Exhaustive AI Analysis on Handwriting Recognition

Phani Ram Sayapaneni
Department of Computer Science
University at Buffalo
Buffalo, NY 14226
phaniram@buffalo.edu

Abstract

The problem of handwriting recognition could be analyzed through 3 mutually exclusive and exhaustive methods, 1.) Knowledge based approach, 2.) Simple machine learning approach 3.) Deep learning approach. Our interest is to find best solution in each of these approach and compare with the results across each approach.

1 Introduction

The problem of handwriting recognition solves many real world problems such as crime detection(forensics), fraud detection in banking etc. The problem statement breaks down to a binary question, "whether 2 samples of handwriting came from same person or not?". Due to improvements(efficient implementations) in open source AI libraries, the above mentioned machine learning approaches have been tried and tested in short period of time.

2 Knowledge based approach based on probabilistic graphical models

2.1 Exploratory data analysis

The dataset consists of handwritten samples of the word "AND". The characteristics consists of 9 features entered by a human (document examiner). Theses 9 features represent characteristics of each letter such as initial stroke for letter 'a', staff for letter 'a' and 'd', number of arches foe letter 'n' etc. Total number of samples 3524.

Pearson's $\chi 2$ test has been applied to determine the correlation between multiple features in the data.
Pearson's $\chi 2$ Co-efficient between 2 features X, Y can be computed as: cov(X,Y)/(sigmaX, sigmaY). Where is cov(X, Y) gives the covariance between X, Y and sigmaX is standard deviation of X, similarly sigmaY is standard deviation of Y.

	f1	f2	f3	f4	f5	f6	f7	f8	f9
f1	1.000000	0.114705	0.050296	0.054837	0.090000	0.076701	0.018310	0.113312	0.236818
f2	0.114705	1.000000	0.008641	0.044863	0.015562	0.138967	0.026788	0.055768	0.096609
f3	0.050296	0.008641	1.000000	0.064748	0.093494	0.053149	0.067699	-0.004516	-0.006052
f4	0.054837	0.044863	0.064748	1.000000	0.050800	0.031756	0.000697	0.064316	0.067617
f5	0.090000	0.015562	0.093494	0.050800	1.000000	0.030996	-0.014647	0.065161	0.321086
f6	0.076701	0.138967	0.053149	0.031756	0.030996	1.000000	0.024580	0.115629	0.154918
f7	0.018310	0.026788	0.067699	0.000697	-0.014647	0.024580	1.000000	-0.010505	-0.138434
f8	0.113312	0.055768	-0.004516	0.064316	0.065161	0.115629	-0.010505	1.000000	0.226027
f9	0.236818	0.096609	-0.006052	0.067617	0.321086	0.154918	-0.138434	0.226027	1.000000

Fig. 1

2.2 Probabilistic graphical model construction

Before constructing the model it is important to observe the correlation values and interpret relationships amongst them.

Interpreting correlation values:

The correlation value: Interpretation

–0.70 :A strong downhill (negative) linear relationship

–0.30: A weak downhill (negative) linear relationship

0: No linear relationship

+0.30: A weak uphill (positive) linear relationship

+0.70: A strong uphill (positive) linear relationship

+1: A perfect uphill (positive) linear relationship

Based on the correlation values obtained in Fig1, there has been not a single instance with moderate or strong correlation amongst any features Xi, Xj where X is the feature i,j belongs to [1,9] .

It is simply illogical to construct a PGM just based on Pearson's co-efficients, since such a model would not have any directed edges between any feature Xi, Xj where X is the feature and i,j belongs to [1,9] . Hence domain knowledge, general logical have been applied to build the PGM.

Let Xi, Yi be the features of 2 images X, Y and C be the class the features belong to (C= 0 when the images come from same writer and C= 1 otherwise).

The following hypothesis have been developed.

1. The combination of variables essentially depend upon on the class and not the class depending on features.
2. The features within a single image is fairly independent, i.e, Xi _|_ Xj from Fig1.

Hence the model below shown is constructed.

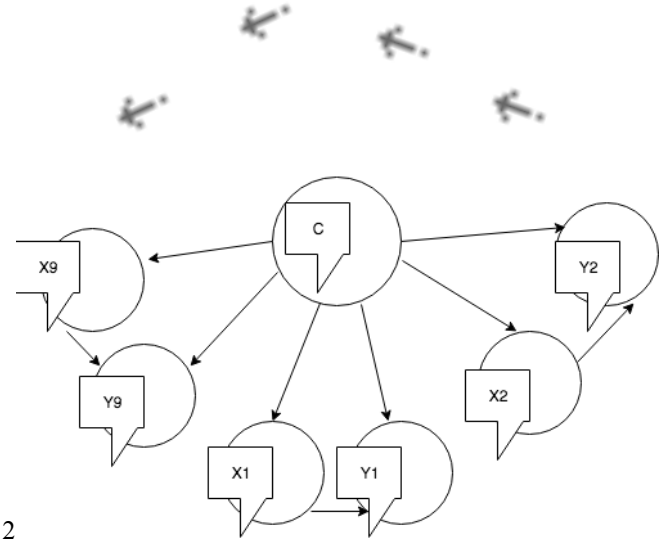

Fig. 2

Important elements about the graph.

1. The graph represents the independencies (poor correlation) of features from the data in Fig.1
2. Given the class, (Xi, Yi) becomes very strongly correlated(positive if same class, negative if different class). This comes from the assumptions, a person tends to write in similar fashion (i.e, similar staff, similar strokes etc.) at multiple instances and also these features truly represent the actual similarities or dissimilarities in writing style.

2.3 Implementation and Results

The original data has unequal distribution across both the classes, hence the data has been sampled such that all the training and testing data has equal distribution across the 2 classes. This data has been randomly sampled and distributed into training set, testing set. These 2 sets have been cross trained and cross validated(50:50).

The model has been implemented using PGMPY library. A Bayesian network has been constructed as shown in Fig 2. The model has been trained on the network and tabular CPDs(conditional probability distribution) have been computed. These CPDs capture the probabilities of children given it's parent.

Once the model has been trained, local independencies have been tested. For example: Given the class c, Xi is independent of all Xj, Yj where j not equal to i. Once these independencies have been verified, the model has been tested.

The testing is done by applying inference over the model. The inference applied here is exact inference, specifically Variable Elimination technique. Any query requires, calculation of joint probability, thereby marginalizing required variables across all other variables. This is a sum of products. By elimination out suitable variables first and re-arranging computation to product sum, efficiency in computation could be achieved. Since the feature size is small, when computed naively, it would take 18 computations on numerator and 37 computations on denominator, a total of 55 computations for each query. When variable elimination is applied this computation gets reduced much further depending upon on the range of variable's values for the variable eliminated.

Every time it computes P(c| Evidence = X, Y)

The overall testing accuracy has been 74.26 %. When this accuracy is broken down across individual classes, the prediction for a different class is 86.82% accurate and prediction for

same class is only 61.7% accurate.

2.3 Conclusion

This model has been pretty accurate, however it is more accurate for the class c=1 (86.8% when samples come from different writers) than the class c=0. This could due to the fact that the independencies of the variables captured in the model were very satisfactory enough to identify class c=1. However the similarities in features when class c=0, have either not been captured by the model(model too simple), or the variable values/features chosen are not true representation of the similarity.

3 Simple Machine Learning Approach

The main idea of this approach is to apply well established algorithms to extract features from the images and use those features to classify which class it belongs to, hence named as simple machine learning approach.

3.1 Exploratory Data Analysis

While the dataset consists of handwritten “AND” images, our task is to train a model which classifies whether 2 images have been written by same person or not. Some images were identified as outliers/error prone data sets. Out of 15518 images, 142 images turned out to be outliers, they had different words altogether such as “Jin”, “Passed”, “Months”, “Good”, “attend”, “Grant”. Training on such data would be a complete mistake, all those images have been eliminated from training, testing dataset. The identification of these outliers has been very simple. The average size of all images has been calculated. It is estimated that 99.1% of all images fall within size of 256x256, only 0.9% of images fall beyond this threshold.

3.2 Preprocessing and Sampling

There are total of 15518 images written by only 1568 writers, so on average there are 10 images per writer. The datasets have been sampled such that both classes have equal share within the training and testing dataset combined. After labelling is done, the images have been padded with 1s across the rows and columns to form a square of size 256x256. These images were then down sampled to provide smaller images of size 64x64.

3.3 Feature Extraction

Several algorithms have been explored for feature extraction. In the past, General Motors autonomous self driving teams have shown great interest in SIFT(Scale Invariant Feature Transform) features for their autonomous self driving applications. This prompted the interest to apply a very general feature extraction algorithm such as SIFT to extract features out of handwritten AND images. Another big reason to choose SIFT is that all the captured images are of various dimensions, bringing all of them to constant size, either requires changing in the aspect ratio of images or changing the scale of those images. Also the very process of capturing these images adds a lot of variance in the scale of these images. In that case we need an algorithm which is not dependent on the scale of the features/images. Thus SIFT fits exactly into this requirement.

SIFT Algorithm:

The main aspects of SIFT algorithm is that it identifies key feature points of the image. These key points are identified by finding intensity changes, using difference of gaussians(2 different standard deviation) at nearby scales(sigma, 2*sigma). Since these are difference of gaussians(DoGs), these points would always identified even if the scale changes(as long as we try enough DoGs). Now that key points have been identified, it becomes our interest to describe each of these key points. The Description of these key points from it’s neighbouring pixel values. A 16x16 neighborhood is taken and divided in 16 small squares. At each square 8bit long direction/orientation(gradient) histogram is identified. 16x8, total 128 bit long feature vector is obtained for each key point.

3.3 SIFT Implementation

First, SIFT key points have been identified, then the descriptor values for the images have been computed. The below figure shows the SIFT key points. These keys points have been computed such that even if the size of the image changes, the key points remain same.

Fig. 3

3.4 Machine Learning Implementation

Now a simple neural network or a logistic regression had to be applied to classify if features from 2 sets of images came from same person or not. To implement such a model, the input size of the training data needs to be constant, however the SIFT algorithm generated different number of key points for each image. This creates a problem to train and classify the images.

3.5 Bag of Visual Words for Image Classification

This is a very interesting concept similar to Bag of Words from natural language processing. The idea is to create a vocabulary that can best describe the image in terms of features. Thus even if there are different number of key points, we bag them into distinct bags and describe image still effectively.

Implementation:

It has been estimated that there are an average of 60 SIFT key points across all the images. For every image, based on the SIFT descriptor values, the SIFT key points have been clustered into 1.) 20 clusters, 2.)12 clusters separately using KMeans Clustering algorithm.

The training, testing dataset now comprises of closest key points and their descriptor values to the centroid of these clusters. As shown below. It is interesting to realize how few of the 9 features from first part overlap with the SIFT features, especially for letters ‘N’, ‘D’.

Fig. 4

These points represent the major clusters from the key points of this data.

3.6 Results

Similarly such major key points(cluster representatives) have been passed into the neural network and trained. The resultant model performed poorly on the test data, generating 50% accuracy(both the training, testing data combined have equal number of data points for each class). The model has been tested for various sizes of the cluster, but still the results were poor. Even centroids were computed and trained on the neural network, which did not improve the results either.

3.6 Implementation using Cedar-Fox

Cedar-Fox is a patent protected software developed by CEDAR(center of excellence for document analysis and recognition), University at Buffalo. This tool provides a good reference to compare results on handwriting recognition.

Given the image of AND dataset, the software tool compares the two images at document level, word level, character level. The document level features are termed under macro features and the character level features are termed under the micro features.

Micro features for characters and words are nothing but GSC features. GSC stands for gradient, structural and concavity. Per character it consists of 192 bits of gradient information about the image, 192 bits of structural information about the image and 128 bits of concavity information about the character. For example for the 192 bits if gradient information, the image is broken into a 4x4 matrix an in each matrix, histogram of gradient directions are computed (12 directions), thus 4x4x12= 192 make up the 128 bit information about the gradient.

The Log Likelihood Ratio across each character is computed as

LLR(micro) =
ln(Product(ProbabilitySameWriter(sij))/Product(ProbabilityDifferentWriter(sij)))

where i = 1,2,..N (number of characters)
sij is the similarity index between the character pairs (i,j)
ProbabilitySameWriter(x) is the probability distribution density on variable x(similarity), for data points coming from pairs of characters written by same writer. All those pairs follow a Univariate Gaussian density with a definite mean u, and variance v.ss
ProbabilityDifferentWriter(x) is the probability distribution density on variable x(similarity), for data points coming from pairs of characters written by same writer.
About 70 pairs of images have been manually analyzed using Cedar-Fox tool. Based on the values generated by tool, if the total LLR (log likelihood ratio) >0, the image pair is

classified as “Same Writer” else if the total LLR (log likelihood ratio) <0, the image pair is classified as “Different Writer”. Although the magnitude of LLR represents, confidence of the classification, we have restricted the classification just based on the direction (positive or negative) since the number of classes we want to classify is 2. Based on the result generated by the tool, the classification worked pretty well and the accuracy of prediction is computed as 81.4285%

	A	B	C	D	E	F
1	Image left	Image Right	Macro LLR	Micro LLR	Total LLR	Correct
2	0001a_num1	0001a_num3	5.13	0	5.13	1
3	0001c_num4.png	0001b_num2.png	7.56	0	7.56	1
4	0001c_num3.png	0001c_num5.png	-3.75	0	-3.75	0
5	0001c_num5.png	0001c_num2.png	8.31	0	8.31	1
6	0001c_num1.png	0001c_num1.png	9.03	0	9.03	1
7	0002b_1.png	0002b_2.png	-0.71	0	-0.71	0
8	0002c_1.png	0002c_2.png	6.27	0	6.27	1
9	0003a_1.png	0003a_3.png	1.6	0	1.6	1
10	0003a_2.png	0003a_2.png	2.07	0	2.07	1
11	0003b_3.png	0003b_1.png	-27.73	0	-27.73	1
12	0004a_num1.png	0004a_num1.png	3.11	3.53	6.64	1
13	0004a_num3.png	0004a_num5.png	1.23	0	1.23	1
14	0004aa_num1.png	0004b_num1.png	0.99	0	0.99	1
15	0004bb_num2.png	0004bb_num1.png	-22.72	0	-22.72	0
16	0004c_num2.png	0004c_num4.png	-8.16	0	-8.16	0
17	0001c_num5.png	0004a_num7.png	-14.06	0	-14.06	1
18	0007a_num2.png	0007c_num4.png	3.62	0	3.62	1
19	0008a_num1.png	0007a_num3.png	-25.26	0	-25.26	1
20	0008c_num1.png	0008c_num2.png	1.16	0	1.16	1
21	0012a_num4.png	0011c_num1.png	-9.19	0	-9.19	1
22	0013aa_num2.png	0013b_num1.png	6.72	0	6.72	1
23	0020a_num2.png	0020b_num1.png	1.36	0.69	2.05	1
24	0028b_num1.png	0028a_num4.png	0.64	0	0.64	1
25	0030c_num3.png	0030c_num2.png	-7.68	0	-7.68	0
26	0034a_num3.png	0034b_num4.png	-0.73	0	-0.73	0
27	0046a_num2.png	0047c_num2.png	-14.13	0	-14.13	1
28	0072b_num2.png	0056c_num4.png	-33.88	0	-33.88	1
29	0070b_num3.png	0070b_num4.png	-11.78	0	-11.78	0
30	0078b_num3.png	0078b_num2.png	-31.16	0	-31.16	1

Fig. 3

3.7 Conclusion

Regarding the SIFT features, the inconsistencies in clustering the key points might have caused the problem, that is at time ti, Image I, feature fi could be clustered in clustered in Ci, however at a different instant it could get clustered in a different cluster Cj(i not equal j). This results in more randomness in the input vector. In other words we might be comparing oranges with apples, even though we had apples in our bag.

The Cedar-Fox software worked pretty well, since it is exclusively designed for tasks involving document analysis, the tool was very good at classifying if the images came from same writer unlike the aforementioned Bayesian network. However this tool might severely fail when applied at different classification tasks such as object recognition. The tool is not applicable for any general image classification task. However the research efforts to come up with such an effective tool for handwritten data analysis should be appreciated.

4 The Deep Learning Solution

.In the recent times with the advancements in the hardware technology (CPU, GPU), deep learning has really caught up the attention across various academic researchers and corporate industries. In our project, the entire solution could be broken down into 2 parts: Feature Learning and Deep Neural Network (classification)

4.1 Feature Learning

Feature learning algorithms find the common patterns that are important for classification amongst the classes. As we have seen in Simple ML Approach, SIFT, GSC algorithms are tailored exclusively to certain datasets such as scale variant images, handwriting recognition tasks. However they could not be applied to any datasets, deep learning solves this problem through convolutional neural networks.

Convolutional Neural Networks are shift invariant and space invariant neural networks, inspired from biological mechanisms within mammalian visual cortex. Mammalian visual cortex has evolved to extract visual features to protect themselves from predators and to improve survival chances while hunting for food. As the name convolution suggests, a kernel (small matrix) is made to convolve across the pixel locations, all over the image. This convolution operation captures the neighborhood information of the pixel values. When different kernels are used and multiple convolutions are performed we develop a 3 dimensional cuboid. The depth of this cuboid corresponds to the number of kernels used. While moving the kernel across the image, we can strategically move the kernel across the image, this is called as striding. It is the number of pixels, we slide our kernel over the image. More is the stride size, smaller is the size of our resultant feature maps. By convolution, it is meant that the neural network learns the parameters which result into a convolution operation.

Convolution network architecture:

$Cij = Xij*Kd$

Where Ci is the convolution output at location (i,j) and K is the kernel at depth d.

Parameter details:

Size of the image: 64x64x3 (downsized to improve computation speed)

The initial convolution depth has been: 32 filters

Kernel size: 3x3

The convolution is applied twice and a max pooling layer is applied. This down samples the feature map size, reduces the precision of location of features but still carries sufficient neighborhood information.

This CNN operation is carried out for both the images where the CNN layers share the weights across the parameters and the resultant tensors are flattened and concatenated to provide an input tensor to the deep neural network.

4.2 Deep Neural Network (classification)

The nonlinear features from the CNNs need to be learned to classify the images across 2 images. Since we are trying to learn complex, non-linear features, it is necessary to use a deep neural network for the classification, instead of a simple one.

Once the convolutional neural network ends, the resultant tensor is used as input and trained on a deep neural network for classification across the 2 classes.

Parameter details:

Hidden Layers: 4

Units in hidden layer: 512

Activation function: Relu (Efficient and good enough to capture the non linearities)

Use of “relu” across the CNN and Deep NN: Every layer is now nonlinear combinations of weighted variables. And the computation is also much efficient, since no exponentials are

involved here, yet we still attain non-linearisty.

Output classes: 2(binary class vector)

Loss function: Categorical Cross Entropy

Categorical cross entropy has been chosen, instead of a binary cross entropy. This is to make the implementation (code) open/flexible to add few more classes in future, such as “Not sure” could also be a class, to improve the application.

Optimizer: Stochastic gradient descent (Incremental and minimizes the loss in neat steps)

4.3 Results and Conclusion

Implementation of the deep learning part has been the most difficult part. Due to limited hardware resources (mackbook air), the time to obtain results and refine/tune the parameters has been very huge. Initially the size of the image was 256x256 and the entire hardware would quickly run out of memory. The images have been resized to 64x64, unfortunately the deep neural network wouldn’t learn anything. The training accuracy and testing accuracy was at constant 50% line. Essentially the neural network wasn’t learning anything. Many changes have been made to fix this, such as: 1.) Modify the kernel size, 2.) Modify the batch size, 3.) Modify the optimizer, 4.) Modify the architecture, where instead of having 2 separate CNN channels for 2 images, only 1 CNN channel has been implemented and weights have been shared while training across both the images. 5.)Removed the dropout layers, since there was no clear problem of overfitting, the dropout layers have been eliminated.

These steps improved the model and performance. The model was tested on a new dataset and the testing accuracy was found to be 73.85%, the error was 0.535, due to limited hardware resources, the number of epochs have been restricted to only 20 iterations.

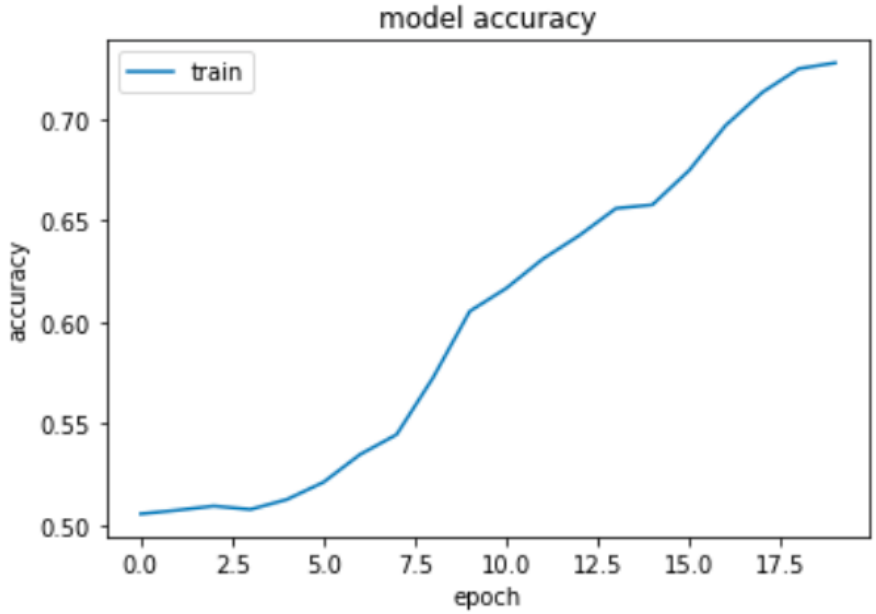

Fig.5

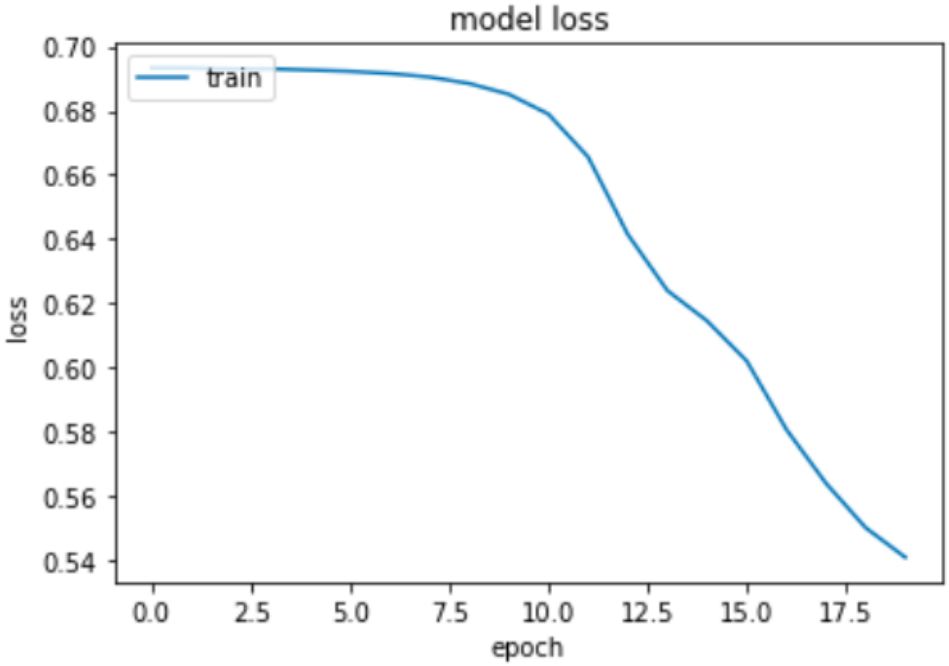

Fig.6 Model Loss

5 Final Analysis

After comparing the three exhaustive approaches, it is very evident that Deep learning approach has been the most innovative and most generalizable solution, it didn't require any domain knowledge or feature extraction techniques. The accuracy of deep learning has been robust when cross validated. The deep learning kernel convolution has been very smart to learn the complex nonlinear features. The activation function added the required non linear complexities across different layers. This level of complexity could not captured across Bayesian networks or Simple machine learning models, where feature extraction has to be determined by a expert. Clearly deep learning solution is the winner here.

Acknowledgments

Would like to thank our Prof. Sargur Srihari and his teaching assistants Mihir Chauhan, Mohammad Abuzar Shaikh, for designing and suggesting good ideas while struck at the roadblocks in the project.

References

[1] Huiyu Zhoua, YuanYuanb, Chunmei Shic. (2008) Object tracking using SIFT features and mean shift, *Computer Vision and Image Understanding Volume 113, Issue 3, March 2009*.

[2] Cong Geng, Xudong Jiang, (2009) Face recognition using sift features (ICIP), 2009 16th IEEE International Conference on Image Processing.

[3 Siyuan Chen, S. Srihari, (2005) Use of exterior contours and shape features in off-line signature verification. Eighth International Conference on Document Analysis and Recognition (ICDAR'05).

www.keras.io

http://www.robots.ox.ac.uk/~az/icvss08_az_bow.pdf, Bag of visual words.