Expert Systems With Applications 146 (2020) 113160

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A-Stacking and A-Bagging: Adaptive versions of ensemble learning algorithms for spoof fingerprint detection

Shivang Agarwal, C. Ravindranath Chowdary*

Department of Computer Science and Engineering, Indian Institute of Technology (BHU), Varanasi 221005, India

ARTICLE INFO

Article history:
Received 16 May 2019
Revised 18 December 2019
Accepted 21 December 2019
Available online 24 December 2019

Keywords:
Stacking
Bagging
Ensemble learning
Spoof fingerprint detection

ABSTRACT

Stacking and bagging are widely used ensemble learning approaches that make use of multiple classifier systems. Stacking focuses on building an ensemble of heterogeneous classifiers while bagging constructs an ensemble of homogenous classifiers. There exist some applications where it is essential for learning algorithms to be adaptive towards the training data. We propose A-Stacking and A-Bagging; adaptive versions of stacking and bagging respectively that take into consideration the similarity inherently present in the dataset. One of the main motives of ensemble learning is to generate an ensemble of multiple "experts" that are weakly correlated. We achieve this by producing a set of disjoint experts where each expert is trained on a different subset of the dataset. We show the working mechanism of the proposed algorithms on spoof fingerprint detection. The proposed versions of these algorithms are adaptive as they conform to the features extracted from the live and spoof fingerprint images. From our experimental results, we establish that A-Stacking and A-Bagging give competitive results on both balanced and imbalanced datasets.

© 2019 Elsevier Ltd. All rights reserved.

1. Introduction

Ensemble learning is useful in overcoming the problems of single classifier systems, i.e. computational problems: when the learning process of a weak classifier is imperfect, statistical problems: when learning data is too small to capture the entire hypotheses space and representational problems: when the true target function cannot be found by any of the hypothesis from the hypotheses space (Dietterich, 1997). One of the active areas of research in supervised learning has been to study methods for constructing good ensembles of classifiers (Dietterich, 2000a).

It has been observed that the performance of ensemble learning depends heavily on the diversity among the individual classifiers of an ensemble. Polikar (2006) defines four ways to increase the diversity among the base classifiers: 1) by using different training data to train the base classifiers, 2) by using diverse training parameters, 3) by using different features for training the base classifiers, and 4) by combining different types of classifiers.

Multiple classifier systems (MCS) sometimes referred to as a committee of classifiers or a mixture of experts have been exploited by various algorithms (Polikar, 2006). Bagging, boosting, stacking and random forest are the popular methods based on MCS paradigm. Multiple variants of these ensemble methods have been proposed and used in the past, such as Ubagging (Liang & Cohn, 2013), AdaBoost (Freund & Schapire, 1997; Sun, yue Jia, & Li, 2011), AveBoost (Oza, 2003), conservative boosting (Kuncheva & Whitaker, 2002), GA-stacking (Ledezma, Aler, Sanchis, & Borrajo, 2010), cooperative ensemble learning system (CELS) (Yong Liu & Xin Yao, 1998), etc.

Stacking (Wolpert, 1992) and bagging (Breiman, 1996) are two popular ensemble learning approaches applied in various real-world scenarios such as intrusion detection, spam classification, credit scoring etc. (du Jardin, 2018; Papouskova & Hajek, 2019; Porwik, Doroz, & Wrobel, 2019; Ruano-Ords, Yevseyeva, Fernandes, Mndez, & Emmerich, 2019; Syarif, Zaluska, Prugel-Bennett, & Wills, 2012; Zhang & Mahadevan, 2019).

Stacking uses a meta-classifier to fuse the ensemble outputs, whereas voting, weighted majority voting etc. are the common ways to combine ensemble outputs in bagging. Also, the diversity in stacking is achieved by using heterogeneous classifiers on the same training set, whereas in bagging we try to gain diversity by using the same base classifier on different training sets (Bian & Wang, 2007). However, as these different training sets are bootstrapped from a single dataset, they are not entirely disjoint with each other, which results in low diversity (Banfield, Hall, Bowyer, & Kegelmeyer, 2005).

Several modified versions of popular ensemble learning approaches have been proposed in the past (Cheplygina, Tax, & Loog,

* Corresponding author.
E-mail addresses: shivanga.rs.cse16@iitbhu.ac.in (S. Agarwal), rchowdary.cse@iitbhu.ac.in (C.R. Chowdary).

https://doi.org/10.1016/j.eswa.2019.113160
0957-4174/© 2019 Elsevier Ltd. All rights reserved.



(a) Ecoflex (b) Gelatin (c) Latex (d) Woodglue (e) Siligum (f) Playdoh

Fig. 1. Examples of various types of spoof fingerprints.

2016; Ditzler, LaBarck, Ritchie, Rosen, & Polikar, 2018; Ting & Witten, 1997), but to the best of our knowledge the adaptiveness of the algorithm towards the dataset has not been explored yet.

Ensemble learning-based approaches have been used in the past for spoof fingerprint detection where the decisions of multiple base classifiers are integrated to classify an image as "live" or "spoof" (Ding & Ross, 2016; Kho, Lee, Choi, & Kim, 2019). Although ensemble learning is well-known for this particular application, to the best of our knowledge, stacking has not been used for spoof fingerprint detection. We claim that for such applications, instead of straightforward usage of base classifiers, it is crucial to adapt to the features of the dataset and to adjust the learning model accordingly.

Merz (1999) argues that having a disjoint set of classifiers is advantageous in the ensemble learning as it yields weakly correlated predictions. This motivated us to maintain the diversity of the ensemble by dividing the original training set into multiple subsets using clustering. In that way, we are able to generate a diverse set of classifiers by considering the features extracted from live and spoof fingerprint images of the dataset.

The models for fingerprint recognition are vulnerable to attacks by spoof fingerprints made of different moulds of substances like silicon, wood glue, latex, gelatin, etc. Therefore, it is required to perform liveness detection before fingerprint recognition to ensure that fabricated moulds are not used for authentication. Examples of spoof fingerprints generated using these substances are shown in Fig. 1.

Local Binary Patterns (LBP) is an efficient way to determine the texture of an image by labelling each pixel with a binary value based on the thresholds on the neighbouring pixels (Jia et al., 2014; Nanni & Lumini, 2008). LBP considers the central pixel as the threshold and based on that it assigns the binary values to the neighbouring pixels. LBP value of the pixel is calculated by summing up the element-wise product of the binary values with their weights. LBP histograms are robust in terms of grayscale variations, making them suitable for spoof fingerprint detection, as they can easily incorporate fingerprints with skin distortions, different skin qualities, dry, moist or dirty skin.

1.1. Contributions

- We explore the behaviours of stacking and bagging with various base classifiers on spoof fingerprint detection problem.
- We emphasize that the learning algorithms must be adaptive towards the properties inherent in the dataset.
- We establish that the diversity among the ensemble of classifiers can be achieved by performing clustering on the original training set and forming subsets of it.
- We propose adaptive models of stacking and bagging for spoof fingerprint detection and show their competitiveness on class balanced and imbalanced datasets.

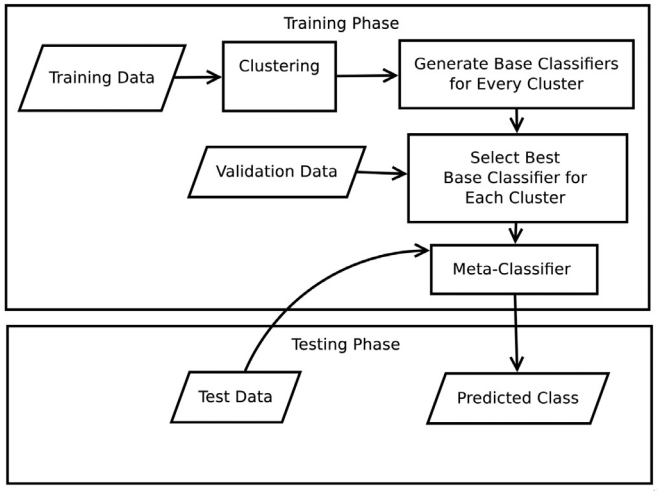


Fig. 2. Conceptual model of A-Stacking.

2. Stacking

Stacking (Wolpert, 1992) is a learning approach based on ensemble learning which combines the predictions made by multiple base classifiers generated by using different learning algorithms $L_1, L_2, \ldots L_n$. These classifiers are trained on the same training data D_{Train} containing examples in the form $s_i =< x_i, y_i >$, where x_i is the input vector, and y_i is the class label associated with it.

In the first phase, base classifiers $l_1, l_2, \ldots l_n$ make predictions for the query instance x_q. In the second phase, the meta-classifier M combines the predictions made by base classifiers and predicts the final class label. A critical issue in stacking is the choice of meta-classifier. Comparative studies have been presented in the past (Džeroski & Ženko, 2004) to analyse the performance of different learning algorithms as the meta-classifier. Logistic regression (le Cessie & van Houwelingen, 1992) is the current most popular learning scheme used in stacking as a meta-classifier.

The effort of using a meta-classifier is justified only when the performance of the ensemble of classifiers is better than the best individual base classifier (Džeroski & Ženko, 2004). Therefore, it is advised to consider the performance of the best individual base classifier to be the baseline for the performance of the ensemble.

2.1. A-Stacking

Algorithm 1 explains the methodology of A-Stacking and the conceptual model is illustrated in Fig. 2. The first step is to perform clustering on the original training data D_{Train} to form n clusters $(c_1, c_2, \ldots, c_n)$ of instances. Instances belonging to the same cluster are similar to each other and dissimilar to the instances belonging to other clusters. By performing clustering, we are able to generate disjoint bags of instances. These bags of instances are used as individual training datasets. For each cluster c_i, we generate base clas-

sifiers $l_i^1, l_i^2, ., l_i^n$ by applying different classifiers $L_1, L_2, ., L_n$. These base classifiers are tested on the validation data D_{valid} to select the best base classifier for each cluster. The individual accuracies of base classifiers are calculated using 10-fold cross-validation, and the best performing individual base classifier from each cluster is chosen and sent to the meta-classifier M. In the second phase, we combine the predictions made by the qualified base classifiers by using the meta-classifier. Therefore, for a query instance x_q from the test data D_{Test}, first we consider predictions from the base classifiers, and later we combine these predictions using the meta-classifiers to get the final class label.

Algorithm 1: A-Stacking Algorithm.

Data:
training data D_{Train},
test data D_{Test},
validation data D_{Valid},
a clustering algorithm C,
a set of classification algorithms $< L_1, L_2, ., L_n >$,
a meta-classifier M,
number of base classifiers n.
Result: ensemble classifier Z

1 $\{c_1, c_2, ., c_n\} \leftarrow C(D_{Train})$;
2 **for** *i= 1 to n* **do**
3 $< l_i^1, l_i^2, ., l_i^n > \leftarrow < L_1, L_2, ., L_n > (c_i)$;
4 Check the accuracies $< a_i^1, a_i^2, ., a_i^n >$ of $< l_i^1, l_i^2, ., l_i^n >$ on D_{Valid};
5 Select the best performing base classifier l_i from $< l_i^1, l_i^2, ., l_i^n >$ and send it to M;
6 Integrate the qualified base classifiers $< l_1, l_2, ., l_n >$ using the meta-classifier M;
7 **return** the ensemble classifier Z integrated by M to classify the instances in D_{Test};

3. Bagging

Bagging (Breiman, 1996) is a method of generating multiple versions of a base classifier by making bootstrapped replicates of training data and using them to get an aggregated predictor. The performance of Bagging improves if used with an unstable learner, i.e. if the learner causes significant changes by perturbing the training set.

Let the size of the original training set D_{Train} is N. Our task is to generate n bags of size N each by sampling D_{Train} with replacement. These n bags of instances may have duplicate instances, and the union of all these bags is a subset of D_{Train}. Each bag is used by the learning algorithm L to generate a base classifier l_i. These n base classifiers $l_1, l_2, \ldots, l_n$ are combined using majority voting to get the final predicted class label.

3.1. A-Bagging

Algorithm 2 explains the working mechanism of A-Bagging and the conceptual model is illustrated in Fig. 3. We use the same idea as A-Stacking in A-Bagging. The first step is to use the original training data D_{Train} for clustering to form n clusters. These clusters act as n bags of instances to be used for training and testing of individual base classifiers. In that way, we are able to generate disjoint bags of instances. The union of these bags of instances is the original training data D_{Train}. Therefore A-Bagging is different from traditional Bagging, which uses bootstrapped samples of training data to train the base classifiers.

Algorithm 2: A-Bagging Algorithm.

Data:
training data D_{Train},
test data D_{Test},
validation data D_{Valid},
a clustering algorithm C,
a classification algorithm L,
number of base classifiers n.
Result: ensemble classifier Z

1 $\{c_1, c_2, ., c_n\} \leftarrow C(D_{Train})$;
2 **for** *i= 1 to n* **do**
3 $l_i \leftarrow L(c_i)$;
4 Check the accuracy a_i of l_i on D_{Valid};
5 **return** the ensemble Z of $< l_1, l_2, ., l_n >$ to classify the instances in D_{Test};
6 Classify instances in D_{Test} using the weight co-efficients generated by Equation 2;

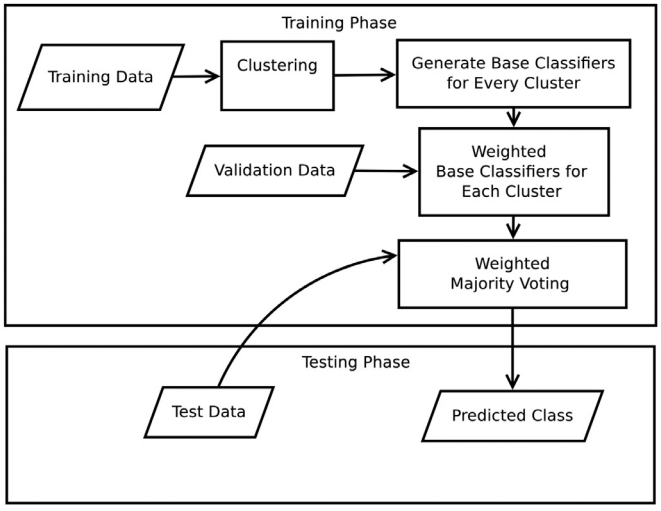


Fig. 3. Conceptual model of A-Bagging.

Later, these predictions are combined using weighted majority voting. The weights are assigned to the individual classifiers according to their performance on the validation data. Higher weight is assigned to the classifier having better performance. For classifying a query instance x_q, we associate the weight of the base classifier with its predicted class label. For the query instance x_q, the class label with the highest coefficient (i.e. the sum of weights of the base classifiers) is assigned. We use Eqs. (1) and (2) to assign weights to the classifiers.

$$w_y^x = \sum_{i=1}^{n} a_{iy}^x \tag{1}$$

where, w_y^x is the total weight associated with the class label y for an instance x. n is the number of base classifiers and a_{iy}^x is the accuracy of i^{th} base classifier which has predicted the class label y for instance x on validation data. The final class label y_f determined by the weighted majority is given by Eq. (2):

$$y_f^x = argmax_y(w_y^x) \tag{2}$$

4. Spoof fingerprint detection

The application we consider in this paper is spoof fingerprint detection which has its importance in forensics and information security (Ding & Ross, 2016; Kho et al., 2019; Nogueira, de Alencar

Lotufo, & Campos Machado, 2016; Rattani, Scheirer, & Ross, 2015). The machine learning methods for spoof/liveness detection are usually grouped into two categories: dynamic features based methods and static features based methods (Marasco & Ross, 2014). Dynamic features are identified as skin elasticity or perspiration pattern during the scanning process. Dynamic features are not usually preferred for liveness detection as they change over time, and the detection time is increased. Static features such as textural details can be observed from a single impression of a fingerprint to detect the liveness. Static features do not change during the scanning process, but the choice of features may drastically change the performance of the spoof detector. There are numerous approaches based on deep neural networks proposed in the past for fingerprint spoof detection. Menotti, et al., 2016; Nogueira et al., 2016; Chugh et al., 2018 use convolutional networks, (Ding & Ross, 2016; Kho et al., 2019; Rattani et al., 2015) use SVMs trained over local descriptors, (Gragnaniello, Poggi, Sansone, & Verdoliva, 2015) use bidimensional contrast-phase histogram as feature vector associated with fingerprint images.

There are instances where static features extracted using local descriptors such as LBP, LPQ etc. outperform deep learning-based feature extractors (Jia et al., 2014; Nogueira et al., 2016) and achieve good results on fingerprint liveness detection.

We use LivDet 2011 spoof detection dataset (Yambay et al., 2012) which captures this application comprehensively. We claim that while learning, the similarity inherently present in the dataset must be considered to form clusters of instances. Therefore, we propose new models and use them for detecting spoof fingerprints.

The proposed models use the concept of multiple classifier systems by generating a set of base classifiers, pruning them based on their performance on validation data and integrating them using logistic regression or weighted majority voting. The idea of using multiple classifier systems is justified to reduce the false predictions made by a single classifier system that is untrained on the class of a given test instance.

The first part of the proposed models is to generate the ensemble of base classifiers. The ensemble is created by using the following components: (i) Training data $D_{Train} = (x_1, y_1), \ldots . (x_n, y_n)$ contains n training examples belonging to both live and spoof classes, where x_i is a set of attributes generated from an image feature extraction algorithm (in our case, LBP) and y_i is the corresponding class label. (ii) A clustering algorithm C is used to cluster the training examples based on the similarity inherently present in the data. The target is to create a group of clusters $c_1, \ldots \ldots, c_k$ where the examples belonging to one cluster possess similar values of attributes whereas the examples belonging to different clusters possess different values of the attributes defined by LBP feature. (iii) A classification algorithm L (in A-Stacking $L = \{L_1, L2, ., L_n\}$) to train the base classifiers on each c_i. L uses each c_i to generate a base classifier l_i, which is used to make a decision individually. In this way, the decision boundary of each base classifier is different from others, resulting in an ensemble of diverse base classifiers. Later these decisions are integrated by logistic regression or weighted majority voting to decide for the whole ensemble. In this section, we present our experimental results and give a performance analysis of A-Stacking and A-Bagging.

5. Results and discussion

5.1. Experimental setup

We use python Weka wrapper to use clustering and classification functionalities of Weka (Hall et al., 2009). All the original datasets have been randomized and divided into 80:20 ratio for training D_{Train} and validation D_{Valid}, so that the validation set remains disjoint from the training set. We use Simple-kMeans (Arthur & Vassilvitskii, 2007) as our clustering algorithm which performs reasonably well on the chosen datasets with $k = 3$. We encourage the readers to experiment with various values of k and choose the optimal value empirically.

In this study, we compare A-Stacking and A-Bagging with the traditional stacking and bagging respectively with three base classifiers: SMO (Implements John Platt's sequential minimal optimization algorithm for training a support vector classifier) (Platt, 1998), Random Forest (RF) (Breiman, 2001) and Voted Perceptron (VP) (Freund & Schapire, 1998). These classifiers have been applied and proven to be efficient in various applications related to image classification. SVMs were used in fingerprint spoof detection in the recent literature (Kho et al., 2019; Rattani et al., 2015), whereas random forest and voted perceptron have also been a preferred choice of classifiers in spoof fingerprint and other image classification applications (Khardon & Wachman, 2007; Nayak, Dash, & Majhi, 2016). We report the accuracy of the classifiers as well as their false positive rate and emphasize that both of these measures must be used simultaneously to evaluate the performance of the classifier. Accuracy (Acc) signifies the overall performance of the classifier on both classes; however, the false positive rate (FPR) is crucial in such applications because of the high misclassification cost. A spoof fingerprint being classified as the live fingerprint may have a considerable impact, therefore for a classifier to be used in a real-world domain, it is necessary to maintain high accuracy and low false positive rate.

For stacking, we use the above-mentioned three base classifiers along with logistic regression (le Cessie & van Houwelingen, 1992) as the meta-classifier. The meta-classifier is responsible for integrating the individual decisions of the base classifiers. As the meta-classifier brings overhead to the system, the effort is only justified when the overall accuracy of the ensemble is greater than the best performing base classifier SelectBest.

A variety of schemes have been proposed for combining the ensemble outputs (Verikas, Lipnickas, Malmqvist, Bacauskiene, & Gelzinis, 1999), such as majority vote (Dietterich, 2000b), weighted majority vote (Dietterich, 2000b), average (Dietterich, 2000b), weighted average (Dietterich, 2000b), the Bayes approach (Sinha, Chen, Danu, Kirubarajan, & Farooq, 2008), the probabilistic schemes (Lee, Hong, & Kim, 2009), combination by a neural network (Sinha et al., 2008), etc. Majority voting has been a popular way of combining the ensemble outputs in bagging, whereas stacking requires the presence of a meta-classifier. In this study, for a fair comparison, we consider majority voting for both A-Bagging and bagging and logistic regression as a meta-classifier for both A-Stacking and stacking.

5.2. Datasets

The description of the datasets used in this paper is given in Table 1. We use livdet2011 database (Yambay et al., 2012), which was used in fingerprint liveness detection competition 2011. The goal of this competition was to compare software-based fingerprint liveness detection methodologies and fingerprint systems which incorporate liveness detection capabilities. The dataset consists of fingerprint images broadly classified in two classes: "live" and "spoof". These fingerprints are tested on four biometric sensors: Biometrika, DigitalPersona, ItalData and Sagem. For all the sensors, we have approximately 1000 fingerprint images each for both "live" and "spoof" classes in training and the same number of images in testing.

Further, the images belonging to the spoof class can be categorized in multiple subcategories based on the fabrication material used for creating the spoof or fake fingerprint. These materials are gelatin, latex, playdoh, wood glue, silicone etc. The dataset has 200 images, each belonging to five of these subcategories. To show the

Table 1
Description of datasets.

Database		Live (Train/Test)	Spoof (Train/Test)
LivDet2011	Biometrika	1000/1000	1000/1000 (ecoflex, gelatin, latex, silgum, wood glue)
	DigitalPersona	1004/1000	1000/1000 (gelatin, latex, playdoh, silicone, wood glue)
	ItalData	1000/1000	1000/1000 (ecoflex, gelatin, latex, silgum, wood glue)
	Sagem	1008/1000	1008/1036 (gelatin, latex, playdoh, silicone, wood glue)

Table 2
Performance evaluation of A-Stacking on class-balanced datasets.

Dataset	Stacking (SMO+RF+VP)		SelectBest		A-Stacking (SMO+RF+VP)	
	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)
Biometrika	82.35	0.23	78.85(RF)	0.25	**82.55**	0.23
DigitalPersona	82.85	0.23	85.30(SMO)	0.18	**85.35**	**0.18**
ItalData	66.75	0.16	70.00(SMO)	0.15	**70.05**	**0.15**
Sagem	86.24	0.10	84.23(SMO)	0.14	85.11	0.10
Average	79.54	0.17	79.59	0.18	**80.76**	**0.16**
± **Std Error**	4.35	0.03	3.49	0.02	**3.62**	0.03

effectiveness and robustness of the proposed algorithms, we conduct the experiments in two ways. One, while learning the whole training data in a batch and the other with 1000 live images and 400 images of two subcategories of the spoof class. In this way, we show the competitiveness of the models on class-balanced and imbalanced datasets. Note that for clustering the value of k was kept same for both class-balanced and imbalanced datasets.

5.3. Pre-processing

The image features need to be transformed into numeric attributes to perform classification on the fingerprint images. We do that by using LBP feature extraction on Weka. We use Binary Patterns Pyramid LBP from the imageFilter package of Weka[1]. It is a batch filter for extracting a pyramid of rotation-invariant local binary pattern histograms from images. Each local binary pattern represents an intensity pattern (e.g. an edge or a corner) around a point. A histogram of local binary patterns, therefore, encodes the larger-scale patterns that occur across regions of images. Local binary patterns are useful for texture and face recognition.

The fingerprint image features are transformed into 756 numeric attributes and one nominal class attribute making a high dimensional dataset.

5.4. Results

The results of the comparison between A-Stacking and stacking along with SelectBest on the class-balanced datasets measured by accuracy and false positive rate are described in Table 2.

It can be seen from Table 2 that the performance of stacking on DigitalPersona is lower than both A-Stacking and SelectBest. Therefore, the effort of using the meta-classifier in stacking is not justified. The performance of the proposed model is slightly lesser than its counterpart, but yet the performance is better than SelectBest, which supports the usage of meta-classifier. On average, A-Stacking performs well and yields competitive results to its counterparts.

The performance of A-Bagging is shown in Table 3. A-bagging performs well on class-balanced datasets, but the performance is marginally below the rivals in some cases.

The performance of A-Stacking on imbalanced class datasets of Biometrika sensor is given in Table 4. It is evident that A-Stacking performs better than its counterparts in most of the cases, and it is always superior to the best individual base classifier SelectBest, which makes it more effective than conventional stacking.

[1] https://github.com/mmayo888/ImageFilter.

The results of the comparison between A-Stacking and conventional stacking on class-imbalanced datasets of DigitalPersona sensor are given in Table 5. A-Stacking performs better than SelectBest in every case, however stacking is outperformed by SelectBest in three cases, which proves that the role of meta-classifier is fully justified in A-Stacking but not in stacking.

Table 6 shows the performance comparison on class-imbalanced datasets of ItalData sensor. It is evident that A-Stacking gives better performance than stacking in every case and its performance is always competitive with SelectBest, but it is never below the accuracy of SelectBest. It is worthwhile to note that SMO is the best individual base classifier in all the cases making it an excellent choice for spoof fingerprint detection.

The performance evaluation of A-Stacking on class-imbalanced Sagem datasets is given in Table 7. The performance is consistent and always better than SelectBest. The overall accuracy and false positive rate are better than its counterparts.

The performance of A-Bagging on class-imbalanced Biometrika datasets is evaluated in Table 8. We compare the accuracies and false positive rates of A-Bagging with traditional bagging on respective base classifiers. Here, the highlighted values show the better performance of A-bagging on its bagging counterpart. It is evident from Table 8 that A-Bagging performs better than bagging in most of the cases irrespective of the base classifier.

Table 9 shows the performance comparison between A-Bagging, and it's counterparts on class-imbalanced datasets of DigitalPersona sensor. Here, the performance is marginally lesser with SMO and VP as base classifiers in some cases, but the average performance is better with RF base classifier.

Table 10 shows the results on imbalanced-class datasets of ItalData sensor. The performance is competitive with traditional bagging in most of the cases. However, there is a room for improvement in the false positive rate with SMO and VP base classifiers.

The performance of different bagging predictors on class-imbalanced datasets of Sagem sensor is given in Table 11. It is evident that A-Bagging produces competitive results both in terms of accuracy and false positive rate.

5.5. Discussion on results

As we are proposing the adaptive versions of the ensemble learning algorithms, it is vital to consider an application where this adaptiveness is essential. We take spoof fingerprint detection problem to show the working mechanism of the proposed models. Livdet 2011 is a high dimensional dataset which makes it easier to provide a thorough analysis of the performance of the proposed

Table 3
Performance evaluation of A-Bagging on class-balanced datasets.

Dataset	Bagging (SMO)		Bagging (RF)		Bagging (VP)		ABagging (SMO)		ABagging (RF)		ABagging (VP)	
	Acc (%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)
Biometrika	79.10	0.29	80.15	0.23	78.05	0.33	78.60	0.31	78.85	0.25	77.15	0.29
DigitalPersona	85.25	0.17	81.75	0.26	78.10	0.34	**85.30**	0.18	79.75	0.27	77.70	0.34
ItalData	68.55	0.17	57.15	0.12	56.15	0.12	**70.00**	**0.15**	**58.80**	0.13	52.45	0.13
Sagem	85.36	0.12	83.05	0.16	77.70	0.16	84.23	0.14	**83.10**	0.16	**78.97**	0.20
Average	79.56	0.19	75.52	0.19	72.50	0.24	79.53	0.19	75.12	0.20	71.57	0.24
± Std Error	3.95	0.04	6.15	0.03	5.45	0.06	**3.50**	0.04	**5.52**	0.03	6.38	**0.05**

Table 4
Performance evaluation of A-Stacking on class imbalanced Biometrika datasets.

Dataset	Stacking(SMO+RF+VP)		SelectBest		A-Stacking(SMO+RF+VP)	
	Accuracy(%)	FPR(0–1)	Accuracy(%)	FPR(0–1)	Accuracy(%)	FPR(0–1)
Live+Ecoflex+Gelatin	71.05	0.53	68.75(SMO)	0.57	70.30	0.55
Live+Ecoflex+Latex	73.90	0.49	71.80(SMO)	0.54	**74.30**	**0.47**
Live+Ecoflex+Silgum	74.80	0.44	71.45(SMO)	0.51	**74.95**	0.44
Live+Ecoflex+WoodGlue	67.00	0.62	67.15(SMO)	0.59	**67.15**	**0.59**
Live+Gelatin+Latex	79.25	0.35	76.25(SMO)	0.41	**79.50**	0.35
Live+Gelatin+Silgum	78.85	0.33	75.80(SMO)	0.39	**80.20**	**0.30**
Live+Gelatin+WoodGlue	73.60	0.46	72.25(SMO)	0.48	**73.70**	0.46
Live+Latex+Silgum	77.05	0.39	73.60(SMO)	0.47	76.55	0.41
Live+Latex+WoodGlue	69.80	0.55	70.95(SMO)	0.50	**71.15**	**0.52**
Live+Silgum+WoodGlue	69.25	0.53	67.40(VP)	0.56	**69.35**	**0.50**
Average	73.45	0.47	71.54	0.50	**73.71**	**0.46**
± Std Error	1.32	0.03	1.00	0.02	1.36	0.03

Table 5
Performance evaluation of A-Stacking on class imbalanced DigitalPersona datasets.

Dataset	Stacking(SMO+RF+VP)		SelectBest		A-Stacking(SMO+RF+VP)	
	Accuracy(%)	FPR(0–1)	Accuracy(%)	FPR(0–1)	Accuracy(%)	FPR(0–1)
Live+Gelatin+Latex	71.70	0.52	69.65(SMO)	0.54	**72.15**	**0.49**
Live+Gelatin+Playdoh	69.10	0.61	68.05(SMO)	0.63	68.90	0.61
Live+Gelatin+Silicone	66.90	0.63	68.35(SMO)	0.61	**68.35**	**0.61**
Live+Gelatin+WoodGlue	69.00	0.56	67.65(SMO)	0.57	**69.90**	**0.54**
Live+Latex+Playdoh	72.90	0.50	71.20(SMO)	0.55	72.60	0.51
Live+Latex+Silicone	71.90	0.51	71.80(SMO)	0.52	**72.80**	**0.49**
Live+Latex+WoodGlue	65.95	0.63	70.00(SMO)	0.54	**70.05**	**0.54**
Live+Playdoh+Silicone	67.75	0.61	67.80(SMO)	0.61	**67.80**	0.61
Live+Playdoh+WoodGlue	75.95	0.44	75.40(SMO)	0.44	**76.25**	**0.43**
Live+Silicone+WoodGlue	74.30	0.46	73.60(SMO)	0.47	73.95	**0.46**
Average	70.54	0.55	70.35	0.55	**71.27**	**0.53**
± Std Error	1.05	0.02	0.83	0.02	0.86	0.02

models. In addition to that, it is important to show the models' behavior on class-imbalanced datasets. We accomplished this by taking the whole "live" class along with two subcategories of the "spoof" class.

As it is mentioned earlier, having an ensemble of weakly correlated base classifiers increases the diversity of the ensemble which results in better performance. We take this as our motivation and build classifiers on different subsets of training data. We perform clustering on the training data to form subsets of data. In this study, we considered Simple-kMeans as the clustering algorithm, but the models are independent of the choice of the clustering algorithm. In A-Stacking, we use multiple base classifiers on the subsets of data, and the performance on both class-balanced and imbalanced datasets as given in Tables 2, 4–7 is competitive to its counterparts. We compared the results with the traditional stacking approach as well as with the best individual base classifier of the ensemble. As our results are always better than SelectBest, we justify the effort of using a meta-classifier. It is evident from our experimental results that traditional stacking often gives lower accuracy than the SelectBest, which is against the argument.

In A-Bagging, we worked with the same motivation, and here also, the results are competitive. We used a weighted majority voting scheme to combine the predictions made by the base classifier on different subsets of data. We compared the performance of A-Bagging with conventional bagging with corresponding base classifiers on class balanced and imbalanced spoof detection datasets and showed that the results are equally well on both types of datasets.

We claim that the proposed adaptive algorithms are generic and can be applied in various applications. The motivation is to draw attention towards the properties intrinsic to the datasets, which makes the proposed algorithms different from the traditional ensemble learning algorithms. Therefore, A-Stacking and A-Bagging are not restricted to fingerprint spoof detection and can be exploited for applications where stacking and bagging have been used, i.e., network intrusion detection (Syarif et al., 2012), credit scoring (Brown & Mues, 2012; Wang, Hao, Ma, & Jiang, 2011; Xia, Liu, Da, & Xie, 2018), bio-informatics (Yang, Hwa Yang, B. Zhou, & Y. Zomaya, 2010), Nosiheptide fermentation product concentration prediction (peng Niu, li Wang, ling Zhang, kuo He, & xing

Table 6
Performance evaluation of A-Stacking on class imbalanced ItalData datasets.

Dataset	Stacking(SMO+RF+VP)		SelectBest		A-Stacking(SMO+RF+VP)	
	Accuracy(%)	FPR(0–1)	Accuracy(%)	FPR(0–1)	Accuracy(%)	FPR(0–1)
Live+Ecoflex+Gelatin	63.40	0.46	69.95(SMO)	0.31	**69.95**	**0.31**
Live+Ecoflex+Latex	67.35	0.41	68.10(SMO)	0.39	**68.35**	0.40
Live+Ecoflex+Silgum	60.00	0.44	61.90(SMO)	0.40	**61.90**	**0.40**
Live+Ecoflex+WoodGlue	69.80	0.28	70.25(SMO)	0.28	**70.65**	0.29
Live+Gelatin+Latex	68.35	0.28	69.95(SMO)	0.25	**69.95**	**0.25**
Live+Gelatin+Silgum	59.40	0.42	64.35(SMO)	0.27	**64.35**	**0.27**
Live+Gelatin+WoodGlue	67.95	0.31	70.20(SMO)	0.24	**70.20**	**0.24**
Live+Latex+Silgum	63.60	0.38	66.05(SMO)	0.32	**66.05**	**0.32**
Live+Latex+WoodGlue	70.80	0.33	71.45(SMO)	0.30	**71.45**	**0.30**
Live+Silgum+WoodGlue	63.85	0.38	69.60(SMO)	0.32	**69.60**	**0.32**
Average	65.45	0.37	68.18	0.31	**68.24**	0.31
± Std Error	1.26	0.02	0.98	0.02	0.99	0.02

Table 7
Performance evaluation of A-Stacking on class imbalanced Sagem datasets.

Dataset	Stacking(SMO+RF+VP)		SelectBest		A-Stacking(SMO+RF+VP)	
	Accuracy(%)	FPR(0–1)	Accuracy(%)	FPR(0–1)	Accuracy(%)	FPR(0–1)
Live+Gelatin+Latex	76.91	0.38	79.76(SMO)	0.30	**79.76**	**0.30**
Live+Gelatin+Playdoh	66.89	0.62	66.20(SMO)	0.61	66.20	**0.61**
Live+Gelatin+Silicone	69.15	0.56	68.02(SMO)	0.56	**70.87**	**0.54**
Live+Gelatin+WoodGlue	73.13	0.51	78.48(SMO)	0.35	**78.48**	**0.35**
Live+Latex+Playdoh	80.10	0.32	80.84(SMO)	0.29	**81.18**	0.29
Live+Latex+Silicone	76.62	0.40	75.63(SMO)	0.40	76.12	0.40
Live+Latex+WoodGlue	68.95	0.58	69.84(SMO)	0.56	**69.89**	**0.55**
Live+Playdoh+Silicone	66.55	0.59	67.63(SMO)	0.59	**67.68**	**0.59**
Live+Playdoh+WoodGlue	75.04	0.43	76.71(SMO)	0.38	**76.71**	**0.38**
Live+Silicone+WoodGlue	72.98	0.48	72.69(SMO)	0.48	72.69	0.48
Average	72.63	0.49	73.58	0.45	**73.96**	**0.45**
± Std Error	1.45	0.03	1.71	0.04	1.65	0.04

Table 8
Performance evaluation of A-Bagging on class imbalanced Biometrika datasets.

Dataset	Bagging (SMO)		Bagging (RF)		Bagging (VP)		ABagging (SMO)		ABagging (RF)		ABagging (VP)	
	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)
Live+Ecoflex+Gelatin	67.25	0.61	62.30	0.74	60.65	0.75	**68.75**	**0.57**	**64.65**	**0.69**	**64.50**	**0.67**
Live+Ecoflex+Latex	69.15	0.58	67.20	0.64	67.10	0.63	**71.80**	**0.54**	**68.80**	**0.60**	**68.20**	**0.61**
Live+Ecoflex+Silgum	67.20	0.60	71.00	0.53	69.70	0.57	**71.45**	**0.51**	**71.05**	**0.53**	**70.60**	**0.55**
Live+Ecoflex+WoodGlue	67.45	0.59	56.10	0.87	61.65	0.75	67.15	0.59	**57.60**	**0.84**	56.05	0.86
Live+Gelatin+Latex	75.15	0.44	73.45	0.49	73.75	0.48	**76.25**	**0.41**	**74.00**	**0.48**	**75.15**	**0.44**
Live+Gelatin+Silgum	74.95	0.41	74.00	0.45	72.25	0.50	**75.80**	**0.39**	**74.65**	**0.44**	**72.85**	**0.50**
Live+Gelatin+WoodGlue	70.55	0.50	60.90	0.76	66.50	0.63	**72.25**	**0.48**	60.70	**0.76**	59.10	0.80
Live+Latex+Silgum	71.55	0.51	72.05	0.50	69.85	0.55	**73.60**	**0.47**	**72.20**	**0.50**	**72.85**	**0.48**
Live+Latex+WoodGlue	71.15	0.50	64.70	0.67	65.05	0.64	70.95	**0.50**	63.95	0.68	**69.00**	**0.55**
Live+Silgum+WoodGlue	66.25	0.59	65.95	0.60	60.55	0.72	**67.05**	**0.58**	**66.70**	**0.59**	**67.40**	**0.56**
Average	70.06	0.53	66.76	0.62	66.70	0.62	**71.50**	**0.50**	**67.43**	**0.61**	**67.57**	**0.60**
± Std Error	1.00	0.02	1.87	0.04	1.49	0.03	1.01	0.02	1.81	0.04	1.94	0.04

Table 9
Performance evaluation of A-Bagging on class imbalanced DigitalPersona datasets.

Dataset	Bagging (SMO)		Bagging(RF)		Bagging (VP)		ABagging (SMO)		ABagging (RF)		ABagging (VP)	
	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)
Live+Gelatin+Latex	72.05	0.50	61.40	0.76	57.55	0.84	69.65	0.54	**61.55**	**0.75**	56.20	0.87
Live+Gelatin+Playdoh	68.15	0.63	65.75	0.68	66.25	0.66	68.05	0.63	65.45	0.68	**67.40**	**0.64**
Live+Gelatin+Silicone	68.95	0.59	62.40	0.73	65.45	0.68	68.35	0.61	62.35	**0.72**	**67.70**	**0.61**
Live+Gelatin+WoodGlue	69.10	0.55	53.05	0.93	60.90	0.76	67.65	0.57	**60.95**	**0.76**	52.45	0.94
Live+Latex+Playdoh	71.95	0.52	66.80	0.65	59.90	0.80	71.20	0.55	**66.95**	**0.64**	**60.75**	**0.78**
Live+Latex+Silicone	72.20	0.50	64.40	0.69	62.50	0.74	71.80	0.52	**65.35**	**0.68**	**62.85**	**0.72**
Live+Latex+WoodGlue	70.40	0.55	60.60	0.75	59.90	0.77	70.00	**0.54**	**60.90**	**0.74**	55.60	0.87
Live+Playdoh+Silicone	68.70	0.59	64.00	0.70	63.60	0.70	67.80	0.61	63.70	0.71	**63.80**	0.71
Live+PlaydohSilicone+WoodGlue	74.65	0.46	66.10	0.66	58.15	0.83	**75.40**	**0.44**	**67.15**	**0.63**	56.95	0.85
Live+Silicone+WoodGlue	73.50	0.45	64.60	0.69	58.40	0.82	**73.60**	0.47	**65.00**	**0.68**	**61.10**	**0.76**
Average	70.96	0.53	62.91	0.72	61.26	0.76	70.35	0.55	**63.93**	**0.70**	60.48	0.77
± Std Error	0.70	0.02	1.27	0.02	0.97	0.02	0.83	0.02	0.75	0.01	1.62	0.03

Table 10
Performance evaluation of A-Bagging on class imbalanced ItalData datasets.

Dataset	Bagging (SMO)		Bagging (RF)		Bagging (VP)		ABagging (SMO)		ABagging (RF)		ABagging (VP)	
	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)
Live+Ecoflex+Gelatin	67.30	0.31	56.75	0.75	58.40	0.40	**69.95**	**0.31**	**59.85**	**0.66**	57.90	**0.24**
Live+Ecoflex+Latex	66.90	0.42	59.35	0.73	63.15	0.35	**68.10**	**0.39**	**62.55**	**0.67**	59.50	**0.31**
Live+Ecoflex+Silgum	62.65	0.39	56.25	0.73	56.00	0.30	61.90	0.39	**58.20**	**0.67**	**57.40**	0.45
Live+Ecoflex+WoodGlue	71.00	0.25	59.85	0.67	58.75	0.26	70.25	0.28	**62.15**	**0.60**	**62.60**	**0.21**
Live+Gelatin+Latex	70.70	0.25	64.00	0.58	63.00	0.33	69.95	0.25	**66.40**	**0.53**	**66.35**	**0.22**
Live+Gelatin+Silgum	67.55	0.25	57.85	0.63	60.80	0.41	64.35	0.27	**59.30**	**0.58**	58.15	**0.32**
Live+Gelatin+WoodGlue	72.50	0.23	62.10	0.59	60.20	0.24	70.20	0.24	**62.90**	**0.56**	**61.95**	0.25
Live+Latex+Silgum	64.70	0.38	61.30	0.63	61.20	0.23	**66.05**	**0.32**	**61.75**	**0.57**	55.10	0.37
Live+Latex+WoodGlue	71.35	0.29	64.00	0.58	60.00	0.24	**71.45**	0.30	**66.35**	**0.52**	**69.10**	0.29
Live+Silgum+WoodGlue	66.40	0.32	54.05	0.69	56.75	0.21	**69.60**	**0.32**	**59.30**	**0.61**	**60.85**	0.34
Average	68.10	0.31	59.55	0.66	59.82	0.30	**68.18**	0.31	**61.87**	**0.60**	**60.89**	0.30
± Std Error	1.01	0.02	1.06	0.02	0.75	0.02	0.98	0.02	0.90	0.02	1.35	0.02

Table 11
Performance evaluation of A-Bagging on class imbalanced Sagem datasets.

Dataset	Bagging (SMO)		Bagging (RF)		Bagging (VP)		ABagging (SMO)		ABagging (RF)		ABagging (VP)	
	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)	Acc(%)	FPR(0–1)
Live+Gelatin+Latex	80.59	0.28	70.92	0.56	58.15	0.81	79.76	0.30	70.72	**0.56**	**61.73**	**0.72**
Live+Gelatin+Playdoh	68.56	0.57	62.72	0.73	65.91	0.66	66.20	0.61	61.59	0.75	65.07	0.67
Live+Gelatin+Silicone	69.49	0.54	66.99	0.64	67.19	0.59	68.02	0.56	66.01	0.65	**67.42**	0.61
Live+Gelatin+WoodGlue	77.55	0.35	66.25	0.67	63.26	0.69	**78.48**	0.35	65.61	0.69	62.42	0.72
Live+Latex+Playdoh	80.35	0.31	71.31	0.54	62.27	0.72	**80.84**	**0.29**	**73.72**	**0.48**	60.70	0.75
Live+Latex+Silicone	75.39	0.41	72.93	0.50	67.97	0.51	**75.63**	**0.41**	**73.37**	**0.49**	67.23	0.55
Live+Latex+WoodGlue	68.90	0.57	63.45	0.69	61.83	0.68	**69.84**	**0.56**	64.34	**0.68**	61.24	**0.66**
Live+Playdoh+Silicone	67.63	0.59	65.76	0.65	65.91	0.63	**67.63**	**0.59**	65.52	**0.65**	**66.11**	**0.62**
Live+Playdoh+WoodGlue	77.21	0.38	69.74	0.59	69.44	0.54	76.71	**0.38**	69.44	0.59	68.36	0.59
Live+Silicone+WoodGlue	72.29	0.48	70.57	0.56	65.76	0.57	**72.69**	**0.48**	**70.67**	0.56	**65.81**	**0.56**
Average	73.79	0.45	68.06	0.61	64.76	0.64	73.58	0.45	**68.10**	**0.61**	64.61	0.64
± Std Error	1.59	0.03	1.11	0.02	1.06	0.02	1.71	0.04	1.28	0.03	0.90	0.02

Jia, 2011), wireless sensor network target localization (Kim, Park, Yoo, Kim, & Park, 2013), predicting high performance concrete compressive strength (Chou & Pham, 2013), bankruptcy prediction (Barboza, Kimura, & Altman, 2017), etc.

6. Conclusions

In this study, we explore the behaviour of various ensemble learning approaches to spoof fingerprint detection. We propose A-Stacking and A-Bagging: the adaptive versions of ensemble learning approaches Stacking and Bagging, respectively. We hypothesize that the learning algorithms must take into consideration the similarity inherently present in the data. By doing so, the experts can be made adaptive towards the task associated with the dataset.

To maintain diversity among the ensemble, we perform clustering that creates expert spoof detectors trained on different subsets of training data. We used logistic regression as the meta-classifier in A-Stacking and showed that our results are always better than the best individual base classifier, which justifies the extra effort of employing a meta-classifier. We use A-Bagging by applying the same base learner to different subsets of data and combine their predictions using weighted majority voting. From our experimental results, we establish that the proposed adaptive models perform reasonably well on spoof detection in terms of accuracy and false positive rate. In future, the proposed algorithms may be used with applications where the adaptivity towards the data can be utilised to address the class imbalance.

Author contribution statements

Ravindranath Chowdary C defined the problem statement and Shivang Agarwal worked on the problem under the supervision of Ravindranath Chowdary C. This work was done as part of the PhD programme of Shivang Agarwal which started in 2017. This work is our original work and is currently not submitted anywhere else.

Declaration of Competing Interest

Tha authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Results for *k* values 4 and 5

To establish the robustness of the proposed models with respect to number of clusters, we give the performance of A-Stacking and A-Bagging in comparison with traditional stacking and bagging for k values 4 and 5. For stacking, SMO outperforms the other base classifiers in most of the cases, therefore to see a significant difference in the results, we exclude it from the ensemble and replace it with J48 and Naive bayes (NB) classifiers.

Table A.12 gives the performance analysis of A-Stacking with Naive Bayes, J48, Random Forest and Voted perceptron classifiers on class-balanced datasets.

Table A.13 presents the average results of A-Stacking with class-imbalanced datasets. Tables A.14 and A.15 represent the results of A-Bagging on class-balanced and imbalanced datasets respectively.

We also present a sample of results of A-Stacking when $k = 5$ and base classifiers are NB, J48, RF, Random Tree (RT) and VP in Table A.16.

Table A1
Performance evaluation of A-Stacking (NB+Rf+VP+J48) on class balanced datasets with $k = 4$.

Dataset	Stacking		SelectBest		AStacking	
	Accuracy	FPR	Accuracy	FPR	Accuracy	FPR
Biometrika	80	0.26	78.85(RF)	0.25	79.65	0.25
DigitalPersona	81.95	0.24	79.75(RF)	0.27	82.85	0.22
ItalData	64	0.23	66.25(VP)	0.14	62.4	0.27
Sagem	84.53	0.13	83.3(VP)	0.12	83.3	0.13
Average	77.62	0.21	77.04	0.19	77.05	0.22

Table A2
Average results of performance of A-Stacking (NB+Rf+VP+J48) on class imbalanced datasets with $k = 4$.

Dataset	Stacking		SelectBest		AStacking	
	Accuracy	FPR	Accuracy	FPR	Accuracy	FPR
Biometrika	72.76	0.45	71.96	0.47	68.8	0.47
DigitalPersona	70.14	0.53	69.17	0.56	66.37	0.59
ItalData	64.83	0.38	61.89	0.42	63.48	0.45
Sagem	71.47	0.48	71.48	0.49	69.34	0.51
Average	69.80	0.46	68.62	0.48	67.00	0.50

Table A3
Performance evaluation of A-Bagging on class balanced datasets with $k = 4$.

Dataset	Bagging (SMO)		Bagging (RF)		Bagging (VP)		ABagging (SMO)		ABagging (RF)		ABagging (VP)	
	Accuracy	FPR	Accuracy	FPR	Accuracy	FPR	Accuracy	FPR	Accuracy	FPR	Accuracy	FPR
Biometrika	78.2	0.34	80.2	0.23	77.75	0.35	78.6	0.31	78.85	0.25	77.15	0.3
DigitalPersona	84.2	0.23	81.55	0.26	77.3	0.37	85.3	0.18	79.75	0.27	77.7	0.34
ItalData	69.9	0.2	56.6	0.13	58.4	0.14	70	0.15	58.8	0.13	52.45	0.13
Sagem	85.95	0.14	82.85	0.15	78.19	0.17	84.23	0.14	83.1	0.15	78.98	0.2
Average	79.56	0.23	75.3	0.19	72.91	0.26	79.53	0.19	75.12	0.2	71.57	0.24

Table A4
Average results of the performance of A-Bagging on class imbalanced datasets with $k = 4$.

Dataset	Bagging (SMO)		Bagging (RF)		Bagging (VP)		ABagging (SMO)		ABagging (RF)		ABagging (VP)	
	Accuracy	FPR	Accuracy	FPR	Accuracy	FPR	Accuracy	FPR	Accuracy	FPR	Accuracy	FPR
Biometrika	68.49	0.58	66.76	0.63	65.59	0.65	71.5	0.50	67.43	0.61	67.57	0.60
DigitalPersona	69.72	0.57	63.73	0.71	60.13	0.78	70.35	0.55	63.93	0.70	60.48	0.77
ItalData	68.64	0.34	59.57	0.66	60.82	0.33	68.18	0.31	61.87	0.60	60.89	0.30
Sagem	73.05	0.46	67.98	0.59	64.26	0.63	73.58	0.45	68.10	0.61	64.61	0.64
Average	69.97	0.49	64.51	0.65	62.70	0.60	70.90	0.45	65.33	0.63	63.39	0.58

Table A5
Performance of A-Stacking on class-balanced datasets with $k = 5$.

Dataset	AStacking		Stacking		SelectBest	
	Accuracy	FPR	Accuracy	FPR	Accuracy	FPR
Biometrika	79.65	0.25	80.65	0.24	78.85(RF)	0.25
DigitalPersona	82.85	0.22	81.8	0.24	79.75(RF)	0.27
ItalData	62.4	0.27	57.55	0.23	58.8(RF)	0.13
Sagem	83.3	0.13	83.59	0.13	83.1(RF)	0.16
Average	77.05	0.22	75.9	0.21	75.12	0.2

References

Arthur, D., & Vassilvitskii, S. (2007). k-means++: the advantages of careful seeding. In N. Bansal, K. Pruhs, & C. Stein (Eds.), *Proceedings of the eighteenth annual ACM-SIAM symposium on discrete algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7–9, 2007* (pp. 1027–1035). SIAM.

Banfield, R. E., Hall, L. O., Bowyer, K. W., & Kegelmeyer, W. (2005). Ensemble diversity measures and their application to thinning. *Information Fusion, 6*(1), 49–62. doi:10.1016/j.inffus.2004.04.005.

Barboza, F., Kimura, H., & Altman, E. (2017). Machine learning models and bankruptcy prediction. *Expert Systems with Applications, 83*, 405–417. doi:10.1016/j.eswa.2017.04.006.

Bian, S., & Wang, W. (2007). On diversity and accuracy of homogeneous and heterogeneous ensembles. *International Journal of Hybrid Intelligence Systems, 4*(2), 103–128.

Breiman, L. (1996). Bagging predictors. *Machine Learning, 24*(2), 123–140. doi:10.1007/BF00058655 .

Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5–32. doi:10.1023/A:1010933404324.

Brown, I., & Mues, C. (2012). An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert Systems with Applications, 39*(3), 3446–3453. doi:10.1016/j.eswa.2011.09.033.

le Cessie, S., & van Houwelingen, J. (1992). Ridge estimators in logistic regression. *Applied Statistics, 41*(1), 191–201.

Cheplygina, V., Tax, D. M. J., & Loog, M. (2016). Dissimilarity-based ensembles for multiple instance learning. *IEEE Transactions on Neural Networks and Learning Systems, 27*(6), 1379–1391. doi:10.1109/TNNLS.2015.2424254.

Chou, J.-S., & Pham, A.-D. (2013). Enhanced artificial intelligence for ensemble approach to predicting high performance concrete compressive strength. *Construction and Building Materials, 49*, 554–563. doi:10.1016/j.conbuildmat.2013.08.078.

Chugh, T., Cao, K., & Jain, A. K. (2018). IEEE Transactions on Information Forensics and Security. *Fingerprint Spoof Buster: Use of Minutiae-Centered Patches, 13*(9), 2190–2202. doi:10.1109/TIFS.2018.2812193.

Dietterich, T. G. (1997). Machine-learning research–four current directions. *AI Magazine, 18*, 97–136.

Dietterich, T. G. (2000a). Ensemble methods in machine learning. In *Multiple classifier systems* (pp. 1–15). Berlin, Heidelberg: Springer.

Dietterich, T. G. (2000b). Ensemble methods in machine learning. *Multiple classifier systems, 1857*, 1–15.

Ding, Y., & Ross, A. (2016). An ensemble of one-class SVMS for fingerprint spoof detection across different fabrication materials. In *IEEE international workshop on information forensics and security, WIFS 2016, Abu Dhabi, United Arab Emirates, December 4–7, 2016* (pp. 1–6). IEEE. doi:10.1109/WIFS.2016.7823572.

Ditzler, G., LaBarck, J., Ritchie, J., Rosen, G., & Polikar, R. (2018). Extensions to online feature selection using bagging and boosting. *IEEE Transactions on Neural Networks and Learning Systems, 29*(9), 4504–4509. doi:10.1109/TNNLS.2017.2746107.

Džeroski, S., & Ženko, B. (2004). Is combining classifiers with stacking better than selecting the best one? *Machine Learning, 54*(3), 255–273. doi:10.1023/B:MACH.0000015881.36452.6e.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences, 55*(1), 119–139. doi:10.1006/jcss.1997.1504.

Freund, Y., & Schapire, R. E. (1998). Large margin classification using the perceptron algorithm. In *11th annual conference on computational learning theory* (pp. 209–217). New York, NY: ACM Press.

Gragnaniello, D., Poggi, G., Sansone, C., & Verdoliva, L. (2015). Local contrast phase descriptor for fingerprint liveness detection. *Pattern Recognition, 48*(4), 1050–1058. doi:10.1016/j.patcog.2014.05.021.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The weka data mining software: An update. *SIGKDD Explor. Newsl., 11*(1), 10–18. doi:10.1145/1656274.1656278.

du Jardin, P. (2018). Failure pattern-based ensembles applied to bankruptcy forecasting. *Decision Support Systems, 107*, 64–77. doi:10.1016/j.dss.2018.01.003.

Jia, X., Yang, X., Cao, K., Zang, Y., Zhang, N., Dai, R., … Tian, J. (2014). Multi-scale local binary pattern with filters for spoof fingerprint detection. *Information Sciences, 268*, 91–102. doi:10.1016/j.ins.2013.06.041.

Khardon, R., & Wachman, G. (2007). Noise tolerant variants of the perceptron algorithm. *Journalof Machine Learning Research, 8*, 227–248.

Kho, J. B., Lee, W., Choi, H., & Kim, J. (2019). An incremental learning method for spoof fingerprint detection. *Expert Systems with Applications, 116*, 52–64. doi:10.1016/j.eswa.2018.08.055.

Kim, W., Park, J., Yoo, J., Kim, H. J., & Park, C. G. (2013). Target localization using ensemble support vector regression in wireless sensor networks. *IEEE Transactions on Cybernetics, 43*(4), 1189–1198. doi:10.1109/TSMCB.2012.2226151.

Kuncheva, L. I., & Whitaker, C. J. (2002). Using diversity with three variants of boosting: Aggressive, conservative, and inverse. In *Multiple classifier systems* (pp. 81–90). Berlin, Heidelberg: Springer Berlin Heidelberg.

Ledezma, A., Aler, R., Sanchis, A., & Borrajo, D. (2010). Ga-stacking: Evolutionary stacked generalization. *Intelligent Data Analysis, 14*(1), 89–119.

Lee, H., Hong, S., & Kim, E. (2009). Neural network ensemble with probabilistic fusion and its application to gait recognition. *Neurocomputing, 72*(7), 1557–1564. doi:10.1016/j.neucom.2008.09.009.

Liang, G., & Cohn, A. G. (2013). An effective approach for imbalanced classification: Unevenly balanced bagging. In *Proceedings of the twenty-seventh AAAI conference on artificial intelligence, july 14–18, 2013, bellevue, washington, USA*.

Marasco, E., & Ross, A. (2014). A survey on antispoofing schemes for fingerprint recognition systems. *ACM Computing Surveys, 47*(2), 28:1–28:36. doi:10.1145/2617756.

Menotti, D., Chiachia, G., Pinto, A., Schwartz, W. R., Pedrini, H., FalcÃ£o, A. X., & Rocha, A. (2015). IEEE Transactions on Information Forensics and Security. *Deep Representations for Iris, Face, and Fingerprint Spoofing Detection, 10*(4), 864–879. doi:10.1109/TIFS.2015.2398817.

Merz, C. J. (1999). Using correspondence analysis to combine classifiers. *Machine Learning, 36*(1), 33–58. doi:10.1023/A:1007559205422.

Nanni, L., & Lumini, A. (2008). Local binary patterns for a hybrid fingerprint matcher. *Pattern Recognition, 41*(11), 3461–3466. doi:10.1016/j.patcog.2008.05.013.

Nayak, D. R., Dash, R., & Majhi, B. (2016). Brain mr image classification using two-dimensional discrete wavelet transform and adaboost with random forests. *Neurocomputing, 177*, 188–197. doi:10.1016/j.neucom.2015.11.034.

peng Niu, D., li Wang, F., ling Zhang, L., kuo He, D., & xing Jia, M. (2011). Neural network ensemble modeling for nosiheptide fermentation process based on partial least squares regression. *Chemometrics and Intelligent Laboratory Systems, 105*(1), 125–130. doi:10.1016/j.chemolab.2010.11.007.

Nogueira, R. F., de Alencar Lotufo, R., & Campos Machado, R. (2016). Fingerprint liveness detection using convolutional neural networks. *IEEE Transactions on Information Forensics and Security, 11*(6), 1206–1213. doi:10.1109/TIFS.2016.2520880.

Oza, N. C. (2003). Boosting with averaged weight vectors. In T. Windeatt, & F. Roli (Eds.), *Multiple classifier systems, 4th international workshop, MCS 2003, Guilford, UK, June 11–13, 2003, proceedings*. In *Lecture Notes in Computer Science: 2709* (pp. 15–24). Springer. doi:10.1007/3-540-44938-8_2.

Papouskova, M., & Hajek, P. (2019). Two-stage consumer credit risk modelling using heterogeneous ensemble learning. *Decision Support Systems, 118*, 33–45. doi:10.1016/j.dss.2019.01.002.

Platt, J. (1998). Fast training of support vector machines using sequential minimal optimization. *Advances in kernel methods – support vector learning*. MIT Press.

Polikar, R. (2006). Ensemble based systems in decision making. *IEEE Circuits and Systems Magazine, 6*(3), 21–45. doi:10.1109/MCAS.2006.1688199.

Porwik, P., Doroz, R., & Wrobel, K. (2019). An ensemble learning approach to lip-based biometric verification, with a dynamic selection of classifiers. *Expert Systems with Applications, 115*, 673–683. doi:10.1016/j.eswa.2018.08.037.

Rattani, A., Scheirer, W. J., & Ross, A. (2015). Open set fingerprint spoof detection across novel fabrication materials. *IEEE Transactions on Information Forensics and Security, 10*(11), 2447–2460. doi:10.1109/TIFS.2015.2464772.

Ruano-Ords, D., Yevseyeva, I., Fernandes, V. B., Mndez, J. R., & Emmerich, M. T. (2019). Improving the drug discovery process by using multiple classifier systems. *Expert Systems with Applications, 121*, 292–303. doi:10.1016/j.eswa.2018.12.032.

Sinha, A., Chen, H., Danu, D., Kirubarajan, T., & Farooq, M. (2008). Estimation and decision fusion: A survey. *Neurocomputing, 71*(13), 2650–2656. doi:10.1016/j.neucom.2007.06.016. Artificial Neural Networks (ICANN 2006) / Engineering of Intelligent Systems (ICEIS 2006).

Sun, J., yue Jia, M., & Li, H. (2011). Adaboost ensemble for financial distress prediction: an empirical comparison with data from chinese listed companies. *Expert Systems with Applications, 38*(8), 9305–9312. doi:10.1016/j.eswa.2011.01.042.

Syarif, I., Zaluska, E., Prugel-Bennett, A., & Wills, G. (2012). Application of bagging, boosting and stacking to intrusion detection. In P. Perner (Ed.), *Machine learning and data mining in pattern recognition* (pp. 593–602). Berlin, Heidelberg: Springer.

Ting, K. M., & Witten, I. H. (1997). Stacking bagged and dagged models. In *Proceedings of the fourteenth international conference on machine learning*. In *ICML '97* (pp. 367–375). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Verikas, A., Lipnickas, A., Malmqvist, K., Bacauskiene, M., & Gelzinis, A. (1999). Soft combination of neural classifiers: A comparative study. *Pattern Recognition Letters, 20*(4), 429–444. doi:10.1016/S0167-8655(99)00012-4.

Wang, G., Hao, J., Ma, J., & Jiang, H. (2011). A comparative assessment of ensemble learning for credit scoring. *Expert Systems with Applications, 38*(1), 223–230. doi:10.1016/j.eswa.2010.06.048.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks, 5*(2), 241–259. doi:10.1016/S0893-6080(05)80023-1.

Xia, Y., Liu, C., Da, B., & Xie, F. (2018). A novel heterogeneous ensemble credit scoring model based on bstacking approach. *Expert Systems with Applications, 93*, 182–199. doi:10.1016/j.eswa.2017.10.022.

Yambay, D., Ghiani, L., Denti, P., Marcialis, G. L., Roli, F., & Schuckers, S. A. C. (2012). Livdet 2011- fingerprint liveness detection competition 2011. In A. K. Jain, A. Ross, S. Prabhakar, & J. Kim (Eds.), *5th IAPR international conference on biometrics, ICB 2012, new delhi, india, march 29 - april 1, 2012* (pp. 208–215). IEEE. doi:10.1109/ICB.2012.6199810.

Yang, P., Hwa Yang, Y., B. Zhou, B., & Y. Zomaya, A. (2010). A review of ensemble methods in bioinformatics. *Current Bioinformatics, 5*(4), 296–308.

Yong Liu, & Xin Yao (1998). A cooperative ensemble learning system. In *1998 ieee international joint conference on neural networks proceedings. IEEE world congress on computational intelligence (cat. no.98ch36227): 3* (pp. 2202–2207 vol.3). doi:10.1109/IJCNN.1998.687202.

Zhang, X., & Mahadevan, S. (2019). Ensemble machine learning models for aviation incident risk prediction. *Decision Support Systems, 116*, 48–63. doi:10.1016/j.dss.2018.10.009.