters.


(c) Clusters, $h = 1.0$.

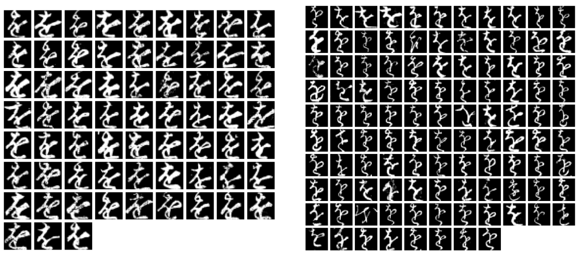


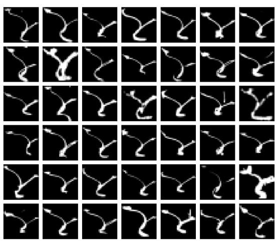
(d) Heatmaps for the 18 resulting clusters.

Figure 1: Visualisation of a subset using one character from the Kuzushiji-MNIST dataset. (a) Visualisation using low resolution. (b) Heatmaps from each cluster. (b) Visualisation using high resolution. (c) Heatmaps from each cluster. The number above the heatmaps tells how many images are superimposed to make the heatmap. Figure best viewed in color.

Figure 1 shows the result of clustering using the character を from the Kuzushiji-MNIST dataset. In Figure 1a) a rather large scale factor $h = 3.0$ is chosen and three main clusters appears representing the three main ways of writing the character を. By decreasing the scale factor to $h = 1.0$ more clusters are found as shown in Figure 1c) and the heat maps in Figure 1d) reveal more variation among the individual characters.

In Figure 2 all characters from four different clusters are are depicted. Each cluster contains rather similar ways


(a) Two different sub-clusters from the upper main cluster.


(b) Another variant of the character from the lower left cluster.

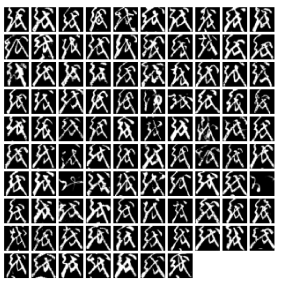

(c) Yet another variant from the lower right cluster.

Figure 2: Visualisation of the actual content of some clusters. (a) Two different clusters belonging to the first heatmap in Fig. 1b, which are represented in Fig. 1d as the second, third and fourteenth heatmap respectively. (b) This cluster belongs to the second heatmap in Fig. 1b, and is the eleventh in Fig. 1d. (c) This cluster belongs to the third heatmap in 1b, and is the first in Fig. 1d.

of writing each character. This is an important feature of the clustering since it makes it possible to create subspaces that correspond to each character variation, i.e. it makes it possible to classify similar looking characters by each such subspace.

2.4 Subspace Definition

We follow the same definition of subspaces as in [HLV19, Laa07, OK88]. 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 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,

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

By projecting the vector $\mathbf{x}$ into each and every subspace $\mathscr{L}_{\mathbf{U}_k}$ classification is performed. The vector $\hat{\mathbf{x}}$ will therefore be a reconstruction of the input vector $\mathbf{x}$, using all vectors in the subspace through

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

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

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

Since all the vectors in $\mathbf{U}$ are normalised, 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 subsequently also have the largest norm $||\hat{\mathbf{x}}||^2$.

3 CASCADING AND ENSEMBLES

Cascading [GB00], first introduce by Viola and Jones for face detection [VJ01] can be regarded as a special case of ensemble learning [OM99, Rok10]. It is based on the concatenation of several classifiers rather than combining the result of multiple classifiers in order to obtain a better result. The idea is that further processing is done only on tentative classes and non-relevant classes are progressively excluded. By implementing cascading for a neural network, the overall workload will not only decrease, but in some sense the network will mimic what humans do. When being presented to a classification problem a person often instantly see the correct answer, i,e. when being sure enough, no further examination is necessary and therefore we sometimes naturally make use of what is known as *early termination*. However, sometimes a person have to take a closer look and choose from two or more tentative answers by scrutinising the images further, e.g. "is the number a sloppily written '5' or a '6'?". Hence, the decision has already been made that it cannot be any other number. Such closer examination helps in making a more accurate classification and therefore humans also deploy the strategy of cascading in such cases.

3.1 Ensembles

Ensembles can be setup in several ways, and here was chosen to use four EPSC networks working on the very same test data images but using four different variants of features that are extracted from the images. Hence, the t-SNE will create a bit different looking clusters depending on what kind of image features the different features extractors are focusing on.

3.2 Features

The features used herein are handcrafted in order to be faster than learned features from CNN's. They were made from different combinations of HoG [DT05] and combinations of some of the most significant elements of the magnitude of the FFT (subsequently referred to as mFFT), i.e. low pass filters, [MHWS17] of different parts of the image and also in combination with down scaled versions of the image itself. These were made ad hoc through experiments in order to find good features yielding both high accuracy used alone, but also used together in an ensemble. In the paper introducing EPSC [HLV19] a similar approach, using a combination of HoG and a down scaled version of the image, achieved an accuracy of 99.2%, i.e. error of 0.8%. As shown in table 1 using mFFT helped lowering the error below that error for all features.

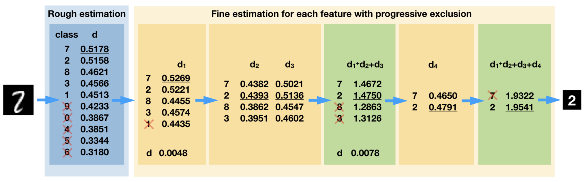
Feature 1 was setup by combining two HoG features with different cell size and five mFFT features from five different partitions of the image. The second Feature is a combination of one HoG and elevn mFFT of eleven different partitions. The third feature is a combination of one HoG, five mFFT and the image itself down scaled to a quarter of its size. The fourth feature is a combination of one HoG and five mFFT.

3.3 Cascading

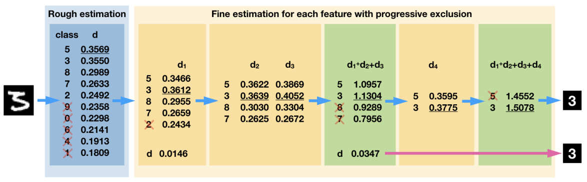
The cascading was setup in six stages as visualised in Figure 3. In the first stage a rough estimation on feature 1 is done by using a rather large scale parameter $h = 2.5$ for clustering, which will produce a quite small neural net that produces good enough results to distinguish with a high probability the (obvious) false positives from the tentative true positives.

In the second stage, one net (still using the same feature) evaluates only the top five candidates in a more accurate but larger net that was produced by using a smaller scale parameter $h = 0.5$ in order to get smaller clusters. Hence, only the parts of the larger net that corresponds to these five candidates need to be evaluated. The lowest ranking candidate is then excluded from further processing. The advantage here is that only one step of t-SNE is used for this particular feature, used in the first two stages, while two different KDE's are produced. The latter is many times faster than the t-SNE so the overhead is small.

In the third stage, the nets corresponding to the four candidates, evaluates the second and third features. These are summed together in the fourth stage, together with the output of the second stage, and a new ranking is done based on the accumulated results from these three finer nets. The two lowest score candidates are then discarded and in the fifth stage the remaining two candidates are evaluated using the fourth feature and their corresponding nets. Finally, a total accumulated



(a) This time a '2' is fed into the network. Both the rough estimation and the first network regards it to be a '7'. However, the others conclude that it must be a '2'.



(b) An image of the number '3' is fed into the network. Note that the rough estimation regards it to be a '5' but the finer estimations concludes it its a '3'. Early termination can be done in this case since the difference between the top two classes is large enough.

Figure 3: Two examples of how the rough classification finds 5 candidates for further investigation in the cascading. Next one net evaluates the candidates using a more accurate but larger net and removes the lowest ranking candidate. Next two nets evaluates the remaining four and a ranking is done based on the accumulated results from the three finer nets. Finally only the top two candidates are evaluated by the fourth net, and a total accumulated result is used to determine the winner. Figure best viewed in color. Note that the top candidate is underlined for each step.

score from all four stages is used to finally determine the class.

The whole procedure is visualised for two examples of non trivial images in Figure 3. Note how the pipeline eventually comes up with the correct answer even if the top candidate is wrong in the beginning of the pipeline.

3.4 Early Termination

If the difference in score between the top candidates at stage 2 or 4 exceeds a certain threshold θ, it can in most cases be safely concluded that the top candidate is already found, since the uncertainty is considered low. If the difference in score on the other hand is low, it means that the nets cannot easily tell which class it is, i.e. the uncertainty is high, and further processing is necessary. Experimentally it was found that a threshold of $\theta = 0.03$ in both stages worked well. However, since stage 4 is the sum of three scores it means that the values compared are higher and the difference generally is higher, which in its turn means that the threshold is in fact set more tight in this stage.

4 METHOD AND DATASETS

This section describes the method used in the experiments and the individual datasets.

Three important questions needed to be addressed. First of all, how much would an ensemble of EPSC improve the classification results? The second is, how much would cascading improve speed and how close to the accuracy of the ensemble could be achieved? Finally, how much would an early termination mechanism affect both speed and accuracy?

First different variants of features where run on MNIST to find good candidates. A combination of HoG and mFFT on the whole image (or partitions of the image) turned out to give better results than just using HoG and a down scaled version of the image. As explained earlier they were made ad hoc in a trial and error process. This means that other features might do better for other types of input data, and we propose this for future research.

Next combinations of four features where run in an ensemble to find what four features actually give the most different classifications in order to diminish the total error in the ensemble.

Finally, the experiments where repeated on the other data sets without changing any parameters or features. This was done in order to determine whether the cascading and ensembles still improves on these datasets.

4.1 Datasets

The proposed EPSC framework has been tested on the images from the following datasets:

- The MNIST database of handwritten digits [LCB10], which is the standard benchmark dataset within the machine learning community. It consists of a training set of 60,000 examples, and a test set of 10,000 examples.
- The E-MNIST (MNIST) dataset has the same example and test set sizes as MNIST [CATvS17], but the conversion process tried to optimise the size of each digit.
- L-EMNIST has the same example and test set sizes as MNIST and contains letters (26 in total).
- B-MNIST has the same example and test set sizes as MNIST and contains both letters and digits (47 different in total).
- The Kuzushiji-MNIST (K-MNIST) dataset has the same example and test set sizes as MNIST [CBK*18] and contains 10 differennt Japanese Hentaigana, where each character may possess rather different forms. The letters present in the dataset are あきすつなはまやれを.
- The Fashion MNIST (F-MNIST) [XRV17] dataset has the same example and test set sizes as MNIST and contains miniature images of clothes and bags.

These were chosen for several reasons. First of all most of them contain handwritten numbers and letters, which is of our main interest, except for the F-MNIST dataset that contains miniature images of clothes and bags. Secondly, they all have similar format, making them easy to evaluate for algorithms that have already been tested on the MNIST dataset, which is very well researched for machine learning. Nevertheless, it is a good starting point for understanding how well classification of handwritten text can perform.

5 RESULTS

Table 1 shows the classification errors on the datasets using different approaches and features. In all cases the four features used on ordinary EPSC (denoted $F1$, $F2$, $F3$ and $F4$) perform a little better than using the feature for EPSC reported in [HLV19].

More importantly, cascading on all features (C-ESCP) as well as a full ensemble without cascading (E-EPSC) perform much better than any of the single networks using any of the features. One can also note that cascading performs almost as well as the full ensemble, but being much faster. Furthermore, the cascading on just one feature (C-F1), which corresponds to the first stage in Figure 3 also works very well. It is, not surprisingly, the fastest of all methods as can be seen in Table 3, but is of course not the most accurate since it does not make use of all the features and the full cascading pipeline.

The result of using C-EPSC compared to what was presented in different papers introducing each of the datasets are shown in Table 2. Certainly, better accuracy have been achieved by much more complicated deep learning architectures since the datasets where introduced. However, the point here is not to beat the state of the art by using the C-EPSC, but rather to show that it is possible to come very close to a much lesser cost, both for learning but also for the classification.

The cascading with early termination, denoted C_{et}, comes close to what the cascading does when it comes to accuracy, but is quite a lot faster. In fact, it comes close to what cascading using only one feature (C-F1) does. The reason is that the whole cascading is seldom necessary for an accurate classification. Table 4 shows the percentage of classifications that goes on beyond stage 2 and 4. Hence, the lower percentage, the more cases can be terminated early.

It can be seen in Table 4 that the percentage vary quite a lot among the datasets and one reason is probably the fact that the same threshold of $\theta = 0.03$ for both stages were used for all datasets. This implies that this is one parameter that could be learned for an implementation of word classification.

Table 1: Prediction error (%) using different features, cascading (C), with early termination (C_{et}) and the full ensemble (E)

Dataset	EPSC	E-EPSC	C-EPSC	C_{et}-EPSC	C-F1	F1	F2	F3	F4
MNIST	0.80	0.56	0.56	0.56	0.69	0.69	0.75	0.70	0.74
E-MNIST	0.78	0.58	0.59	0.60	0.66	0.65	0.63	0.66	0.61
L-MNIST	7.60	6.38	6.43	6.43	6.90	6.89	6.97	6.84	6.98
B-MNIST	14.13	12.62	12.65	12.66	13.35	13.32	13.28	13.52	13.66
K-MNIST	3.08	2.17	2.20	2.19	2.52	2.51	2.15	2.35	2.58
F-MNIST	11.88	8.73	8.73	8.73	10.11	10.11	10.05	9.82	10.30

Table 2: Classification accuracy (%) for some popular datasets and comparison with other learning methods. Note that we have taken the results from each paper proposing these methods and therefore the table is not covering the use of all methods on all datasets.

Learning method	MNIST	E-MNIST	L-MNIST	B-MNIST	K-MNIST	F-MNIST
# classes	10	10	26	47	10	10
C-EPSC	99.44	**99.41**	**93.57**	**87.35**	97.80	**91.23**
EPSC [HLV19]	99.20	99.22	92.40	85.87	96.92	88.12
OPIUM [CATvS17]	-	96.22	85.15	78.02	-	-
MLP [XRV17]	97.20	-	-	-	-	87.1
Keras CNN[CBK*18]	99.06	-	-	-	95.12	-
PreActResNet-18 [CBK*18]	99.56	-	-	-	97.82	-
PreActResNet-18 + Manifold Mixup [CBK*18]	**99.54**	-	-	-	**98.83**	-

Table 3: Wall clock time of classification in seconds of using different features, cascading (C), with early termination (C_{et}) and the full ensemble (E).

Dataset	E-EPSC	C-EPSC	C_{et}-EPSC	C-F1	F1	F2	F3	F4
MNIST	825	381	192	187	250	247	178	157
E-MNIST	797	379	193	194	248	228	177	153
L-MNIST	3667	834	592	509	1088	1050	736	696
B-MNIST	4296	691	565	478	1321	1198	888	786
K-MNIST	614	306	172	159	186	183	131	118
F-MNIST	684	326	257	164	202	202	147	134

Table 4: Percentage of how many are not terminated early in step 2 and 4 respectively, but passes to the next step in the cascading with respect to the total amount of images to be classified.

Dataset	ET-2	ET-4
MNIST	6.23%	1.04%
K-MNIST	11.52%	2.75%
F-MNIST	63.20%	33.73%
E-MNIST	4.01%	0.77%
L-MNIST	45.30%	21.22%
B-MNIST	30.20%	12.15%

6 CONCLUSION

Deep learning architectures and the learning and classification process are not easy to visualise, nor to explain. The EPSC on the other hand is based on clustering of embedded features. Both the clusters and corresponding images as well as the resulting neural nets can easily be visualised and comprehended. Therefore, EPSC is an interesting alternative in the domain of XAI. It was experimentally shown that both ensembles and cascading can improve the speed and accuracy. Furthermore, early termination bring down the cost of classification quite a lot, without compromising the accuracy.

For future work we propose to use EPSC for word images rather on individual letters. Moreover, it is necessary to find some plausible explanation why HoG and features based on the magnitude of FFT works so well together.

7 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).

[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.

[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 *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* (June 2005), vol. 1, pp. 886–893 vol. 1.

[EKSX96] Ester M., Kriegel H.-P., Sander J., Xu X.: A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (1996), KDD96, AAAI Press, pp. 226–231.

[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.

[GB00] Gama J., Brazdil P.: Cascade generalization. *Machine Learning 41*, 3 (Dec 2000), 315–343.

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

[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.

[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.

[JMF99] Jain A. K., Murty M. N., Flynn P. J.: Data clustering: a review. *ACM computing surveys (CSUR) 31*, 3 (1999), 264–323.

[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.

[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.

[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).

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

[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.

[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.

[OM99] Opitz D., Maclin R.: Popular ensemble methods: An empirical study. *J. Artif. Int. Res. 11*, 1 (July 1999), 169–198.

[RM00] Roerdink J. B., Meijster A.: The watershed transform: Definitions, algorithms and parallelization strategies. *Fundam. Inf. 41*, 1,2 (Apr. 2000), 187–228.

[Rok10] Rokach L.: Ensemble-based classifiers. *Artificial Intelligence Review 33*, 1 (2010), 1–39.

[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.

[VJ01] Viola P., Jones M.: Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001* (Dec 2001), vol. 1, pp. I–I.

[VS91] Vincent L., Soille P.: Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence 13*, 6 (June 1991), 583–598.

[WEG87] Wold S., Esbensen K., Geladi P.: Principal component analysis. *Chemometrics and intelligent laboratory systems 2*, 1-3 (1987), 37–52.

[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).

[XW05] Xu R., Wunsch D.: Survey of clustering algorithms. *IEEE Transactions on neural networks 16*, 3 (2005), 645–678.