Evolutionary Intelligence
https://doi.org/10.1007/s12065-020-00427-3

RESEARCH PAPER

Investigating the use of an ensemble of evolutionary algorithms for letter identification in tremulous medieval handwriting

Ronnypetson Souza da Silva[1] · **Márjory Da Costa-Abreu**[2] · **Stephen Smith**[3]

Received: 16 December 2019 / Revised: 16 March 2020 / Accepted: 1 May 2020

Abstract
Ensemble classifiers are known for performing good generalization from simpler and less accurate classifiers. Ensembles have the ability to use the variety in classification patterns of the smaller classifiers in order to make better predictions. However, to create an ensemble it is necessary to determine how the component classifiers should be combined to generate the final predictions. One way to do this is to search different combinations of classifiers with evolutionary algorithms, which are largely employed when the objective is to find a structure that serves for some purpose. In this work, an investigation is carried about the use of ensembles obtained via evolutionary algorithm for identifying individual letters in tremulous medieval writing and to differentiate between scribes. The aim of this research is to use this process as the first step towards classifying the tremor type with more accuracy. The ensembles are obtained through evolutionary search of trees that aggregate the output of base classifiers, which are neural networks trained prior to the ensemble search. The misclassification patterns of the base classifiers are analysed in order to determine how much better an ensemble of those classifiers can be than its components. The best ensembles have their misclassification patterns compared to those of their component classifiers. The results obtained suggest interesting methods for letter (up to 96% accuracy) and user classification (up to 88% accuracy) in an offline scenario.

Keywords Tremor · Classification · Ensemble · Evolutionary · Genetic algorithm

1 Introduction

Handwriting can give us valuable information about the clinical condition of a person, as handwriting involves the workings of the brain, the eyes, and the hands. Besides that, in the context of retrospective diagnosis, documents left by people who passed away are a very important source of information for diagnosis, and sometimes it is the only registry available. Paleographers who study the ageing of old scribes, for instance, need to analyse the features from old documents in order to identify illnesses like the common condition known as "essential tremor".

The aim of this work is to develop and analyse a framework of letter identification in a context of limited data from medieval scripts. Such framework makes use of different techniques like ensemble classifiers, evolutionary search, and convolutional neural networks, besides some preprocessing and data obtention techniques like image flood-filling and image binarization.

In the present setting of medieval handwriting letter classification and limited data, it is verified how the ensemble classifiers obtained with evolutionary search take advantage of the base classifiers accuracy. It is proposed that an ideal ensemble, one which performs at least as good as its best component in every class, can be found with the combination of techniques proposed.

Classifiers created from different techniques and trained in different configurations can also specialise in correctly classifying instances from different subsets of a given data set. This variety of prediction in a set of classifiers can be used to take advantage of the classifiers' specialties and therefore it can also be used to obtain a better classifier than the previous ones.

In the context of evolutionary search, trees representing the combination of operations over the outputs of the base

✉ Márjory Da Costa-Abreu
m.da-costa-abreu@shu.ac.uk

[1] UNICAMP, Sao Paulo, Brazil

[2] Sheffield Hallam University, Sheffield, UK

[3] University of York, York, UK

Published online: 15 June 2020

classifiers can be looked for and serve as the structure of an ensemble classifier. The resulting ensemble can then be evaluated so one can know wheter any improvement happened. The measure of improvement can be obtained from comparing the accuracy of the component classifiers with the accuracy of the ensemble. The comparison can be made over the overall accuracy of the ensemble and its components or it can be made considering not just overall accuracy, but also the accuracy in each class.

Understanding classifiers' capabilities and limits in relation to some given data can guide us in the process of either continually looking for better ensemble classifiers or ending the search because no further functionality can be obtained with the base classifiers and data at hand. In this essay, the predictions of classifiers trained on a limited data set are combined by ensemble structures in the search for better classification accuracy. The resulting ensembles are then compared with its components in every one of the classes on the given tasks of vowel identification and writer identification. It is expected that a good ensemble performs at least as well as the best individual classifier in a given class.

This type of search engine is ideal for the specific problem of how to perform user identification in medieval manuscripts which are many centuries old and can have a wide range of different problems such as noise, ink degradation, and/or ink failures in writing. In the present case, the interest is in finding the possibility of medical analysis for such old and problematic documents.

Evolutionary search of the ensemble structures has one advantage that the space of composition structures contains many more types of aggregations than the space of linear combinations. The search in such a large space is important because in testing whether an ensemble can perform better then what is expected from the specialties of its components, it is necessary to check as many different possible combinations of classifiers as possible.

Confusion matrices from the base classifiers and ensembles are looked in order to check where each classifier, ensemble or not, is better than another classifier and to establish if an ensemble underperforms or outperforms its components' strengths. Some of the ensembles end up with accuracy better than all of its components in all of the classes, although some underperform in one or more classes. The last cases indicate that maybe the search performed was not sufficient, or in other words, could be left running for more time or in more trials.

2 Background

Since the main goal is to analyse the distorted script of the anonymous thirteenth-century scribe known by historians as the Tremulous Hand of Worcester, the work followed a very strict path in order to understand and extract the data necessary for the experiments. The writing, though neat and carefully-executed, has a visible tremor, which is especially pronounced in long vertical strokes.

In a previous study, Thorpe and Alty [13] made a close visual scrutiny of the features of the script, which indicated that he had essential tremor—a common condition which in handwriting manifests itself as a fine and regular-amplitude tremor, with a relatively fast frequency. The study concluded that greater insight into the dynamic features of this scribe's writing, especially the speed of writing, could give this diagnosis more certainty.

Thus, it was decided to explore the flood-filling technique for the feature extraction in this case, because the text segmentation is a very important pre-processing step for this work. This technique extracts the flood-filling of the text pages in order to identify the letters. Flood-fill is a graph algorithm, for which the input is an undirected graph with values stored in its vertices [14]. The graph is traversed in such a way that adjacent vertices with the same value are considered to be in the same cluster. The result of this algorithm is a set of groups in which every group contains connected vertices with equal value.

Images can be represented as rectangular grids, in which every pixel is represented by a vertex with the pixel's intensity and pixels sharing a side or point having a correspondent edge between their vertices. Therefore, flood-fill can be applied to images to separate groups of pixels of the letters from the pixels of the background of an image. Such separation can be improved if a binary version of the image is fed to the algorithm.

3 Using ensembles with evolutionary algorithm and neural networks

In the context of handwritten character classification, the use of ensembles derived from aggregation trees generated by evolutionary algorithm is investigated.

Ensembles are known for their capacity to make generalizations based on classifiers that are individually less accurate. Accordingly to Murphy [11], ensemble classification is the aggregated classification of the base classifiers, like weighted voting, although a space of more complex compositions is explored, that is the space of aggregation trees (see Fig. 1).

Opitz and Maclin [9] define bagging classifiers as an ensemble method in which every classifier is trained on a random redistribution of the data. In this type of training, samples are randomly drawn from a data set and the base classifiers are trained in these subsets of data. It differs from boosting in that new classifiers are trained on data sets

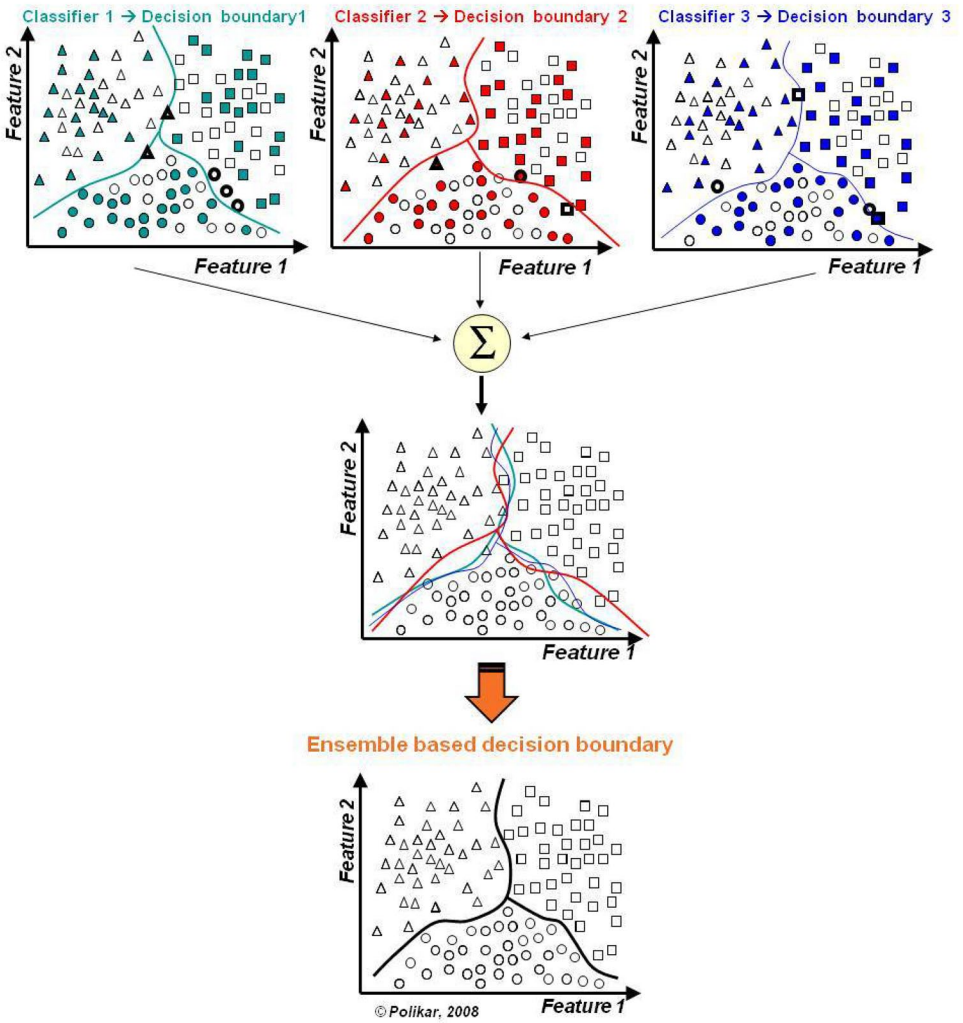


Fig. 1 Ensemble border as a combination of its classifiers' borders

chosen based on the performance of previous classifiers in a series [9].

Ashlock [1] defines evolutionary algorithms as computation which uses algorithms for operating over data structures by selection and variance. In such algorithms, a population of data structures is created randomly at the beginning or is the output of other algorithms. Every individual is represented by a sequence of characters called genes. The population is updated iteratively by replacing the less fit data instances with the more fit ones. In this context, fitness is a function that maps the genes of individuals to a element in an ordered set, like a real number [1] (see Fig. 2).

Because ensemble classifiers can be seen as a composition of operations over the classification from other classifiers, and because it is possible to represent composition of operators as trees, one can search for such trees of operators with evolutionary algorithms, as these trees are just a particular case of data structure.

Neural networks are used as base classifiers in this work. They are widely applied to image classification tasks, and

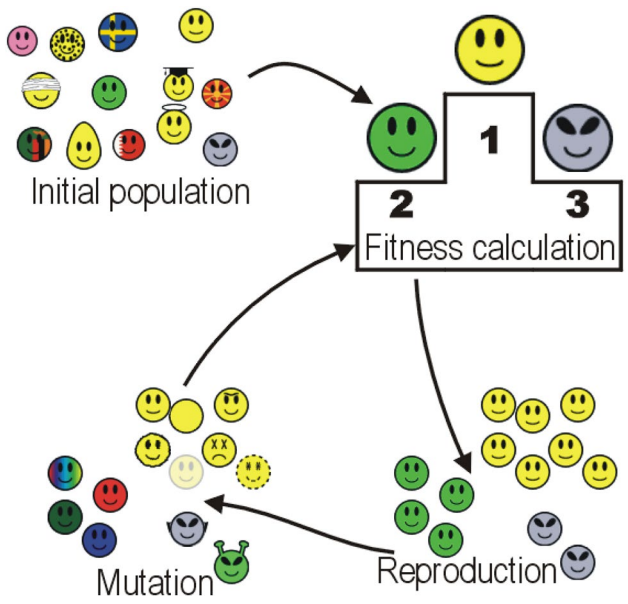


Fig. 2 Illustration of evolutionary search

when it comes to handwritten character classification, simple architectures can produce good results, as is repeatedly demonstrated with the MNIST data set. For instance, an artificial neural network with few layers of convolutions is capable of achieving higher than 90% accuracy on MNIST [7].

According to Gurney [4], an Artificial Neural Network is an interconnected system of processing units that are loosely based on animal neurons. The different functions an ANN can perform are characterized by the strength in the connections between its units. That means that the network connections can be adjusted in order to approximate functions [15]. The neurons in an ANN receive signals from other neurons or from input and forward a signal based on how strong the cumulative signal received is. This is implemented with activation functions, like a TLU and sigmoid, although recent implementations prefer ReLUs.

In the Multi-layer perceptron model, neurons are divided in layers, which may perform different transformations on their input. In this configuration, input is forwarded layer by layer until the final layer gives the network's output.

Given a task to be solved, we can define a cost function, which associates each candidate set of parameters from the network to a real number, and no configuration of parameters can have its cost less than the optimal function. This is a very important concept in learning, as it measures how distant a configuration of the network state is from the optimal solution.

One of the most popular methods in the optimization of ANNs is backpropagation. With a defined cost function, backpropagation can be used to update the network's parameters based on the gradient of the cost function of a given set of inputs [8]. For instance, in a supervised learning scenario, the cost function can be defined as the euclidean distance between the network's output and the true output for the given input. Then the backpropagation algorithm would apply gradient descent or some variant for updating the parameters. On the other hand, one could also define cost functions based on other factors, like a reward function in the context of a simulation. In this scenario, more-or-less a reinforcement learning situation, the cost function could be the penalization given in the simulation. Backpropagation would then proceed in the same way as in the supervised case, updating the model's parameters based on the gradient of the cost function with respect to the parameters.

One type of ANN that has been receiving considerable attention in recent years is Convolutional Neural Networks, due to its ability to learn high level features, especially in images and spatially distributed data. This architecture is a feed-forward neural network that has special layers of convolution and pooling and is inspired in the connection patterns of the human visual cortex [10]. One of the biggest advantages of CNNs is the ability to learn image features that generally need to be manually engineered in other methods. Convolutional layers have this name because they are basically a set of convolution filters that "slide" through the input, simulating the visual stimuli of a neuron. When combined, the stimuli of the neurons from the first layer form the receptive field.

Other than convolutional layers, CNNs generally employ pooling layers for dimensionality reduction through sampling, after convolutions. A pooling layer combines the output of a group of neighbouring neurons into a single signal [2]. One very known type of pooling is max pooling, which chooses the highest value from a group of neurons.

4 Related work

Since we are investigating the use of ensembles obtained via evolutionary algorithm for identifying individual letters in old manuscripts, it is important to understand how this sort of problem is solved in the literatuure for, at least, handwriting of highly noisy documents. Thus, this section will present the main techniques used in this sort of situation.

Smith et al. [6] examine generation of ensembles with Evolutionary Algorithms in two levels: first, the base classifiers are obtained from evolutionary runnings; and second, an expression tree for the aggregation of the trained classifiers is also obtained via evolutionary algorithm. They employ a basic approach, without using modern techniques like coevolution or multi-evolution optimization, but with deterministic, probabilistic crowding, and single-point mutation operators. Similarly to this work, their ensembles are created from aggregation trees where the nodes are the primitive operations and the trees are the more complex compositions. The ensemble style applied is stacking, via stratified cross-validation during the component classifiers training. One important difference between their work and the present work is the fact that we use evolutionary search only for finding the final aggregator, while they also create the base classifiers from evolutionary search. They also used data from the UCI repository, which is, in number of data sets and instances, much more than the data used in this work.

Sylvester and Chawla [12] combine a heterogeneous set of learning agents in a meta-agent generated by evolutionary algorithm with accuracy as fitness score. In their study, the ensemble is obtained in a stacking fashion and the evolutionary search looks for weights for the votes of the simpler classifiers using uniform crossover and best-fit selection. They make experiments for demonstrating that the aggregated model outperforms the best individual agent and also the majority voting. In contrast with this work, they search for ensembles in a more limited space than the space of aggregated operations, that is the space of weights based aggregations. This is done by representing the candidate individuals of the evolutionary process

by a 0–1 normalized vector of real numbers. The size of the vector is equal the number of base classifiers, one element for each base classifier, representing the classifier influence on the final decision.

Gagné et al. [3] present an ensemble learning approach in which a population of base classifiers is acquired by evolutionary search and the ensemble is formed greedly by selecting the classifiers with higher margin histogram from a pool of learned classifiers. The evolution of base classifiers is guided by the optimization of a multimodal fitness function, which takes into account both accuracy and variance in the population. The selection of classifiers for the ensemble is done in two ways: online and offline. In the first case, the classifiers are selected iteratively while the evolution process happens, whilst in the second case the selection is made at once in the end. Both methods employ classification margin as the measure of quality of the base classifiers. Gagné et al.'s study is different from this one in its emphasis on diversification of classifiers, which is not based on the structure of the classifiers and hyperparameters, but mainly on the co-evolution strategy. It also differs greatly in the selection process, because their selection is centered on the base classifiers and majority voting, while the current selection is based on the aggregation aspect of the ensemble with fixed classifiers.

Kim et al. [5] propose a meta evolutionary system in which initial classifiers compete by trying to classify data instances and ensembles compete for base classifiers. The ensembles evolve by reward based on their performance, and for the base classifiers, the more a data point is misclassified, the greater is the reward for correctly classifying that point. The diversity of the classifiers is not based on data subsampling, but on the selection of subsets of features instead. Their main focus is to obtain small optimal classifiers in a two-level evolutionary environment, whilst in this study the base classifiers are already trained and the evolutionary level is for the candidate ensembles alone.

It is obvious that none of the works in the literature deal specifically with the problem of the very noisy old manuscript data analysis. Our approach of combinig evolutionary ensebles is, to the best of our knowledge, being used for the first time for this task.

5 Methodology for medieval letter and scribe classification

In this section, some details of the methods and practices present during the development of the current work are shown. The objective is to allow the reader to understand by which means this work could be reproduced and possibly to be instructive about the process of creating a similar framework.

5.1 The data used

This analysis of handwritten text is built on the character level; therefore the data set consists of characters cropped from images of manuscripts. It has been necessary to clean the background of certain characters, to enable pixel density extraction. Characters were extracted from:

(a) a sample of writing by the thirteenth-century 'Tremulous Hand of Worcester' (see Figs. 3 and 4)
(b) a sample of writing by a non-tremulous thirteenth century scribe and,
(c) a reproduction of the text from the tremulous writer by a modern expert calligrapher, whom writes with perfection.

In specific contexts where it is desired to differentiate between specific authors, the data available may be very limited in quantity. In the current situation for the study of such cases, the source of the data is just three pages of handwritten text and it had fewer than one thousand letter samples, considering just the four vowels used in this work. In cases like this, it is common to use data augmentation, like image transformations that preserve the letter in the image, but as one of the objectives of this work is to test the ability of ensembles to generalize to unseen data, we ignored data augmentation and focused on creating ensembles and

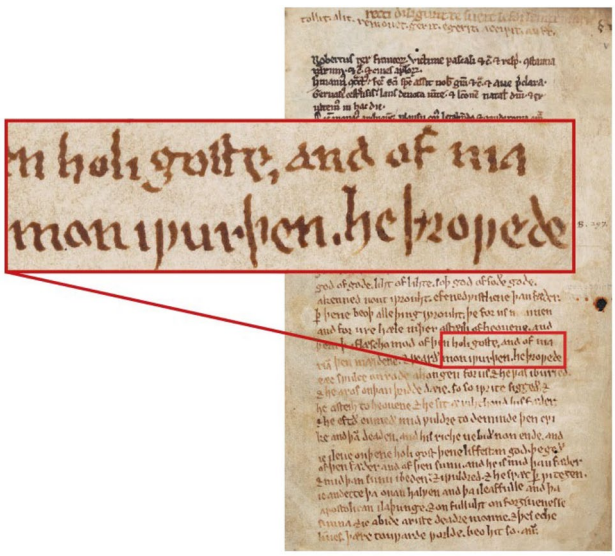


Fig. 3 Page from the Tremulous Hand of Worcester *Source*: Detail of Oxford, Bodleian Library, Manuscript Junius 121, Folio vi recto

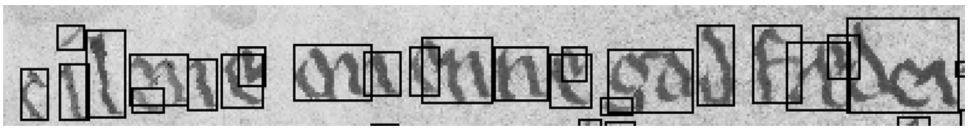


Fig. 4 Letter segmentation obtained after applying *flood fill* to a medieval manuscript page *Source*: Detail of Oxford, Bodleian Library, Manuscript Junius 121, Folio vi recto

testing how much better they can be than their component classifiers.

5.2 Libraries and implementation details

The programming language chosen for the implementation of the models is Python. This language is one of the most prefered by the data science community, if not the most. Widely known libraries that have data processing like image processing (CV2), data formatting (numpy, pandas), and machine learning (Scikit Learn, Tensorflow, Keras, PyTorch, Gensim, Deap) all have interfaces for Python. Also there is the fact that plenty of material exists on the Internet regarding Python implementation, and all the main libraries mentioned are very well documented. This is a very good factor in favor of this language, although someone looking for performance may look into C++ or Lua. That was not the case in this work and Python was sufficient in terms of performance in conjunction with the hardware used.

The library chosen for image processing is OpenCV. OpenCV is a very versatile library for manipulating images. Usually, what is needed to load an image or apply some filter or transformation to an image is one line of code. OpenCV is widely known and has great documentation and support materials on the Internet. In this work, OpenCV was used to binarize images that were fed to Flood-fill, to extract segments obtained from Flood-fill, to resize the images so they have the same dimension as the input layer of the machine learning models used.

Tensorflow is the chosen library for the implementation of the neural networks because of the author's prior experience with the library and because of the big support this library has on the Internet. Tensorflow is very competitive among the other libraries of the kind, like Keras, PyTorch, and Cafe, and has been used for both research and production.

Almost all the known neural networks models can be implemented in Tensorflow, as it has the known layers, activation functions, cost functions, and optimization methods necessary for the majority of the existing architectures. Tensorflow also has support to parallelism and its internal model graph divides the computations during training so to take advantage of the processor cores. Also, Tensorflow's Tensorboard can be very handy to check training processes, as it displays the learning optimization graphs, like the decay of error function during training.

One very important feature of Tensorflow, that is actually present in other libraries too, is saving models during training and loading them at some point later to continue training, or use them as the final classifiers for research or for production.

The plot library Matplotlib is used for plotting ink density graphs during the phase of feature preparation. Like OpenCV, Matplotlib is very versatile and has good documentation and material on the Internet.

The Evolutionary Algorithm search employed in this work is done with the help of the Deap library. Deap is a library for genetic programming and evolutionary search and has support to parallelization, although a superficial configuration can lead to concurrency problems. In this work it was not necessary to harness the parallelism capability of Deap. Parallelism from Tensorflow is default and its configuration was left intact.

Deap has an easy to understand structure when it comes to genetic programming. All that is necessary to set up an evolutionary algorithm environment is defining the set of primitive operations, which compounds the genes, the mate, mutate, and select operators, along with some other parameters, like probability of mutation, initial population size, and number of generations.

5.3 Image processing

This process has been manual with the help of a text segmentation technique for more effort efficient preparation. All manuscript images have been converted to grayscale prior to applying image processing techniques, as the default colors do not bring advantage that compensates the higher dimensionality of the data, and the image representation was a matrix of integers in the range 0–255 as usual. Binarization of the images is obtained by thresholding values higher than 160 in intensity. This threshold was manually chosen among other tested values by visual inspection. Such images are good for the flood-filling process because they have a small number of grey levels, which is two.

The letters in the documents were in part cropped manually, but in order to expedite the preparation of the data set, text segmentation was applied based on grid flood-fill

after a binary filter to detach the characters. However, it is not always possible for Flood-fill to extract single characters. Joined characters inside a word (or even between words) and multiple-stroke characters pose the biggest problems when attempting to extract letters, since they return large chunks of letters in the first case and pieces of letters in the second. For this reason, manual adjustments are required.

Prior to applying flood-fill on an image, binarisation can be improved by filtering the image with threshold, median, or Gaussian filters. These filters may fit together separated strokes of a letter and remove noise. This results in a better separation of characters from background in almost all cases and more connected strokes in some cases.

It is important to recognize that though a binary version of the text image is used in the algorithm, this does not mean that the characters extracted have to be binary images, because information will be important for the positions in which the pixel clusters are located. Thus, after running flood-fill on an image, we compute the rectangle coordinates of the pixel clusters and use them to crop the letters in the image. As long as the letters are in the same place as in the original image, whether or not the letters will be from the raw image is an open choice.

For pixel density alone (Fig. 5), after the rectangular grids from the images of characters were obtained, anything considered beyond a letter stroke was cleaned, what means that the pixel value was set up to 255. Pixel density was computed for each row in every grid, which yielded a vector of densities for each character image. These values could then be used for generating line graphs of pixel density along rows and comparing the graphs of tremulous writers with the graphs of non-tremulous writers.

5.4 Feature selection

Both features mentioned in the previous paragraphs could be used in order to represent the input images of characters, but perfect removal of background for producing pixel density consumes much more time than just cropping letters. This is because the source images are so noisy that each background would have to be manually removed. On the other hand, even if the resulting segments of text from flood-fill contain chunks of joined letters or pieces of letters, these are only a fraction of the cases, and the remaining letters can be extracted manually with some image editing software. This means that depending on the classification method used, pixel density may not be feasible due to the amount of prepared data. Take a neural network for instance. If the cropped letters are not sufficient for training the classifier, the data can be augmented via addition of noise and image transformations that preserve the letters. In this case, some letter pixel density vectors cannot be augmented in a similar way and the size of the data set will stay not sufficient for this kind of classifier.

In the case of this work, it was needed to choose between the raw grayscale image and the raw binary image of the characters. Considering the binary images already have large amounts of noise removed and at the

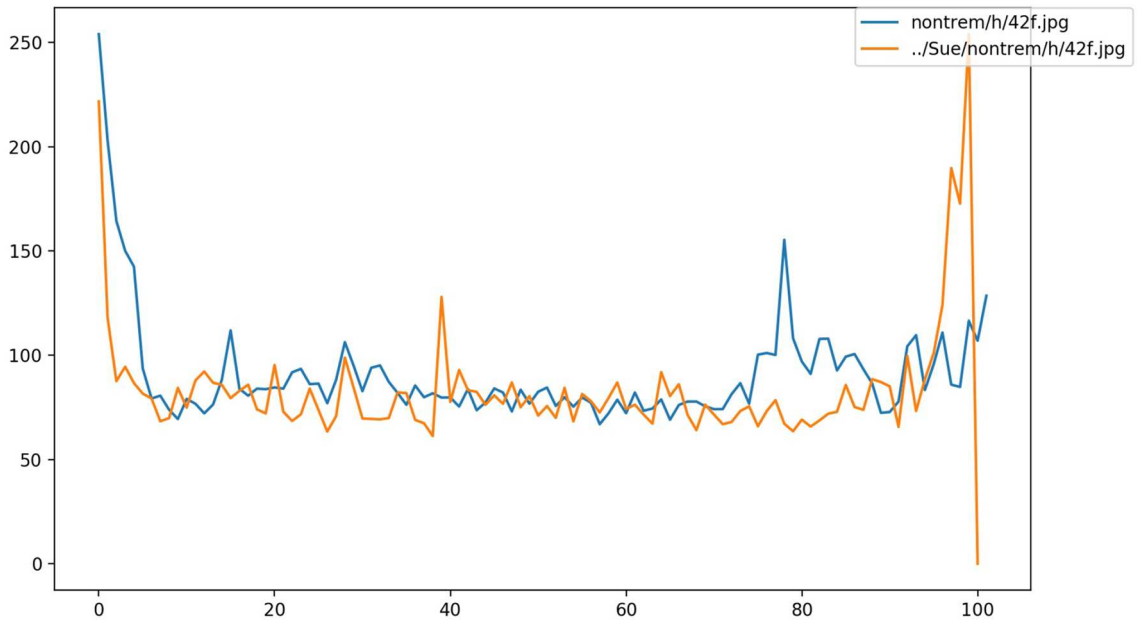


Fig. 5 Pixel density comparison of same letter h between tremulous hand and non-tremulous hand writer

same time the characters are preserved, which was verified by visual inspection, it is expected that the binary versions of the cropped characters work better than the grayscale images. Previous tests on training the models on both grayscale and binary images demonstrated that learning is faster when training with binary images, though it is not guaranteed that higher accuracy will be achieved on this data than if training on the grayscale images. Thus, the binary images were chosen as the input representation.

One very important step is normalizing the pixel intensities in the images, so that all values fall into the range [0,1]. Otherwise, convergence problems can arise when training with the intensity values in the range [0,255].

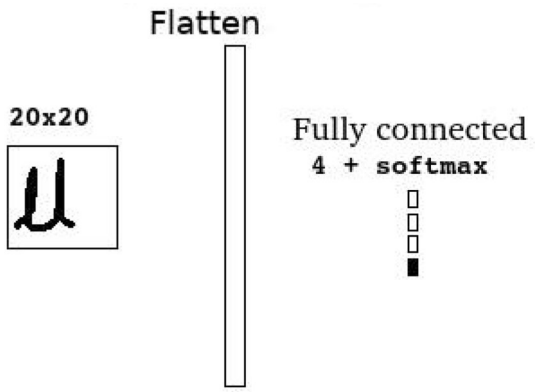


Fig. 7 Shallow architecture used as base classifier

5.5 Base classifiers

For each of the three writers, base classifiers are trained on the respective data set. For the vowel classification task, four base classifiers are obtained for each data set: there are two types of architectures, which are 5-layer convolutional neural network and shallow fully connected neural network, combined with two types of training settings, the first with mini-batches of size 16 and the second with mini-batches of size 64. All models are trained with 5000 training steps.

Figures 6 and 7 illustrate the convolutional and shallow architectures used in more details. The same architectures and combinations were used in the writer identification task, except for the number of unities in the output, which is two for the writer task.

The classifiers are trained on 80% of the data and tested in the remaining data. The learning rate and optimizer used are 0.001 and Adam, respectively.

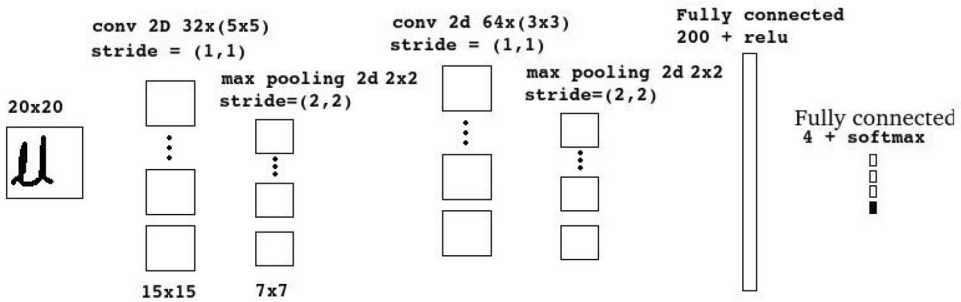
For the vowel task, the defined classes are the vowels a, e, o, and u and the base classifiers are two types of neural networks, one shallow and the other deep, trained with mini-batches of sizes 16 and 64, a total of four combinations for each of the three writers. The features used from the character images are the normalized grayscale (pixel intensity in the range [0,1]) versions of the images after they were reshaped to be 20 pixels by 20 pixels. In the evolutionary algorithm, the candidate ensembles are aggregation trees whose primitive operations are the sum of two classification vectors and the product of a classification vector by a constant in the set {0.0, 0.1, 0.2, ..., 0.9}.

5.6 Ensembles

Evolutionary search of ensembles of pre-trained classifiers is employed in order to improve test classification accuracy after training on a small set (around 300 examples in each of the three data sets) of character images. The tasks assigned are classification in the a, e, o, and u classes in the first case and, for the other task, classification between every pair of the three writers: The Tremulous Hand of Worcester ('TREMULOUS'), the non-tremulous medieval writer ('NON-TREMULOUS'), and a modern calligrapher ('CALLIGRAPHER').

The measure of fitness is the ensemble accuracy on the same test set than the one used for testing the base classifiers. The first population of individuals is generated from a uniform distribution and have trees with depth between 1 and 5. Individuals mate by switching one of their subtrees with some probability (Bernoulli random variable) and express mutation in zero to two of its nodes. The mutation consists of the replacement of a subtree by a tree sampled

Fig. 6 Convolutional architecture used as base classifier



from a uniform distribution. The selection of individuals in one iteration of the algorithm is done by tournaments between pairs of individuals.

In the next two sections, the analysis of both vowel and writer identification tasks are presented. The data from which the analysis are made are the overall accuracy of base classifiers and classwise accuracy of the base classifiers and obtained ensembles.

6 Classification analysis

In this section, the analysis of both vowel and writer identification tasks are presented. The data from which the analysis are made are the overall accuracy of base classifiers and classwise accuracy of the base classifiers and obtained ensembles.

6.1 Evolutionary ensemble of classifiers: vowel task

As expected, which can be seen in Table 1, in all of the three writers the convolutional models outperform the shallow ones or have equal performance in the CALLIGRAPHER data set. The difference in accuracy from the convolutional models to the shallow models in each writer is no higher than 3%, if we do not take into account the 82% accuracy of the shallow model in the TREMULOUS writer. Batch size does not seem to have affected the accuracy of the models in the present configuration, as in each model both choices of 16 and 64 for batch size result in the same accuracy, except for the shallow TREMULOUS models.

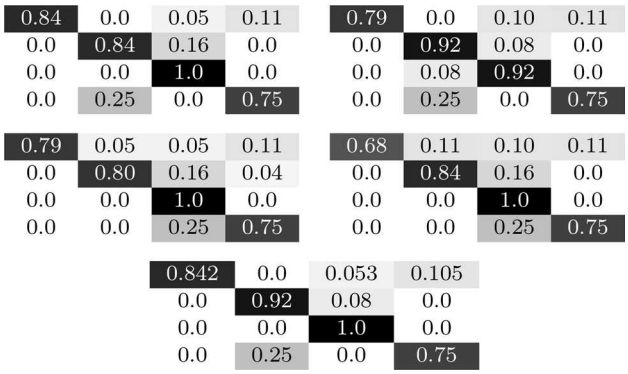
For the TREMULOUS data set, the confusion matrices of its models (Table 2) show that the biggest misclassifications are misclassifying u by e in the convolutional models and miscalssifying u by o in the shallow models. One clear advantage of the convolutional models over the shallow ones is the reduction of misclassification of a by e and also the reduction of misclassification of e by o if we compare the convolutional model of batch size 64 and shallow models. A good ensemble may take this into account and overcome the problems of misclassifying a by e and e by o. The letter o is the best classified in this set, having only 8% misclassification in the convolutional model of batch size 64.

Interestingly, in the data set of the CALLIGRAPHER writer (Table 3), all of the four models performed exactly in the same way, in terms of the values in their confusion matrices. This may be the case that the data samples in the test set have a few pathological instances that get misclassified in all of the four models. This lack of classification

Table 1 Base classifiers accuracy for the vowel identification task

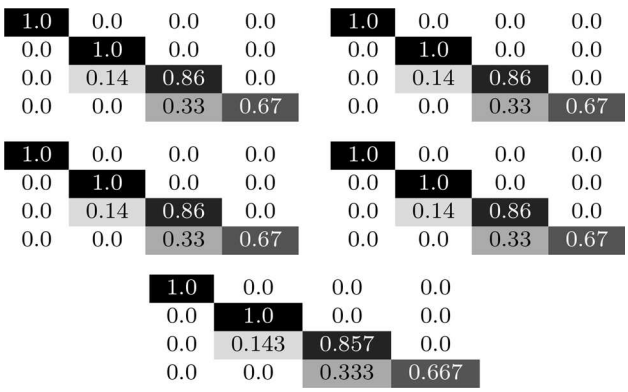
Classifier	Data	Batch size	Accuracy
Conv.	TREMULOUS	16	0.87
Conv.	TREMULOUS	64	0.87
Conv.	CALLIGRAPHER	16	0.94
Conv.	CALLIGRAPHER	64	0.94
Conv.	NON-TREMULOUS	16	0.98
Conv.	NON-TREMULOUS	64	0.98
Shallow	TREMULOUS	16	0.84
Shallow	TREMULOUS	64	0.82
Shallow	CALLIGRAPHER	16	0.94
Shallow	CALLIGRAPHER	64	0.94
Shallow	NON-TREMULOUS	16	0.96
Shallow	NON-TREMULOUS	64	0.96

Table 2 Confusion matrices from the TREMULOUS data set



The top two matrices are from the convolutional models and the middle two matrices are from the shallow models. The models trained with batch size 16 are on the left, while the ones trained with batch size 64 are on the right. The matrix on the bottom represents the best ensemble obtained in the search. Rows represent the true labels a, e, o, u from top to bottom and columns represent the labels given by the classifiers a, e, o, u from left to right

Table 3 Confusion matrices from the CALLIGRAPHER data set



The top two matrices are from the convolutional models and the middle two matrices are from the shallow models. The models trained with batch size 16 are on the left, while the ones trained with batch size 64 are on the right. The matrix on the bottom represents the best ensemble obtained in the search. Rows represent the true labels a, e, o, u from top to bottom and columns represent the labels given by the classifiers a, e, o, u from left to right

variety still may result in an ensemble that is better than its components if the individual instancies are not all classified in the same way by all base classifiers. The fact that one-third of the u's are misclassified for o by these classifiers just reflects the fact that many u's in this data set are closed in the upper part. Such a difference of classification accuracy from the TREMULOUS data set is explained by the quality of the data available, as the characters from the TREMULOUS data set are the ones with much more noise than the others: the data sets CALLIGRAPHER and NON-TREMULOUS are much cleaner than TREMULOUS in terms of background noise.

In contrast with the other classifiers, the shallow classifier trained with 16 samples per batch showed misclassification of a by o in 8% of the cases in the NON-TREMULOUS data set (Table 4), but did equally well on the other letters. As in the CALLIGRAPHER set, this base classifier will not result in a better ensemble depending on the obtained uniform misclassification patterns, but the classification of the individual samples may prove this wrong. Similarly to the results from the CALLIGRAPHER data set, the classifiers showed almost the same classification patterns, such as correctly classifying all the e's, and u's in the test set and, except for the shallow classifier of batch size 16, they also correctly classified all the letters a in the test set. Again, there is a noticeable difference in accuracy from the TREMULOUS set, with the reason being the same as for the CALLIGRAPHER set.

The best ensemble found for the TREMULOUS data set is, overall, around 3% better than its best component, as it can be seen in Table 5. The learned ensemble seems to have inherited the best from its components regarding the letters a, e, and o, but could not overcome the u misclassification problem while ideally the components could correct each other's u assignment in the sense that their errors on u are complementary. This indicates that this is a sub-optimal ensemble and that the search should be performed with different settings, or with more steps, in order to find an ensemble that takes full advantage of its components.

For the CALLIGRAPHER data set, the best ensemble was as good as its components. Also, by comparing the confusion matrix of this ensemble with the ones of the components, one notices that the assignment pattern is the same. This is not surprising given the uniformity of its classifiers in terms of assignments. This strongly suggests that it will be possible to develop a better ensemble only if a more diverse pool of classifiers is out in place of the current one.

From the NON-TREMULOUS data set, the chosen ensemble was equally good as the best components. This was almost the case with the CALLIGRAPHER set, but here the components have a small variety. Although it was expected from the ensemble to overcome the fraction of wrong e assignments, that did not happen. As in the TREMULOUS

Table 4 Confusion matrices from the NON-TREMULOUS data set

1.0	0.0	0.0	0.0
0.0	1.0	0.0	0.0
0.0	0.0	0.90	0.10
0.0	0.0	0.0	1.0

1.0	0.0	0.0	0.0
0.0	1.0	0.0	0.0
0.0	0.0	0.90	0.10
0.0	0.0	0.0	1.0

0.92	0.0	0.08	0.0
0.0	1.0	0.0	0.0
0.0	0.0	0.90	0.10
0.0	0.0	0.0	1.0

1.0	0.0	0.0	0.0
0.0	1.0	0.0	0.0
0.10	0.0	0.90	0.0
0.0	0.0	0.0	1.0

1.0	0.0	0.0	0.0
0.0	1.0	0.0	0.0
0.0	0.0	0.90	0.10
0.0	0.0	0.0	1.0

The top two matrices are from the convolutional models and the middle two matrices are from the shallow models. The models trained with batch size 16 are on the left, while the ones trained with batch size 64 are on the right. The matrix on the bottom represents the best ensemble obtained in the search. Rows represent the true labels a, e, o, u from top to bottom and columns represent the labels given by the classifiers a, e, o, u from left to right

Table 5 Base classifiers accuracy for the writer identification task

Classifier	Data	Batch size	Accuracy
Shallow	TREMULOUS, NON-TREMULOUS	16	0.667
Shallow	TREMULOUS, NON-TREMULOUS	64	0.667
Conv.	TREMULOUS, NON-TREMULOUS	16	0.711
Conv.	TREMULOUS, NON-TREMULOUS	64	0.702
Shallow	TREMULOUS, CALLIGRAPHER	16	0.598
Shallow	TREMULOUS, CALLIGRAPHER	64	0.652
Conv.	TREMULOUS, CALLIGRAPHER	16	0.793
Conv.	TREMULOUS, CALLIGRAPHER	64	0.772
Shallow	CALLIGRAPHER, NON-TREMULOUS	16	0.714
Shallow	CALLIGRAPHER, NON-TREMULOUS	64	0.702
Conv.	CALLIGRAPHER, NON-TREMULOUS	16	0.893
Conv.	CALLIGRAPHER, NON-TREMULOUS	64	0.881

set, this may be due to limited search on the space of aggregation trees in the evolutionary search.

One thousand generations in the evolutionary running resulted in ensembles considerably better than the base classifiers for the TREMULOUS data set. Even though the amount of data available was limited, th resulting ensembles were able to make reasonably better predictions on the testing data than the base classifiers, without using any data augmentation for this set of data.

6.2 Evolutionary ensemble of classifiers: writer task

For the writer task, in both architectures and pairs of writers, except for the shallow classifiers in the TREMULOUS–CALLIGRAPHER pair, raising the batch size from 16 to 64 did not increase the classifier accuracy in the current setting, like in the vowel classification task. This may be because the number of training steps is too small to show the gains in accuracy of training with bigger batch sizes.

It is debatable whether changing the batch size alone produces diverse classifiers. In the vowel classification tests it seemed that changing batch size did not help significantly because there was no significant variance in the classification between the same architecture trained with batches of sizes 16 and 64. This is checked again in the results of the writer task tests that follow. In each pair of writers, the convolutional models performed considerably better than the shallow models—sometimes as much as 20%. Still, the existence of cases where the shallow classifiers perform better than the convolutional classifiers can allow the creation of even better ensembles than the best convolutional classifier at hand. The analysis of the classifiers in each pair of writers that follows can give some insight on how good an ensemble from the trained classifiers can be.

The task of differentiating between the writers TREMULOUS and NON-TREMULOUS was performed with interesting results: all the base classifiers showed difficulty in identifying the letters of the TREMULOUS writer whilst identifying the letters of the NON-TREMULOUS writer seemed significantly easier. This happens despite the existence of only two classes because the neural networks are not trained to learn the "law of excluded middle", so they do not know that if a sample does not belong to a class, then it belongs to the remaining class.

One can expect an optimal ensemble to combine the best predictor for the writer TREMULOUS, that is the shallow model of batch 64, with the best predictor for the writer NON-TREMULOUS, that is the convolutional model of batch 16. We now proceed to check this from the classification pattern of the respective ensemble.

In each class, the best resulting ensemble underperformed some of its components, although the overall accuracy of the ensemble was higher than all of its components. More specifically, in the class TREMULOUS, the ensemble accuracy was around 4% less than the best component model. In the class NON-TREMULOUS the difference from the best component was around 5%. This suggests that the search performed was too limited, as an ideal ensemble would behave in each class at least as good as the best component in that class.

As in the TREMULOUS, LOUS pair (Table 6), the classifiers of the TREMULOUS, CALLIGRAPHER pair had more success in identifying the writer that is not TREMULOUS, except for the shallow model trained with batch size 64. This difficulty in correctly assigning the samples from the TREMULOUS set comes from the noisy aspect of its letters. This is not the case with the other two sets, in which the samples are cleaner.

The best ensemble for the class TREMULOUS this time was the convolutional model of batch size 16, while for the class CALLIGRAPHER it was both convolutional models. It is possible that none of the shallow models is needed in order to find the optimal ensemble from this group of classifiers, as the convolutional models outperform the shallow ones considerably in both classes.

The best ensemble found for the pair TREMULOUS, CALLIGRAPHER (Table 7) performed better than all of its components in the TREMULOUS class and performed equally well to its best component in the CALLIGRAPHER class. As this ensemble had higher accuracy than all of its component classifiers in the CALLIGRAPHER class, we can infer that the base models that scored less in this class actually contributed to the final ensemble because of individual samples. This means that there was specialization of the weaker classifiers in instances that the best ones did not master.

Table 6 Confusion matrices from the pair TREMULOUS, NON-TREMULOUS

0.475	0.525
0.113	0.887

0.639	0.361
0.302	0.698

0.492	0.508
0.038	0.962

0.525	0.475
0.094	0.906

0.59	0.41
0.09	0.91

The top two matrices are from the convolutional models and the middle two matrices are from the shallow models. The models trained with batch size 16 are on the left, while the ones trained with batch size 64 are on the right. The matrix on the bottom is from the best ensemble obtained. Rows represent the true writers TREMULOUS, NON-TREMULOUS from top to bottom and columns represent the assigned writers TREMULOUS, NON-TREMULOUS from left to right

Table 7 Confusion matrices from the pair TREMULOUS, CALLIGRAPHER

0.508	0.492
0.226	0.774

0.656	0.344
0.355	0.645

0.754	0.246
0.129	0.871

0.721	0.279
0.129	0.871

0.787	0.213
0.129	0.871

The top two matrices are from the convolutional models and the middle two matrices are from the shallow models. The models trained with batch size 16 are on the left, while the ones trained with batch size 64 are on the right. The matrix on the bottom is from the best ensemble obtained. Rows represent the true writers TREMULOUS, CALLIGRAPHER from top to bottom and columns represent the assigned writers TREMULOUS, CALLIGRAPHER from left to right

Table 8 Confusion matrices from the pair CALLIGRAPHER, NON-TREMULOUS

0.677	0.323
0.264	0.736

0.645	0.355
0.264	0.736

0.935	0.065
0.132	0.868

0.935	0.065
0.151	0.849

0.935	0.065
0.094	0.906

The top two matrices are from the convolutional models and the middle two matrices are from the shallow models. The models trained with batch size 16 are on the left, while the ones trained with batch size 64 are on the right. The matrix on the bottom is from the best ensemble obtained. Rows represent the true writers CALLIGRAPHER, NON-TREMULOUS from top to bottom and columns represent the assigned writers CALLIGRAPHER, NON-TREMULOUS from left to right

Curiously, in the CALLIGRAPHER, NON-TREMULOUS (Table 8) pair of writers, the shallow models worked better on correctly assigning the NON-TREMULOUS labels, while the convolutional models did better in assigning the CALLIGRAPHER labels. Even though it sounds like a desirable case of specification, what is going to be checked in the best obtained ensemble, this may not be the case because the convolutional classifiers outperformed the shallow models in both classes. In other words, the resultant ensemble may take advantage only of the convolutional models, although it is not strictly necessary, as some individual samples can be the specialty of the shallow models instead of the convolutional models.

Similarly to the TREMULOUS, CALLIGRAPHER pair, in the CALLIGRAPHER, NON-TREMULOUS pair the ensemble performed better than all of its components in one of the classes, that was the NON-TREMULOUS class, and performed as good as its best components in the other class. Again, the weaker classifiers in one of the classes, the NON-TREMULOUS class, contributed with the specialization in some instances that the stronger models could not assign correctly. The resulting ensemble is, therefore, better than if it was the vote of the best ensemble in each class.

Moreover, our empirical results have corroborated with the theoretical conclusions presented in [13, 14], which indicated that the dynamic features related to the speed of writing of this scribe's writing can be used as diagnosis.

7 Conclusion

The use of ensembles for generating robust predictors and classifiers is a common practice nowadays due to the capacity of generalization that ensembles have. Although the classification improvement of ensembles over the base models is nothing new, it is useful to have an idea of how good an ensemble can be based on which classes its components perform better. This comes from the assumption that an ideal ensemble should be at least as good on a given class than its best components on that class.

The results of aggregating classifiers through evolutionary algorithm are investigated in relation to the base classifiers, that were trained on a relatively small quantity of data. The classification patterns of the base models in the form of confusion matrices are analysed and compared to those of the respective ensembles. This analysis gives some intuition on whether each ensemble, in the context of the search settings in which it was generated, took advantage of the base classifiers that perform the best in each class.

In this work a framework for identification of authors and letters of medieval scripts was developed, having in mind that the amount of data in similar contexts may be small. In order to do so, some techniques for preprocessing, classification, ensembling, and search were used.

Some results on a specific data set were analysed. In the performed tests, it was verified that in the majority of the tasks the best ensembles from the evolutionary algorithm have slightly better prediction on our data set, although in some cases it seemed that the parameters of the evolutionary search could be adjusted to find better ensembles, as the best ensembles perform worse in some classes than the best component, even though its overall accuracy was higher than all of its components.

Acknowledgements We would like to acknowledge the great contributions of Dr. Deborah Thorpe on providing the dataset used in this work. This research was the result of the project entitled "Securityfor all: Using smart technologies to promote security for citizens" supported by Newton Research Collaboration Programmeof the Royal Engineering Academy (NRCP1516/1/135).

References

1. Ashlock D (2006) Evolutionary computation for modeling and optimization. Springer, New York
2. Cireşan DC, Meier U, Masci J, Gambardella LM, Schmidhuber J (2011) Flexible, high performance convolutional neural networks for image classification. In: Proceedings of the twenty-second international joint conference on artificial intelligence, IJCAI'11. AAAI Press, vol 2, pp 1237–1242
3. Gagné C, Sebag M, Schoenauer M, Tomassini M (2007) Ensemble learning for free with evolutionary algorithms? CoRR. arXiv :abs/0704.3905
4. Gurney K (1997) An introduction to neural networks. UCL Press, London
5. Kim YS, Street WN, Menczer F (2006) Optimal ensemble construction via meta-evolutionary ensembles. Expert Syst Appl 30(4):705–714
6. Lacy SE, Lones MA, Smith SL (2015) Forming classifier ensembles with multimodal evolutionary algorithms. In: 2015 IEEE congress on evolutionary computation (CEC). pp 723–729
7. Lecun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324
8. Lecun Y (1992) A theoretical framework for back-propagation. IEEE Computer Society Press, Washington
9. Maclin R, Opitz DW (2011) Popular ensemble methods: an empirical study. CoRR. arXiv:abs/1106.0257
10. Matsugu M, Mori K, Mitari Y, Kaneda Y (2003) Subject independent facial expression recognition with robust face detection using a convolutional neural network. Neural Netw 16:555–559
11. Murphy K (2012) Machine learning: a probabilistic perspective. MIT Press, Cambridge, p 580
12. Sylvester J, Chawla NV (2005) Evolutionary ensembles: combining learning agents using genetic algorithms. American Association for Artificial Intelli-gence, pp 46–51. https://www.aaai.org
13. Thorpe DE, Alty JE (2015) What type of tremor did the medieval 'tremulous hand of worcester' have? Brain 138(10):3123–3127 awv232[PII]
14. Torbert S (2014) Applied computer science. Springer, New York, p 158
15. Zell A (1994) Simulation neuronaler Netze. Addison-Wesley, Bonn Paris Reading