STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LXII**, Number 1, 2017
DOI: 10.24193/subbi.2017.1.04

MACHINE LEARNING TECHNIQUES FOR DETECTING FALSE SIGNATURES

MIHAI TELETIN

Abstract. Deciding whether a handwritten signature is legit or it has been falsified is a very complex task. Several methods have been tried out by the graphology experts in order to detect such fraud. However, it is obvious that it is very hard to perform such a classification. In this paper we investigate the possibility to use some supervised learning techniques in order to build models capable to accurately perform such an analysis. The results reported during the testing phase of the obtained model are encouraging for further work.

1. Introduction

Determining the authenticity of signatures is quite an old task. From an algorithmic point of view, even though the solution for this problem was researched for a long time, a state of the art solution does not exist. Furthermore, designing a computational algorithm to cover the experience of a trained eye of a graphology expert may seem impossible.

Obviously, in some cases this problem may be considered a very critical one. For examples, high costly frauds can be avoided if it can be proven that they are based on forger signatures.

Considering that this task is a very complex one and that an outstanding experience of an expert is needed in order to perform such a discrimination, we can say that the problem can be tackled using learning methodologies. Designing such a complex algorithm from scratch may be much harder and may not manage to highlight all the corner cases. However, *machine learning* approaches are known to be very easily adaptable to changes [10] and are relatively easier to be used when solving complex problems.

Received by the editors: April 26, 2017.

2010 *Mathematics Subject Classification.* 68T05, 62M45.

1998 *CR Categories and Descriptors.* I.2.6 [**Computing Methodologies**]: Artificial Intelligence – *Learning*; I.2.8 [**Computing Methodologies**]: Artificial Intelligence – *Problem solving*.

Key words and phrases. Machine learning, Convolutional neural networks, Support vector machines, Classification, Signature verification.

The aim of this paper is to present a *machine learning* approach proposed for this binary classification, to highlight its performance by testing it against new, unseen data. We consider that *supervised learning* approaches can contribute very well in the area of research focused on this task. Furthermore, recent research has shown that deep learning methods can be successfully applied in this area of research [13][4]. Thus, we propose a method to solve the signature verification problem by using a powerful existing feature extractor which is based on *deep convolutional networks*. The method proposed in this paper is novel since it combines a powerful trainable classifier that is the *support vector machine* with the well known convolutional pretrained model.

The remaining of the paper is structured as follow: in Section 2 we are going to present the problem from a machine learning perspective and to highlight its difficulty. In Section 3 we briefly describe the *models* used for the task, the *support vector machine* on top of the *convolutional network based feature extractor*. A short overview of the related work is also presented in Section 3.2. Our approach is then presented in Section 4. We will explain our *training* and *testing* methodologies. The performance of the model is then analysed in Section 5. Finally, the conclusions of the work are outlined in Section 6.

2. Problem Relevance and Difficulty

The specialists in the domain of graphology have developed an amazing sense of truth when dealing with an amazing diversity of signatures. While some of them can easily be classified by any amateur, there are lots of amazingly imitated signatures that can only be classified as forgery by a trained eye. Also, we can also have authentic signatures that look strange and may fool even an expert to decide that it looks like a false one.

Even though the graphologists are recognized for performing really well this classification, computer scientists have tried to come up with algorithmic approaches that can assist the experts for taking this decision. However, trying to define a set of rules capable to reproduce the natural behaviour of experts is a very challenging task.

Nevertheless, a proper solution for defining an algorithm to recognize such signatures can be expressed by *Machine learning* approaches. We are going to highlight that *supervised learning* approaches such as *support vector machines* combined with *convolutional neural networks* can prove that this problem is solvable obtaining some good results. In this *supervised learning* scenario, the model will improve its performance using a training set consisting of already classified images of signature. This problem will be viewed from a *machine learning* perspective as a *binary classification problem*.

Classification is one type of problem which can be solved in a supervised manner. The aim is to learn to develop approximation functions for an unknown function called *target function*. Classification deals with functions that produce discrete output (a finite set of values, in our case the true classes: *legit* and *forgery*).

For the presented *classification* problem, we are going to learn to directly classify images of signatures. For doing so we will use a pretrained feature extractor that is capable to determine a set of relevant features by processing the images. Using this set of features, we can train our own model capable to solve the discussed problem.

3. Background

3.1. **Machine learning models used.** In this section we are going to briefly present the proposed *machine learning* models used in order to perform the classification, the *support vector machine* and the *convolutional neural network*. Furthermore we are going to summarise some of the related work.

3.1.1. *Convolutional neural networks.* Research results have shown that the full automatic models such as *convolutional neural networks* give much more better performance than performing manual feature extraction [9]. In the classical approach the system was split in two subsystems:

- *feature extraction module* - a module which processes the given shape (the raw input), performs diverse *heuristics* and produces the feature vector which is considered to describe best the image;
- *trainable classifier module* - a module which learns to classify the data using the feature vector as input.

The main problem of this approach is that its performance is directly decided by the ability of the *extraction module* to come up with relevant and correct sets of features [11][9]. Moreover, this module is usually implemented from scratch, making the task of feature extraction very complicated.

One of the main reason for which this approach was no longer considered was the development of more powerful machine learning models which could easily handle high-dimensional input values [9]. It is now preferred to build a full model which extracts features by itself from the data and learns to predict the desired class. However, data preprocessing process specific to the problem may always be needed in order to increase performance [9].

A much better approach is a model capable to associate proper classes given an almost raw input (e.g. an image). Basically, this system will have to learn by itself to perform the feature engineering.

Convolutional neural networks are special types of neural networks mainly used for problems which require working with huge number of features. Unlike

simple neural networks, they tend to manage well feature sets that are correlated. One of their main possible applications is image classification. This is due to the fact that the images are made of several hundred or thousands pixels, having all the pixels in a neighbourhood highly correlated [9].

A convolutional architecture contains some special types of layers that are capable to process complex input space: *convolutional layers* and *subsampling layers* [9]. Each of these layers are composed of several feature maps capable to learn to extract different relevant features [9]. The main advantage of these networks is that they can easily adapt to several different problems. Thus, one of the domains were they are successfully applied is *computer vision* [9][15][11][13].

3.1.2. *Support vector machine.* The support vector machine (SVM) is a supervised learner introduced by Cortes and Vapnik [2]. The original model was intended to be used for binary classification. The main goal of the SVM is to search for the optimal separating hyperplane among data in order to perform the classification. By finding the optimal separating hyperplane, the model is minimizing the risk to misclassify new, unseen data.

In most cases the data on which the model is trained is not linearly separable. In this case the SVM model is extended in order to support soft margins [10]. By doing so we introduce a mapping function, named *kernel*. The kernel function is used to map the input space into a higher dimensional space, where a linear separating hyperplane may be computed. The linear separation in the high dimensional space will lead to a non-linear decision boundary in the (lower dimensional) input space [10]. Several kernel functions are available in the literature and usually used for non-linear SVMs: *linear* kernel, *polynomial* kernel, *sigmoid* kernel, *radial basis function* (RBF) kernel. Furthermore, the SVM is enhanced with a new hyperparameter, C, the *misclassification parameter*. This hyperparameter lets SVM intentionally misclassify some training data in order to improve the performance on testing [10].

3.2. **Related work.** One of the first approaches tested on the dataset that we are using for our models was introduced in [6]. It represents a method based on *statistical analysis* of the features expressed by the signatures. The approach is intended to extract multiple features from images and to compute the probability of belonging in the two discussed classes. The approach was further discussed and extended in [5].

Classification using neural networks and different feature engineering techniques are presented in [8]. The authors try to identify and to eliminate the weak points of the process of analyzing and classifying signatures.

Machine learning approaches are discussed in [14]. The approach uses *statistical learning* methods in order to learn to predict the class based on the similarity of the features between the known samples and the new ones.

Deep learning approaches including *Restricted Boltzman Machines* were used in [13]. The authors have developed a verification system for this task built on a two-step hybrid classifier system. They have proven that deep learning methods are capable to learn very well to extract relevant features with very limited prior knowledge.

4. The proposed approach

We consider that this problem is solvable in a supervised manner since we can use already annotated datasets of images of signatures. In this learning scenario, the model will learn to detect whether an image contains a legit signature or a false one, by analyzing such already annotated examples.

Our approach consists of three steps. First, a *feature extraction* step is applied on the input data. For this step we are using the *Tensor flow inception graph* pretrained model [15]. This model was developed by *Google* and it represents a very complex convolutional neural network which is composed of 59 layers. The model was trained on a considerable set of images and was capable to obtain state of the art accuracies on very complex problems, such as *ImageNet* classification [15]. So, instead of training a new convolutional neural network, we intend to use this pretrained model as feature extractor.

The next step consists of training a classifier on the pre-processed data. More specifically, using the features extracted from our dataset of signatures we aim to build a classifier that will learn to identify the forger signatures based on such input data. A *Support Vector Machine* classifier will be used for discriminating between original and false signatures. The trained *SVM* will be then *tested* in order to evaluate its performance.

In the following we will detail the steps of our approach.

4.1. **Feature extraction.**

Our dataset consists of annotated images of signatures. From a mathematical point of view, a colored image can be viewed as a 3 dimensional matrix, containing the values of the pixels in the *RGB* code. Obviously, the dimensionality of such data is huge. Thus, directly applying a learner such as *SVM* may be impossible.

Feature extractors are intended to analyze such huge input spaces and come up with a drastically lower set of features which are as representative as possible for the original input space.

We intend to integrate the previously highlighted model, *inception* as feature extractor. By doing so, we use the model in order to extract the desired

set of features, named *bottleneck* features. This name suggests that the features are coming from the latter layers of the model, making them as abstract as possible.

4.2. **Training.** On the set of extracted features, a SVM is trained. In order to do so, we take all the available samples in the dataset and we apply the feature extractor. Furthermore, the obtained set of instances (vectors of features) is split in 2 sets: training and testing.

In order to train the model, several hyperparameters are used, such as C, the kernel function, the parameters of the kernel (e.g. γ for the RBF kernel). For optimizing the hyperparameters, a grid search is performed in order to find the best suited ones on a 10-fold cross validation approach. The grid search performs repeated trials for each parameter across a specified interval using geometric steps. The quality of a combination is computed as the average of the accuracy rates estimated for each of the 10 divisions of the dataset.

4.3. **Testing.** The performance of the trained SVM model will be tested on a testing set completely disjoint from the training dataset. The testing phase will be performed on unseen data.

Since the considered problem is a binary classification one, the *confusion matrix* will be computed. For building the confusion matrix and computing the measures, we consider that the *forger* signatures are representing the *positive* class while the *negative* class is represented by the *original* ones. A large number of different performance metrics can be computed from the confusion matrix. The *accuracy* (Acc) (Formula 1) is often used, but it is not suitable in the case of imbalanced datasets.

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

For better evaluating the performance in case of imbalanced data, the *Area under the ROC curve* (AUC) measure [3] is used in the literature as a more relevant evaluation measure. For our classifier, the output is directly the class label, thus on the ROC curve there is one single (Pf, Pd) point that can be linked to the $(0, 0)$ and $(1, 1)$ points and the area under this curve can be computed using Formula 2.

$$AUC = (1 - Pf) * Recall + \frac{Pf * Recall}{2} + \frac{(1 - Pf) * (1 - Recall)}{2} \tag{2}$$

In Formula (2), the *recall* also known as *probability of detection* (Pd) is computed as $Recall = \frac{TP}{TP+FN}$ and the *probability of false alarm* (Pf) is $Pf = \frac{FP}{FP+TN}$. *F-measure* will also be reported as an evaluation measure for

the classification task. It is computed as the harmonic mean of *precision* and *recall*, as shown in Formula (3). The *precision* of the classification is expressed as $Precision = \frac{TP}{TP+FP}$.

$$F - measure = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{3}$$

We report both the *accuracy* and the *confusion matrix* related measures because by doing so we can easier interpret the performance of the model. Moreover, since the testing set is imbalanced, these measures can be considered very important.

5. Results and discussion

In this section we start by presenting the experimental results obtained by applying the approach introduced in Section 4 on a publicly available dataset of images representing signatures.

5.1. **Dataset and parameters setting.** The dataset used in our experiments is free and publicly available [7]. It consists of 4000 annotated samples from which 800 are forgeries. In order to construct the dataset, several persons were asked to write down their own signature. Furthermore, another person was asked to try to replicate the original signature.

In the dataset we have multiple signers each of them having the original signature and some forgeries. We intend to train our model in order to distinguish between the two signature types, forgery and original in an offline manner [6]. The available set can help the model to generalize since it consists of both forgeries and original samples coming from several persons.

The feature extraction step (Section 4.1) is first applied. The set of *bottleneck* features extracted from our images consists of 2048 positive real numbers, lower than 1. The pre-processed dataset will be then used for training the SVM.

The following sequences are used for optimizing the hyperparameters C and γ: $C \in \{1, 5, 10, 100, 1000\}$ and $\gamma \in \{1e-1, 1e-2, 1e-3, 1e-4, 1e-5\}$. The following kernels are candidates for the grid search: linear, polynomial, sigmoid and RBF.

The best hyperparameters which were chosen by analysing the results from the *grid search* are:

- kernel: *RBF*
- C=5
- γ=1e-2

5.2. **Results.** For our experiments we have used the *scikit-learn* implementation of *SVM* [12]. 80% of the dataset was reserved for training and on these instances we performed a training methodology which mainly consisted of a *SVM* grid search over the training set. The testing methodology described in Section 4.3 was applied on the trained *SVM* using the rest of the dataset. The obtained *accuracy* (*Acc*) was **95.1%**. For the obtained accuracy we compute the 95% Confidence Interval (CI) [1] as given in Formula (4).

$$CI(Acc) = 1.96 \cdot \sqrt{\frac{Acc \cdot (1 - Acc)}{n}} \tag{4}$$

where n represents the number of samples in the testing set. Accordingly, the 95% confidence interval is computed as follow: $[Acc - CI(Acc), Acc + CI(Acc)]$. For our experiment, the reported 95% CI for the *accuracy* on the testing set is **[0.935, 0.966]**. Thus, there is a 95% confidence that the accuracy of our classifier ranges in the confidence interval.

The confusion matrix from Table 1 provides a better overview on the performance of the proposed model.

		Forgery	Original	Total
Signatures	Forgery	132	23	155
	Original	12	548	560
	Total	144	571	715

TABLE 1. Confusion matrix

The AUC measure computed for our classifier is **0.92** and the *F-measure* is **0.88**. These values express a very good performance for the proposed classification model.

The dataset was reshuffled in order to repeat the random split for the training and testing sets. The proposed experiment was repeated 20 times in order to analyze the evolution of the AUC measure. We observe in Table 2 a low value for the standard deviation, as well as close AUC values for the minimum and maximum AUC reported during the 20 runs. Some of the ROC curve outcomes can be visualised in Figure 1. The random classifier (having an AUC of 0.5) is represented in Figure 1 by the dotted red line.

	min	max	median	mode	mean	stdev
AUC	0.88	0.94	0.90	0.91	0.90	0.0148

TABLE 2. Experimental results for 20 experiments

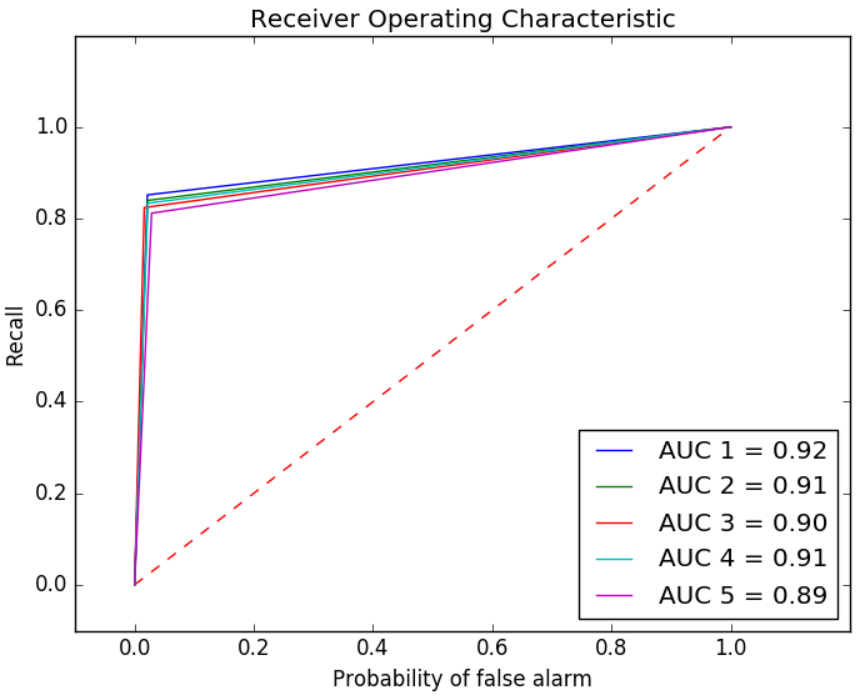

FIGURE 1. ROC curve outcomes for multiple experiments

We illustrate in Table 3 a brief comparison between our approach introduced in Section 4 and the similar related work described in Section 3.2. An exact comparison can be made only with the approaches from Kovari and Charaf [6][5], since they use for evaluation the same dataset as our case study. The other two approaches from [14] and [13] report results on other datasets, thus the comparison is not entirely relevant.

#	**Approach**	**Performance**	**Our approach**
1	Statistical analysis [6][5]	89%	$95\% \pm 0.015$
2	Statistical learning[14]	84%	–
3	Deep learning[13]	$85.03\% \pm 14.25$	–

TABLE 3. Comparison to related work based on the accuracy evaluation measure.

If we look only to the performance measure of the approaches described in Table 3, we observe that our approach is comparable to the related work.

Moreover, the 95% CI obtained by our approach is very small, compared to the one from [13], and this proves again the performance of our model.

6. Conclusions and further work

In this paper we have presented a *machine learning* method based on a feature extractor that can be successfully used in solving the signature verification problem. Considering the good results, we may say that we have confirmed again that this complex task is suitable for *machine learning* solving.

Further work consists in extending the experiment on multiple benchmark datasets in order to have a better overview of the capability of the proposed method. Building a convolutional neural network from scratch will be also considered.

References

[1] L.D. Brown, T.T. Cai, and A. DasGupta. Interval Estimation for a Binomial Proportion (with discussion). *Statistical Science*, 16(2):101–133, 2001.

[2] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[3] Tom Fawcett. An introduction to ROC analysis. *Pattern Recogn. Lett.*, 27(8):861–874, 2006.

[4] Mohsen Fayyaz, Mohammad Hajizadeh Saffar, Mohammad Sabokrou, and Mahmood Fathy. Feature representation for online signature verification. *CoRR*, abs/1505.08153, 2015.

[5] Bence Kovari and Hassan Charaf. Analysis of intra-person variability of features for off-line signature verification. *W. Trans. on Comp.*, 9(11):1359–1368, November 2010.

[6] Bence Kovari and Hassan Charaf. Statistical analysis of signature features with respect to applicability in off-line signature verification. In *Proceedings of the 14th WSEAS International Conference on Computers: Part of the 14th WSEAS CSCC Multiconference - Volume II*, ICCOMP'10, pages 473–478, Stevens Point, Wisconsin, USA, 2010. World Scientific and Engineering Academy and Society (WSEAS).

[7] Bence Kovari and Hassan Charaf. A study on the consistency and significance of local features in off-line signature verification. *Pattern Recognition Letters*, https://www.aut.bme.hu/Pages/Research/Signature/Resources, 2013.

[8] Bence Kovari, Benedek Toth, and Hassan Charaf. Classification approaches in off-line handwritten signature verification. *WSEAS TRANSACTIONS on MATHEMATICS Issue 9*, 2009.

[9] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.

[10] Thomas M. Mitchell. *Machine learning*. McGraw-Hill, Inc. New York, USA, 1997.

[11] Michael A. Nielsen. *Neural Networks and Deep Learning*. Determination Press, USA, January 2016.

[12] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12:2825–2830, November 2011.

[13] Bernardete Ribeiro, Ivo Gonçalves, Sérgio Santos, and Alexander Kovacec. *Deep Learning Networks for Off-Line Handwritten Signature Recognition*, pages 523–532. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.
[14] Harish Srinivasan, Sargur N. Srihari, and Matthew J. Beal. *Machine Learning for Signature Verification*, pages 761–775. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.
[15] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

Department of Computer Science, Faculty of Mathematics and Computer Science, Babeş-Bolyai University, Cluj-Napoca, Romania
E-mail address: tmic1334@scs.ubbcluj.ro