Learning from Imbalanced Data in Face Re-Identification Using Ensembles of Classifiers

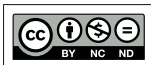
by

Roghaiyeh SOLEIMANI SAMARIN

THESIS PRESENTED TO ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
Ph.D.

MONTREAL, "JAN. 09, 2020"

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC



BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

Dr. Eric Granger, Thesis Supervisor
Department of Automated Manufacturing Engineering

Dr. Giorgio Fumera, Co-supervisor
Department of Electrical and Electronic Engineering, University of Cagliari

Dr. Stephane Coulombe, President of the Board of Examiners
Department of Software and IT Engineering

Dr. Catherine Laporte, Member of the jury
Department of Electrical Engineering

Dr. Thiago Falk, External Independent Examiner
Énergie Matériaux Télécommunications Research Centre, Institut National de la Recherche Scientifique (INRS)

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON "NOVEMBER 11, 2019"

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGEMENTS

First, I would like to express my gratitude to my supervisor Dr. Eric Granger for his guidance and support during my studies and for providing me the opportunity to expand my research expertise. I would also like to express my gratitude to my co-supervisor Dr. Giorgio Fumera for his guidance and support and special appreciation for his hospitality during my visit to University of Cagliari.

I would like to express my deepest appreciation to my husband Mohammad Honarparvar who has fully supported me in every step of the way, in both the hard and good times. I am also very grateful to my family, who have supported and encouraged me from the beginning of the whole journey of my studies even from long distances. Next, I would like to thank all of the colleagues from LIVIA. Last but not the least, I would like to appreciate my friends for being there for me specially Ibtihel, Azadeh, Hani and Fania.

RÉIDENTIFICATION FACIALE: APPRENDRE DES DONNÉES DÉSÉQUILIBRÉES À L'AIDE D'ENSEMBLES DE CLASSIFICATEURS

Roghaiyeh SOLEIMANI SAMARIN

RÉSUMÉ

La ré-identification faciale est une application de vidéosurveillance qui fait appel à des engins de reconnaissance faciale qui sont conçus à partir de visages capturés en séquences vidéo, et qui cherche à les reconnaître dans des vidéos archivées ou en direct dans un réseau de caméras vidéo. Les applications vidéo de reconnaissance faciale posent des défis importants en raison des variations de conditions de capture comme la pose ou l'éclairage. Les autres défis sont de deux ordres: 1) la distribution déséquilibrée entre les visages capturés pour les personnes à ré-identifier et les autres; 2) le degré variable de déséquilibre pendant les opérations par rapport aux données de conception. En général, il est difficile d'estimer la proportion de données déséquilibrées, en partie à cause de l'incapacité de la plupart des systèmes de classification à identifier correctement la classe majoritaire, négative ou non ciblée (visages ou images de personnes à ne pas ré-identifier) de la classe minoritaire, positive ou ciblée (visages ou images de personnes à ré-identifier), car la plupart de ces systèmes sont conçus pour des conditions de données équilibrées.

Plusieurs techniques sont proposées dans la littérature pour appendre des données déséquilibrées, soit des techniques permettant de rééquilibrer les données (en sous-échantillonnant la classe majoritaire et en sur-échantillonnant la classe minoritaire, ou les deux) pour les classificateurs de formation, soit des algorithmes permettant de guider le processus d'apprentissage (avec ou sans approche sensible aux coûts), neutralisant ainsi l'écart de performance dans la classification de la classe majoritaire. Il a été démontré que les techniques ensemblistes comme le bagging et le boosting exploitent efficacement ces méthodes pour remédier au déséquilibre. Cependant, la littérature fait aussi état de problèmes liés à ces techniques: (1) certains échantillons informatifs sont délaissés par suite d'un sous-échantillonnage aléatoire, et l'ajout d'échantillons positifs synthétiques par sur-échantillonnage augmente la complexité de la formation; (2) les facteurs de coût doivent être connus à l'avance ou trouvés; (3) les systèmes de classification sont souvent optimisés et comparés selon des mesures de performance (comme la précision) qui ne conviennent pas au problème de déséquilibre; (4) la plupart des algorithmes d'apprentissage sont conçus et testés d'après un niveau fixe de données déséquilibrées qui peut différer des scénarios opérationnels. Cette thèse a pour objectif de concevoir des ensembles de classificateurs spécialisés pour traiter la question du déséquilibre dans l'application de ré-identification faciale et, comme sous-objectifs, d'éviter les problèmes précités repérés dans la littérature. De plus, obtenir un ensemble de classificateurs efficace nécessite un algorithme d'apprentissage pour concevoir et combiner les classificateurs de composants offrant le bon compromis entre diversité et précision. Pour réaliser cet objectif, quatre contributions majeures sont présentées dans trois chapitres, dont voici un résumé.

Au chapitre 3, une nouvelle méthode d'échantillonnage sous forme d'application regroupera les échantillons du sous-échantillonnage afin d'améliorer le compromis entre diversité et précision des classificateurs de l'ensemble. Dans les applications de ré-identification faciale, la méthode d'échantillonnage proposée tire parti du fait que les régions du visage d'une même personne apparaissant dans le champ de vision d'une caméra peuvent être regroupées en fonction des trajectoires enregistrées par le localisateur facial (face tracker). Une méthode ensembliste de Bagging X est proposée pour tenir compte des variations possibles du niveau de déséquilibre des données opérationnelles en combinant des classificateurs formés à différents niveaux de déséquilibre. Dans cette méthode, tous les échantillons servent aux classificateurs de formation, minimisant ainsi la perte d'information. Au chapitre 4, un nouvel algorithme d'apprentissage ensembliste, le Boosting progressif (PBoost), insère progressivement des groupes d'échantillons non corrélés dans un processus de Boosting pour éviter la perte d'information tout en générant un groupe diversifié de classificateurs. D'une itération à l'autre, l'algorithme PBoost accumule ces groupes d'échantillons non corrélés dans un ensemble qui augmente progressivement en taille et en déséquilibre. Cet algorithme est plus sophistiqué que celui que l'on propose au chapitre 3, car au lieu de former les classificateurs de base sur cet ensemble, on les forme sur des sous-ensembles équilibrés tirés de cet ensemble et validés sur tout l'ensemble. Par conséquent, les classificateurs de base sont plus précis sans compromettre la robustesse face au déséquilibre. De plus, la sélection des échantillons est fondée sur les poids attribués aux échantillons correspondant à leur importance. Aussi, la complexité de calcul de PBoost est inférieure à celle des techniques ensemblistes de Boost dans la littérature, quant à l'apprentissage de données déséquilibrées, parce que les classificateurs de base ne sont pas tous validés sur tous les échantillons négatifs. L'on propose également un nouveau facteur de perte dans PBoost pour éviter de biaiser les performances vers la classe négative. Ce facteur de perte permet de mettre à jour le poids des échantillons et de fixer la contribution des classificateurs dans les prédictions finales en fonction de la capacité des classificateurs à reconnaître les deux classes.

Pour comparer les performances des systèmes de classification vus aux chapitres 3 et 4, il faut disposer d'un espace d'évaluation qui compare les classificateurs en fonction d'une mesure de performance appropriée sur tous leurs seuils de décision, les différents niveaux de déséquilibre des données d'essai et les différentes préférences entre les classes. La mesure F sert souvent à évaluer des classificateurs binaires par rapport aux données déséquilibrées, et aucun espace global d'évaluation de cette mesure n'a été repéré dans la littérature. Par conséquent, au chapitre 5, un nouvel espace global d'évaluation est proposé pour la mesure F, analogue aux courbes de coût par rapport au coût prévu. Dans cet espace, un classificateur est représenté par une courbe montrant sa performance sur tous ses seuils de décision et les niveaux possibles de déséquilibre quant au taux positif réel souhaité par rapport à la précision. Ces propriétés ne paraissent pas dans les espaces ROC (Receiver Operating Characteristic) et de précision-rappel. Cet espace nous permet également d'améliorer empiriquement la performance des méthodes spécialisées d'apprentissage d'ensembles déséquilibrés dans une condition opérationnelle particulière. Par la validation, les classificateurs de base sont combinés d'après une version modifiée de l'algorithme itératif de combinaison booléenne, de sorte que le critère de

sélection dans cet algorithme est remplacé par la mesure F au lieu de l'aire sous la courbe AUC (area under curve), et la combinaison est effectuée pour chaque condition de fonctionnement. Les approches proposées dans cette thèse ont été validées et comparées à partir des bases de données synthétiques et des bases vidéo Faces In Action et COX qui émulent les applications de ré-identification faciale. Les résultats montrent que les techniques proposées sont plus performantes que les techniques actuelles quant aux différents niveaux de déséquilibre et de chevauchement entre les classes.

Mots-clés: Déséquilibre des classes, Apprentissage d'ensemble, Bagging, Boosting, Mesure de la performance, Mesure-F, Outils de visualisation, Ré-identification du visage, Vidéo surveillance

LEARNING FROM IMBALANCED DATA IN FACE RE-IDENTIFICATION USING ENSEMBLES OF CLASSIFIERS

Roghaiyeh SOLEIMANI SAMARIN

ABSTRACT

Face re-identification is a video surveillance application where systems for video-to-video face recognition are designed using faces of individuals captured from video sequences, and seek to recognize them when they appear in archived or live videos captured over a network of video cameras. Video-based face recognition applications encounter challenges due to variations in capture conditions such as pose, illumination etc. Other challenges in this application are twofold; 1) the imbalanced data distributions between the face captures of the individuals to be re-identified and those of other individuals 2) varying degree of imbalance during operations w.r.t. the design data. Learning from imbalanced data is challenging in general due in part to the bias of performance in most two-class classification systems towards correct classification of the majority (negative, or non-target) class (face images/frames captured from the individuals in not to be re-identified) better than the minority (positive, or target) class (face images/frames captured from the individual to be re-identified) because most two-class classification systems are intended to be used under balanced data condition. Several techniques have been proposed in the literature to learn from imbalanced data that either use data-level techniques to rebalance data (by under-sampling the majority class, up-sampling the minority class, or both) for training classifiers or use algorithm-level methods to guide the learning process (with or without cost-sensitive approaches) such that the bias of performance towards correct classification of the majority class is neutralized. Ensemble techniques such as Bagging and Boosting algorithms have been shown to efficiently utilize these methods to address imbalance. However, there are issues faced by these techniques in the literature: (1) some informative samples may be neglected by random under-sampling and adding synthetic positive samples through upsampling adds to training complexity, (2) cost factors must be pre-known or found, (3) classification systems are often optimized and compared using performance measurements (like accuracy) that are unsuitable for imbalance problem; (4) most learning algorithms are designed and tested on a fixed imbalance level of data, which may differ from operational scenarios;

The objective of this thesis is to design specialized classifier ensembles to address the issue of imbalance in the face re-identification application and as sub-goals avoiding the above-mentioned issues faced in the literature. In addition achieving an efficient classifier ensemble requires a learning algorithm to design and combine component classifiers that hold suitable diversity-accuracy trade off. To reach the objective of the thesis, four major contributions are made that are presented in three chapters summarized in the following. In Chapter 3, a new application-based sampling method is proposed to group samples for under-sampling in order to improve diversity-accuracy trade-off between classifiers of the ensemble. The proposed sampling method takes the advantage of the fact that in face re-identification applications, facial regions of a same person appearing in a camera field of view may be regrouped based on their trajectories found by face tracker. A partitional Bagging ensemble method is proposed

that accounts for possible variations in imbalance level of the operational data by combining classifiers that are trained on different imbalance levels. In this method, all samples are used for training classifiers and information loss is therefore avoided.

In Chapter 4, a new ensemble learning algorithm called Progressive Boosting (PBoost) is proposed that progressively inserts uncorrelated groups of samples into a Boosting procedure to avoid loosing information while generating a diverse pool of classifiers. From one iteration to the next, the PBoost algorithm accumulates these uncorrelated groups of samples into a set that grows gradually in size and imbalance. This algorithm is more sophisticated than the one proposed in Chapter 3 because instead of training the base classifiers on this set, the base classifiers are trained on balanced subsets sampled from this set and validated on the whole set. Therefore, the base classifiers are more accurate while the robustness to imbalance is not jeopardized. In addition, the sample selection is based on the weights that are assigned to samples which correspond to their importance. In addition, the computation complexity of PBoost is lower than Boosting ensemble techniques in the literature for learning from imbalanced data because not all of the base classifiers are validated on all negative samples. A new loss factor is also proposed to be used in PBoost to avoid biasing performance towards the negative class. Using this loss factor, the weight update of samples and classifier contribution in final predictions are set according to the ability of classifiers to recognize both classes.

In comparing the performance of the classifier systems in Chapter 3 and 4, a need is faced for an evaluation space that compares classifiers in terms of a suitable performance metric over all of their decision thresholds, different imbalance levels of test data, and different preference between classes. The F-measure is often used to evaluate two-class classifiers on imbalanced data, and no global evaluation space was available in the literature for this measure. Therefore, in Chapter 5, a new global evaluation space for the F-measure is proposed that is analogous to the cost curves for expected cost. In this space, a classifier is represented as a curve that shows its performance over all of its decision thresholds and a range of possible imbalance levels for the desired preference of true positive rate to precision. These properties are missing in ROC and precision-recall spaces. This space also allows us to empirically improve the performance of specialized ensemble learning methods for imbalance under a given operating condition. Through a validation, the base classifiers are combined using a modified version of the iterative Boolean combination algorithm such that the selection criterion in this algorithm is replaced by F-measure instead of AUC, and the combination is carried out for each operating condition.

The proposed approaches in this thesis were validated and compared using synthetic data and videos from the Faces In Action, and COX datasets that emulate face re-identification applications. Results show that the proposed techniques outperforms state of the art techniques over different levels of imbalance and overlap between classes.

Keywords: Class Imbalance, Ensemble Learning, Bagging, Boosting, Performance Metrics, F-measure, Visualization Tools, Face Re-Identification, Video Surveillance

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

LIST OF ABREVIATIONS

Ada	AdaBoost
AUC	Area Under Receiver Operating Characteristic curve
AUPR	Area Under Precision-Recall curve
BC	Brier Curve
CC	Cost Curve
CUS	Cluster Under Sampling
CUSG	Cluster Under Sampling with Growing imbalance
CNN	Convolutional Neural Networks
CSB	Cost Sensitive Boosting
EUS	Evolutionary Under-Sampling
EoC	Ensemble of Classifiers
EC	Expected Cost
FP	False Positives
FPR	False Positive Rate
FN	False Negatives
FNR	False Negative Rate
FIA	Face In Action
G-mean	Geometric Mean
ID	Identification

IBC	Iterative Boolean Combination
KNORA	K Nearest Oracles
LBP	Local Binary Pattern
MSMOTE	Modified Synthetic Minority Over-Sampling Technique
NEC	Normalized Expected Cost
Pr	Precision
PR	Precision-Recall
PRUS	Progressive Random Under Sampling Boosting
PCUS	Progressive Cluster Under Sampling Boosting
PTUS	Progressive Trajectory Under Sampling Boosting
RUS	Random Under-Sampling
RUSG	Random Under-Sampling with Growing imbalance
RUSwR	Random Under-Sampling with Replacement
RUSwRG	Random Under-Sampling with Replacement with Growing imbalance
RSSM	Random Sub Space Method
RB	Random Balance
ROI	Region of Interest
RBF	Radial Basis Function
Re	Recall
ROC	Receiver Operating Characteristic

ROCCH	Receiver Operating Characteristic Convex Hull
RTUS	Random Trajectory Under Sampling
SVM	Support Vector Machines
SV	Support Vector
SMT	SMOTEBoost
SMOTE	Synthetic Minority Over-Sampling Technique
SSBC	Skew Sensitive Boolean Combination
SeEn-SVM	Segmented Ensemble of SVMs
TP	True Positives
TPR	True Positive Rate
TN	True Negatives
TNR	True Negative Rate
TUS	Trajectory Under Sampling
TUSA	Sorted Trajectory Under Sampling in Ascending order
TUSD	Sorted Trajectory Under Sampling in Descending order

LIST OF SYMBOLS AND UNITS OF MEASUREMENTS

α_e	Weight update factor of the e^{th} iteration of Boosting algorithm
α	Parameter of the F-measure performance mesasure
β	Parameter of the F-measure performance measure
A	Accuracy
C_e	The e^{th} classifier
C_{FN}	The cost of misclassifying the positive class samples as negatives
C_{FP}	The cost of misclassifying the negative class samples as positives
δ	The margin distance between positive and negative class cluster centers
d	The number of features in training an SVM
d_t^e	The decision vector corresponding to the t^{th} threshold of the e^{th} classifier
$D_{1,2,3}$	Synthetic training data generated under different settings
D^{cc}	The number of times both classifiers make the correct decision
D^{cw}	The number of times first classifier makes the correct decision and the second classifier makes the wrong decision
D^{ww}	The number of times both classifiers make the correct decision
D^{wc}	The number of times first classifier makes the wrong decision and the second classifier makes the correct decision
ε_e	The loss factor of the e^{th} classifier in the Boosting algorithm
E	The number of classifiers
EC	Expected cost

F_α	The F-measure in terms of the α parameter.
F_{op}	The F-measure when the optimal threshold is selected using the validation step.
F_{D}	The F-measure when the combination function in Boosting ensembles is majority voting of decisions.
γ	The optimal Bayes decision threshold.
$h_e(\mathbf{x})$	The output of C_e (either a classification score or a label)
$H(\mathbf{x})$	The final output of an ensemble of E classifiers
HD_j	Hausdorff distance of the j^{th} non-target trajectory from the target trajectory
ID_i	The track ID assigned by tracker to the i^{th} face
κ	Kappa diversity measure
$K(x', x'')$	RBF kernel of SVM classifier
l^+	The pseudo error of classifier in classifying positive class
l^-	The pseudo errors of classifier in classifying negative class
L_j	The number of classifiers that correctly classify x_j
L_e	The loss factor in the e^{th} step of the PBoost algorithm
λ	Skew level of data $P(-)/P(+)$
Λ	Skew level of data defined as $1:\lambda$
m	The factor of EC for the misclassification costs trade-off
μ_i	The cost factor of sample $\mathbf{x}_i$
M	The number of labelled training samples

M^+	The number of positive training samples
M^-	The number of negative training samples
n_T^-	The number of non-target trajectories
n_{tr}	The number of samples an SVM is trained on
n_{val}	The number of samples an SVM is validated on
n_{SV}	The number of Support Vectors in an SVM model
N_e^P	The number of negative samples in each partition
NEC	The normalized expected cost
$N(m_+, \sigma_+)$	Normal distribution of the target trajectory
$N(m_{-,j}, \sigma_-)$	The normal distribution of a non-target trajectory
P_e	The e^{th} disjoint partition of the negative class
$P(+)$	The probability of the positive class
$P^*_{\text{i,j}}(+)$	The crossing point of two F-measure curves
$q_{i,k}$	Pairwise Q-statistic diversity between two classifiers
Q_{av}	The average Q-statistic diversity between several classifiers
$\mathbf{S}_{\text{tr}}$	The training samples set
SV_j	The j^{th} Support Vector in an SVM model
$\mathbf{t}^+$	The target trajectory
$\mathbf{t}_j^-$	The non-target trajectory
$\mathbf{T}^-$	A collection of non-target trajectories

$\mathbf{T}_s^-$	A collection of non-target trajectories sorted by distance to the target trajectory
$\mathbf{T}_R^-$	A collection of non-target trajectories sorted randomly
Th_t^e	The t^{th} threshold of the e^{th} classifier
$\mathbf{W_e}$	Weight matrix of training sample in the e^{th} iteration of Boosting algorithm
X	Probe sample
$\mathbf{x}_i$	The i^{th} training sample
y_i	True label of the i^{th} training sample
ϕ_+	Cost adjustment function for the positive class samples
ϕ_-	Cost adjustment function for the negative class samples

INTRODUCTION

Biometric recognition of individuals measures and analyzes the unique intrinsic human physical or behavioural characteristics for automatic recognition of a person's identity. One of the applications of face biometric is in video surveillance systems. A video surveillance system consists of a number of video cameras used at airports, banks, department stores etc., for security purposes. In the traditional video surveillance systems, videos are displayed for human operators to detect specific individuals of interest or behaviours. The more the number of cameras increases the more degrades the human operators' performance. The new video surveillance systems are improving by utilizing automatic face recognition algorithms for assisting or replacing human operators. Applications of face recognition in video surveillance include still-to-video face recognition, and video-to-video face recognition (or face re-identification). Face re-identification is the focus of this thesis and refers to recognition of individuals (online or offline) viewed in different video streams at different time instants and/or locations over a network of cameras using systems that are trained on faces captured from videos. A typical face re-identification system consists of a segmentation module in order to separate the region of interest (ROI) of face(s) in the scene from the background, a feature extraction module to extract more compact while useful information form the face, and a recognition module which classifies the probe face using the face models from the individual(s) of interest in the gallery using the feature vectors. The most common classification architecture in the literature for face re-identification consists of one-class or two-class classifiers or an ensemble of them per individual of interest (Radtke *et al.*, 2014; Pagano *et al.*, 2014; De-la Torre *et al.*, 2015c).

The video-based face recognition systems can be divided into two categories. In spatial approaches (Martinel & Foresti, 2012; Bazzani *et al.*, 2012; Stallkamp *et al.*, 2007; Taigman *et al.*, 2014a; Schroff *et al.*, 2015; Parkhi *et al.*, 2015; Ding & Tao, 2017) videos are treated as several images of the face.

In spatio-temporal approaches (Matta & Dugelay, 2007; Ahonen *et al.*, 2006; Hadid & Pietikäinen, 2009; Ye & Sim, 2010; Mitra *et al.*, 2006; Gheissari *et al.*, 2006; Pagano *et al.*, 2014; De-la Torre *et al.*, 2015c) the temporal dynamics in video is also exploited in various ways within the recognition process. In this category, motion information may be used as features in addition to face or appearance (Matta & Dugelay, 2007; Ahonen *et al.*, 2006; Hadid & Pietikäinen, 2009; Ye & Sim, 2010; Mitra *et al.*, 2006; Gheissari *et al.*, 2006), or tracking and recognition may be merged into a single task (Saeed *et al.*, 2006; Zhou *et al.*, 2004) or as individual tasks where the results are combined (Matta & Dugelay, 2007; Mazzon *et al.*, 2012; Tao & Veldhuis, 2009; Pagano *et al.*, 2014; De-la Torre *et al.*, 2015c). In such systems, a face tracking system also appears that finds the location of a face present in a frame by utilizing its location information from the previous frames. Once a face is located and tracked over video frames, the ROIs over these frames are collected into a trajectory.

Problem Statement:

Face re-identification is challenging since facial captures are obtained from video streams that are captured unobtrusively under semi- or uncontrolled conditions. Hence, the conditions of the individuals such as pose and expression, as well as the effects of the environment such as illumination, occlusion and blur can vary significantly over time. One important challenge in this application (that is the focus of this thesis) is that the number of reference samples of individuals of interest is limited and excessively outnumbered by those of others. Therefore, there exists imbalance between the number of target (positive or minority) and non-target (negative or majority) classes. In addition, the skew level during operations may differ from what is considered during design.

Designing most of the conventional classification systems with imbalanced data results in a classification decision boundary skewed towards the minority class and as a consequence low accuracy is achieved in recognizing positive samples. Several approaches have been proposed

in the literature to train classification systems from imbalanced data to avoid this bias of performance. Data level approaches exploit data pre-processing methods prior to classification without any modification in the classification algorithm. These approaches involve up-sampling the minority (target, or positive) class and/or under-sampling the majority (non-target, or negative) class. Algorithm-level approaches define or modify learning algorithms to bias the performance of the classification systems in favour of the minority class and therefore neutralize the bias of performance towards the majority class caused by training on imbalanced data. These algorithms may make use of cost-sensitive methods that improve the classification performance by assigning different misclassification costs to the classes in the learning algorithm. However, there exists algorithm-level methods that guide the learning process without using different misclassification costs.

Ensemble-based methods define ensemble learning algorithms using one of the data-level, cost-sensitive, or cost-free methods, or a combination of them, to train a combination of a set of classifiers for achieving a high level of performance in terms of classification accuracy as well as robustness to imbalance.

Performance of an ensemble of classifiers depends on different factors. The base classifiers that are included in the ensemble must maintain a good level of diversity-accuracy trade-off and should be combined in an effective way. It means that an ensemble outperforms its component classifiers only if the component classifiers provide complementary and accurate enough recognition of the classes, and if these complementary information are combined properly. In addition, a suitable performance measure plays an important rule in optimization of designing an ensemble.

Specialized Boosting and Bagging ensembles (Freund, 1995; Freund *et al.*, 1996; Breiman, 1996; Yan *et al.*, 2003; Li *et al.*, 2013) are commonly used for the issue of class imbalance in the literature. In these ensemble methods diversity is maintained by training component

classifiers on different subsets of data. Boosting and Bagging algorithms (Freund, 1995; Freund *et al.*, 1996) train classifiers iteratively. In Bagging, a bootstrap of data is used to train a classifier in each iteration. Bootstrap are also used in Boosting algorithm to train classifiers iteratively with the difference that in this algorithm training samples are assigned with weights (analogous to misclassification costs) that are updated from one iteration to the next. The weight update is carried out such that from one classifier to the other, more focus is given to correctly classifying the samples that are more difficult to classify. In addition to the training samples in Boosting algorithms, the classifiers are also assigned with weights that determine their contribution in final decision. In data-level Boosting and Bagging ensembles, the learning algorithm is modified by under-sampling the majority class (Seiffert *et al.*, 2010; Barandela *et al.*, 2003) or up-sampling the minority class (Chawla *et al.*, 2003; Hu *et al.*, 2009) , or both (Wang & Yao, 2009; Díez-Pastor *et al.*, 2015). In cost-sensitive Boosting ensembles (Fan *et al.*, 1999; Ting, 2000; Sun *et al.*, 2007), misclassification costs control the weight update and the contribution of each classifier in the ensemble in final prediction.

The challenges that are faced in the literature of classifier ensembles for imbalanced data learning using the aforementioned methods are summarized as following:

- Under-sampling the majority class may put the learning algorithm in the risk of information loss, and up-sampling the target class increases the computation cost.
- The performance level of cost sensitive methods relies on the proper selection of cost values.
- In designing most of existing classification systems in the literature, the imbalance level is considered fixed and known a priori, while in reality the level of imbalance is unknown and may vary over time.

- Most of existing classification systems in the literature are evaluated using performance metrics and spaces that are not suitable for imbalance. In addition these classification systems are rarely optimized in terms of the suitable performance metrics for imbalance.

Objectives and structure of the thesis:

The main objective of this thesis is to design specialized classifier ensembles to address the issue of imbalance in recognition of faces from video in face re-identification application. This system must avoid bias of performance towards the correct classification of the negative class and result in a high performance in correctly classifying faces from videos in presence of imbalance. The design of such systems must handle the challenges of the methods in the literature for imbalanced data classification in general and in face re-identification application.

First, the literature is reviewed in Chapter 1 for (1) face re-identification methods (2) imbalanced data classification approaches, and (3) performance metrics to evaluate classification systems under imbalance. Then in Chapter 2, the experimental methodology used in this thesis is presented.

A sample selection technique called Trajectory Under-Sampling (TUS) is proposed in Chapter 3 that is specialized for face re-identification to design ensembles of classifiers trained on imbalanced data. This sampling technique benefits from the fact that there exists some natural sub-clusters in this application that can be found using tracking information. In fact this sampling method can be used in any application where there exists some natural sub-clusters and use one-versus-all learning strategy. Two ensembles are proposed using TUS and it appears to be more effective than the general-purpose sampling methods in the literature for this application because TUS improves the diversity-accuracy trade off between classifiers of the ensemble for this application. In the proposed ensembles the classifiers are trained on data subsets that have different imbalance levels and the combination of them is optimized based

on their performance in terms of F-measure. Then we extend this ensemble method by further under-sampling the trajectories to support vectors with the intention of decreasing the bias of performance in classifiers that are trained on imbalanced subsets and to use only the most important information from the data. The content of Chapter 3 is an extended version of a conference paper presented in ICPRAM 2016.

In the next step (Chapter 4), a new Boosting ensemble method is proposed that is called Progressive Boosting (PBoost). In this Boosting algorithm, the skew level of data used for designing the classifiers in the ensemble increases progressively. In contrast to the ensembles proposed in Chapter 3, the classifiers in the proposed ensemble in this chapter are not trained on imbalanced subsets of data and are only validated on imbalanced subsets of data. Instead, the classifiers are trained on balanced sets because training on balanced subsets of data increases the accuracy of each individual classifier in the ensemble which can lead to an overall better performance of the ensemble. These subsets are drawn from a growing imbalanced set and this selection depends on the importance of samples based on weights assigned through the proposed Boosting learning strategy. The classifiers in this ensemble are validated on this growing set and therefore result in an ensemble with more robustness to varying imbalance. The weights of the samples and the classifiers in this ensemble are updated based on the performance in terms of F-measure from one iteration to the other. To this aim, the loss function of the ensemble is modified using the F-measure and each classifier is assigned with a weight according to its performance in terms of F-measure. Similarly to the ensemble methods proposed in Chapter 3, the proposed TUS appears to be more effective for this application than the general-purpose sampling methods in the literature. However, note that the proposed PBoost algorithm can be used in any application since if there is no application-based under-sampling method for an application, this algorithm works with general-purpose sampling methods like random under-sampling and cluster under-sampling.

A part of this work that presents the modification of loss function in Boosting ensemble has been published in a conference paper (ICPR2016). The full content of this chapter is published as a regular article in the journal of Expert Systems with applications from Elsevier.

In experiments of Chapters 3 and 4, we compare the classification systems under different imbalance levels in terms of F-measure to evaluate their accuracy as well as their robustness to varying imbalance levels during testing. For this purpose an evaluation space is required that compares classifiers in terms of a suitable performance metric over all of their decision thresholds, different imbalance levels of test data, and different preference between classes. The F-measure has some advantages over other performance metrics for evaluating and comparing the classifiers' performance when data is imbalanced. However, no performance evaluation space presented in the literature allows to compare classifiers in terms of F-measure over all of their decision thresholds, different imbalance levels of test data, and different preference between classes. What's more, the other existing performance evaluation spaces in the literature lack at least one of aforementioned three properties. Therefore, a new space is designed to have all the above properties in Chapter 5. The new space is thoroughly analytically investigated to understand its properties, and it is described how to compare classifiers using it. Then, as a possible application, it is used to design a modified version of adaptive Iterative Boolean Combination (IBC) algorithm. With this algorithm, the selection and combination step of the ensemble learning are adapted to the varying imbalance levels during test in terms of the F-measure. The ensembles proposed in Chapters 3 and 4 are designed statically and account for the varying imbalance level of data during design stage. However, the ensembles in this Chapter are designed adaptively to the estimated skew level of data. The content of this chapter is partially presented in a conference paper (ANNPR2018) and the full content is accepted as a regular article for the Pattern Recognition journal.

Note that the proposed ensemble algorithms in this thesis can be implemented with any discriminative classifier such as logistic regression, neural networks, nearest neighbour, etc.

Contributions and the list of publications:

In summary, the main contributions of this thesis, the issues that they target to solve and the resulting publications are as follows:

1. Proposing an efficient under-sampling method for improving diversity-accuracy trade-off in designing ensembles in face re-identification application and proposing two Bagging ensemble methods using the proposed under-sampling method to increase robustness to varying imbalance (the 5th ICPRAM (Soleymani *et al.*, 2016b)).

2. Proposing a new Boosting ensemble for imbalanced data classification (Expert Systems with Applications (Soleymani *et al.*, 2018b)) with the following contributions:

   - Optimizing the loss function of the Boosting algorithm using the F-measure to avoid bias of performance (the 23rd ICPR (Soleymani *et al.*, 2016c)).
   - Selecting the training subsets from a growing set based on their importance while minimizing the risk of information loss.
   - Validating the base classifiers on varying imbalance levels in order to increase their robustness to imbalance and reducing the computation complexity.

3. Proposing a new global performance evaluation space for the scalar metric; the F-measure (ANNPR (Soleymani *et al.*, 2018a), Pattern Recognition Journal (Soleymani *et al.*, 2019)) with the following properties:

   - Possibility of visualizing the performance of any classifier (soft or crisp) under different imbalance levels of test data.

- Possibility of comparing more than two classifiers over different decision thresholds and under different imbalance levels of test data with the ability of selecting a preference level between classes.

4. Modifying the Iterative Boolean Combination (IBC) method to adapt the selection and combination of classifiers in the ensemble using the proposed F-measure space (Pattern Recognition Journal (Soleymani *et al.*, 2019)).

- **Journal publications:**

1. Roghayeh Soleymani, Eric Granger and Giorgio Fumera "Progressive Boosting for Class Imbalance and Its Application to Face Re-Identification.", Expert Systems with Applications (Soleymani *et al.*, 2018b).
2. Roghayeh Soleymani, Eric Granger and Giorgio Fumera " F-Measure Curves: A Tool to Visualize Classifier Performance Under Imbalance ", Pattern Recognition Journal (Soleymani *et al.*, 2019)).

- **Conference publications:**

1. Roghayeh Soleymani, Eric Granger and Giorgio Fumera "Classifier Ensembles with Trajectory Under-Sampling for Face Re-Identification", Lecture in the 5th ICPRAM, 2016 (Soleymani *et al.*, 2016b). This paper has won the best student paper award and was invited to be included in the series "Lecture Notes in Computer Science" (LNCS) published by Springer.
2. Roghayeh Soleymani, Eric Granger and Giorgio Fumera "Loss Factors for Learning Boosting Ensembles from Imbalanced Data", Lecture in the 23rd ICPR, 2016 (Soleymani *et al.*, 2016c).

3. Roghayeh Soleymani, Eric Granger and Giorgio Fumera "F-Measure Curves for Visualizing Classifier Performance with Imbalanced Data", ANNPR, 2018 (Soleymani *et al.*, 2018a).

CHAPTER 1

A REVIEW OF TECHNIQUES FOR LEARNING FROM IMBALANCED DATA IN FACE RE-IDENTIFICATION

1.1 Face Recognition from Video Surveillance

One of the applications of biometrics especially face is in video surveillance systems. A video surveillance system consists of a number of video cameras used at airports, banks, department stores etc. for security purposes. In the traditional video surveillance systems, a number of videos are displayed for human operators to detect specific people or behaviours. Human operators have low performance in this situation because the scene containing a person or behaviour of interest might not appear so often and also the time spent in watching videos is a problem. The new video surveillance systems are improving by utilizing automatic face recognition algorithms for assisting or replacing human operators.

Face recognition can be applied in many video surveillance systems. Still-to-video face recognition is used in watch list screening application. In this type of application, a high quality image of a person is captured and a model of the face is created and stored in the gallery during enrolment. In deployment (or operational) stage, the faces are captured from the video streams and the model of the detected person's face is compared to the models in the gallery. In systems for video-to-video face recognition (face re-identification) the reference models are created from the face captures from video streams. In a more general form of "Person re-identification", both face and/or overall appearance of the individuals could be used to recognize them.

Video possesses two main characteristics that make it more practical for recognition than still images:

- Video of a person contains multiple images of that person and might be captured in different uncontrolled conditions from the operational environment. Therefore, it includes different

types of information compared to still images. For example, different pose angles of a person's face in video could provide extra information of the face.

- Video affords temporal information. For example, the trajectory of feature points along consecutive frames differs from one person to the other (Zhao *et al.*, 2003).

A typical face recognition system for video surveillance that use both spatial and temporal information from video is comprised of five major modules which are face segmentation (detection), feature extraction, classification, decision modules, and tracking module. Figure 1.1 presents a general block diagram for this application. The segmentation (face-head detector) module initiates a new track when the face appears in the scene and extracts the region of interest (ROI) of the face (or the bounding box around the face). The tracker employs local object detection to find the location of the face in the video frame based on the information from the previous frames. Therefore, the tracker must periodically interact with the detector (segmentation module) to initiate new tracks, or to validate and/or update the object template with new detector bounding boxes (Kiran *et al.*, 2019) (see papers by Breitenstein *et al.* (2009); Smeulders *et al.* (2014); Huang *et al.* (2019) for more on visual tracking and tracking-by-detection). Then the features are extracted by the feature extraction module from the ROIs to be used in the classification module.

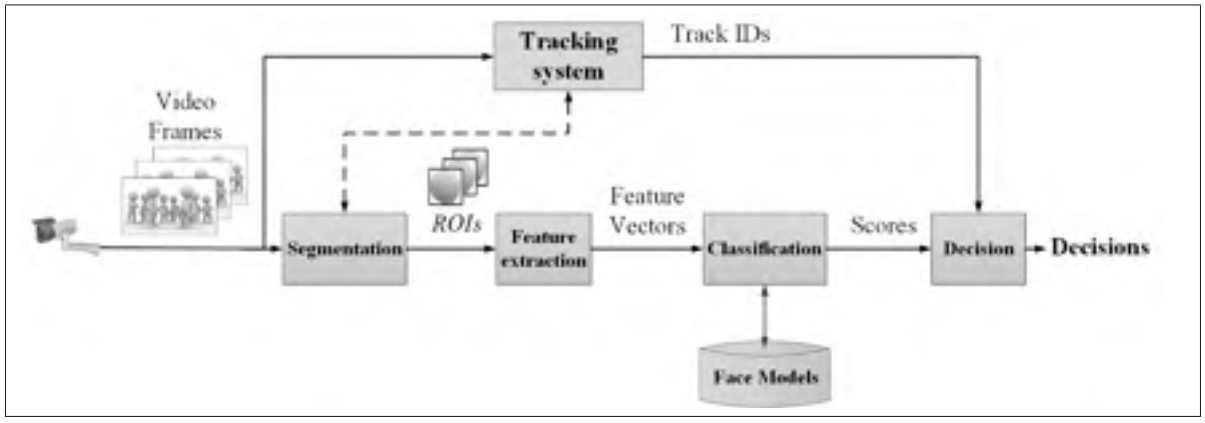


Figure 1.1 General system for face recognition in video surveillance.

The classification module identifies the probe face based on the face models in the gallery. In a common classification architecture for face re-identification a single classifier or an ensemble of classifiers (EoC) is designed for each individual of interest (Radtke *et al.*, 2013; Pagano *et al.*, 2014; De-la Torre *et al.*, 2015a). The classifiers are typically one or 2-class classifiers. The face models in the gallery may be feature vectors that are used directly by a face matcher, or classification models learned during design. The decision module makes the final decision based on the results of the classification along with tracking. In some spatial systems the video is treated as several still images of the face and therefore the tracking information is not used and the decision module is integrated in the classification module. In addition, in deep neural network architectures, the feature extraction and classification is carried out in an end-to-end system. A review of the literature in face recognition from video is presented in the following sub-section.

Learning and recognition of faces from video is challenging due to uncontrolled face capture conditions such as variations in pose, illumination, expression, etc. Another challenging problem in this application, which appears in many other machine learning applications, is the imbalance between classes because the number of face captures from the individual of interest is greatly outnumbered by those of others. The issue of imbalance in this application affects the performance of the classification module in Figure 1.1 and is the main focus of this thesis. In the rest of this section, after the literature in face recognition from video, the methods from literature for the issue of classification under imbalance are reviewed in section 1.2.

1.1.1 Techniques in Face Re-Identification

The proposed methods for video-based face recognition in the literature are generally divided into two main categories based on the type of the video properties they use. The first category is spatial approaches that consider video as a set of images. The second category is spatiotemporal approaches that take advantage of temporal information even partially.

Spatial methods consider the face re-identification as a problem of matching a set of multiple samples from a person. In fact the video frames are considered as several independent images from the same person without considering the temporal relation among them. The images from the individual of interest are used to build the face model for that individual. On the other hand, the images from the probe are matched against the face models of the individuals of interest and the results are combined in four possible levels. The sensor-level, feature-level, score-level and decision-level are widely used in the spatial methods (Martinel & Foresti, 2012).

Some spatial methods extract features from face and soft biometrics such as clothing and combine them for each frame (sensor-level fusion) and then perform the matching for each frame and get the final result by combining the matching scores or decisions of all frames (score-level and decision-level) (Martinel & Foresti, 2012). Some approaches only use the feature-level fusion and extract one representative feature by combining the same features extracted from several frames (Bazzani *et al.*, 2012). Some others only use score level or decision level fusion by combining the classification score or decision obtained from matching each input frame (Stallkamp *et al.*, 2007; Leng *et al.*, 2013).

Unlike spatial approaches, spatiotemporal approaches exploit the significant characteristic of video which is temporal dynamics to correctly recognize individuals especially in uncontrolled and complex environments. Information fusion techniques are also used in this type of methods to combine the spatial and temporal information for recognition in different levels of feature, score or decision.

Spatiotemporal methods may use motion information in addition to face or appearance as features (Matta & Dugelay, 2007; Ahonen *et al.*, 2006; Hadid & Pietikäinen, 2009; Ye & Sim, 2010; Mitra *et al.*, 2006; Gheissari *et al.*, 2006) and/or employ tracking along with spatial methods for efficiency improvement especially in the cases that face is temporarily occluded or the face is not clear due to illumination or motion blur (Pagano *et al.*, 2012; Mazzon *et al.*, 2012; Barry & Granger, 2007; Tao & Veldhuis, 2009; Pagano *et al.*, 2014; De-la Torre *et al.*, 2015c). These methods may do tracking and recognition as individual tasks and combine the

results (Matta & Dugelay, 2007; Mazzon *et al.*, 2012; Tao & Veldhuis, 2009; Pagano *et al.*, 2014; De-la Torre *et al.*, 2015c) or merge tracking and recognition to a single task (Saeed *et al.*, 2006; Zhou *et al.*, 2004). For example, in the work of (Pagano *et al.*, 2012, 2014; De-la Torre *et al.*, 2015c), features of the face are extracted and a user-specific ensemble of two-class classifiers is trained to recognize the face. They also use a tracker to build the face trajectories in parallel to the classification system. The results of the classification and tracking are then accumulated to make a reliable spatiotemporal detection of persons of interest. In the work of Mazzon *et al.* (2012), appearance features are extracted and the movement of the person in non-observed regions is modelled to find the candidate positions for the person. The information from appearance features and tracking are combined for re-identification.

1.1.2 State of the Art Deep Learning Methods

Deep Learning (DL) model refers to the models that are learned by neural networks that have numerous layers to learn from experience on large data sets. The most common type of deep learning architectures for face recognition are end-to-end Convolutional neural networks (CNNs), also known as a ConvNets. The hidden layers of a ConvNet typically consist of convolutional layers, pooling layers and fully connected layers. The basics of CNNs can be learned from the book (Aghdam & Heravi, 2017). In end-to-end systems that use ConvNets for face recognition, features are learned from the training data (face ROIs), instead of being designed as in the traditional face recognition systems. Therefore, in order to generalize better, they must be trained with very large datasets that contain enough variations in capture conditions such as pose and illumination.

After training a CNN architecture, face ROIs can be represented as the output of the fully connected layer before the last layer of the CNN architecture, known as the bottleneck features in the literature. Then the classification system can be trained on these feature vectors. This classification system can be more fully-connected layers (a neural network) or any other classifier (like an SVM), or ensembles of classifiers. Another approach to use CNNs for face recognition is to train an end-to-end architecture with a purpose of verification or identification.

The methods in literature for face recognition using CNNs are reviewed by Trigueros *et al.* (2018) and Guo & Zhang (2019). Bashbaghi *et al.* (2019) also review the still-to-video face recognition systems using CNNs considering the single sample per person problem and Ambeth Kumar *et al.* (2019) present a survey on face recognition in video surveillance. Here, we focus on the systems for video-to-video face recognition using deep learning methods. These methods can also be divided into spatial and spatio-temporal methods.

Spatial methods treat face captures from the videos as several still images and use still-to-still neural network architectures. Siamese network (Chopra *et al.*, 2005) was proposed as an end-to-end system for face verification. During training, contrastive loss function is used to optimize two identical networks such that each of them see different training samples. The output of this network obtains feature descriptors for face captures that can be used for verification or identification.

Zhu *et al.* (2013) propose a deep auto-encoder network that reconstructs a face in the canonical view in order to eliminate variations to pose and illumination. This network has two parts. The first part has three locally connected layers to extract features. The second part reconstructs a face using a fully-connected reconstruction layer.

In a series of papers by Sun et al. (Sun *et al.*, 2014a,b, 2015b,a), ConvNets are learned to obtain deep identification-verification features (Deep hidden IDentity features (DeepID)). The proposed features are extracted from various local face regions centered around the five facial landmarks (two eye centers, nose tip, and two mouse corners). Feature extraction is then followed by either a softmax layer for identification or any classifier for verification.

In DeepFace method proposed by Taigman *et al.* (2014b), a 3D alignment step is used which is followed by Siamese network. The network has 6 locally connected layers to extract features and two fully connected layers to output the descriptor vector and to classify the face captures.

Hu *et al.* (2014) propose a new discriminative deep metric learning (DDML) method. In this method, a set of hierarchical nonlinear transformations are learned by training a deep neural

network. The face pairs are projected by these transformations into a feature subspace which are used for face verification.

In FaceNet, Schroff *et al.* (2015) add triplet-loss optimization step to an architecture similar to Siamese network but replaced L1-norm with L2-norm. The aim of triplet-loss optimization step is increasing the discrimination capability of network between different identities while increasing the accuracy of correctly detecting face captures from the same identity. This loss is applied at multiple layers, not just the final one.

In Light CNN Wu *et al.* (2015), a semantic bootstrapping method is proposed. Using this method, the prediction of the networks become more consistent with noisy labels and exploiting this method in training the networks reduces the number of parameters and computational costs.

VGGFace (Parkhi *et al.*, 2015), train a very deep CNN architecture to identify or verify face images.

In (Chen, 2017), triplet loss is used in a framework to identify few suspects from the crowd in real time for public video surveillance. They use Inception-ResNet-v2 (Szegedy *et al.*, 2017) for feature extraction where the final softmax layer is replaced with a L2-normalization layer.

(Wang *et al.*, 2017) automatically collect and label the data from real-world surveillance videos and then fine-tune VGGFace (Parkhi *et al.*, 2015) with the constructed dataset.

There exists a few spatio-temporal methods in the literature that are designed specially for video-to-video face recognition. A temporal slowness principle is used by Zou *et al.* (2012) to propose an unsupervised learning method for learning invariant features from videos. Domain-invariant discriminative representations are learned in the method proposed by Sohn *et al.* (2017) using an image to video feature-level domain adaptation approach. The proposed method uses a pre-trained face recognition CNN on labeled still images to learn discriminative features, and then adapts them to video domain by synthetic data augmentation. Then domain-

invariant feature are learned through a domain adversarial discriminator. ASML (Hu *et al.*, 2017) measure the statistical characteristics of image sets for video-based face recognition.

1.1.3 Challenges of Face Re-Identification

Systems for video-to-video face recognition deal with various problems. In video surveillance, faces are captured under semi- or unconstrained and complex conditions such as:

1. Scale: The individuals detected in the video streams might move or appear in different distances from the cameras. Therefore, the captured faces may vary in scale from one person to the other or from one frame to the other.
2. Pose: People passing through the surveillance regions may change directions in reference to the location of the camera or move their heads in different directions. Therefore, their pose may vary significantly in different video frames
3. Expression: People may show different facial expressions in different frames of the video.
4. Motion blur: If the person moves rapidly, motion blur may happen.
5. Illumination: While an individual is moving from one place to another, the illumination may increase or decrease during this change of location.
6. Occlusion: Faces might be blocked with other people or objects in the environment.

In addition to the mentioned capture conditions, the quality of the face images may be poor due to low resolution of the camera i.e. face captures in face re-identification may be obtained from low quality video streams.

Another issue is that, most of the people in the video streams in video surveillance systems are not of interest. Consequently, a huge amount of comparisons are carried out for finding a small number of people. In fact, the processing time grows with the number of persons, number of cameras, frame rate, etc.

There is another important challenge in this application which is the main focus of this thesis. The number of non-target individuals in the scene might vary significantly in time and is unknown a priory. Thus, there is a significant imbalance between target and non-target individuals that varies by time. Most of the standard learning algorithms degrade with imbalanced data because they are optimized with an assumption of balanced data distribution. The effects of imbalance on learning algorithms and a review of approaches from the literature to mitigate these effects is presented in more detail in the next section.

1.2 Classification in Imbalanced Environment

The data class distributions in many real-world applications are imbalanced. However, the class with fewer samples is usually of more interest than the others. In this section the effect of imbalance on the performance of one and 2-class classifiers will be presented, and then some indirect imbalance related factors that impose additional degrade in performance of the classifiers will be investigated. After that, techniques from the literature to design and operate classifiers under imbalanced data will be analyzed and finally the critical analysis of the approaches in the literature concludes the section.

When classifiers are trained on imbalanced data, their performance becomes biased to correct classification of the majority class. In a 2-class classification problem shown in Figure 1.2, a 2-class classifier that is optimized in terms of accuracy is trained on two cases: the same prior probabilities (Figure 1.2(a)), and imbalanced prior probabilities (Figure 1.2(b)). The optimal Bayes decision threshold is 1 and 0.5 for balanced and imbalanced cases, respectively. In balanced case the misclassification rate of both classes are the same. However, in imbalanced case, the misclassification rate of positive class (the proportion of the number of misclassified positive samples to all positives) is much larger than the negative class (the proportion of the number of misclassified negative samples to all negatives).

Another example is shown in Figure 1.3 where a Support Vector Machine (SVM) classifier is trained on 2D data distributions with different imbalance levels. It is observed that as the

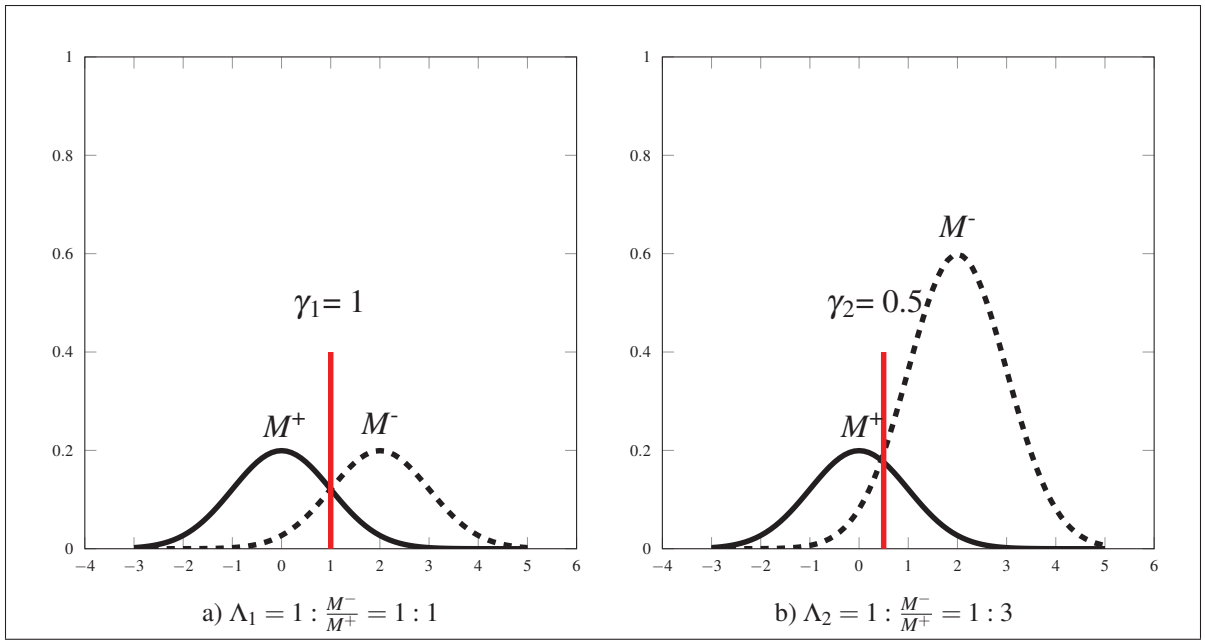


Figure 1.2 Joint probability densities $p(x,C^+)$, $p(x,C^-)$ as the function of a feature value x (x axis) for two classes C^+, C^-, and the optimal Bayes decision boundaries to classify them: balanced vs. imbalanced cases.

imbalance level of training data increases, the decision boundary moves towards the positive class which in turn result in more misclassified positive class samples.

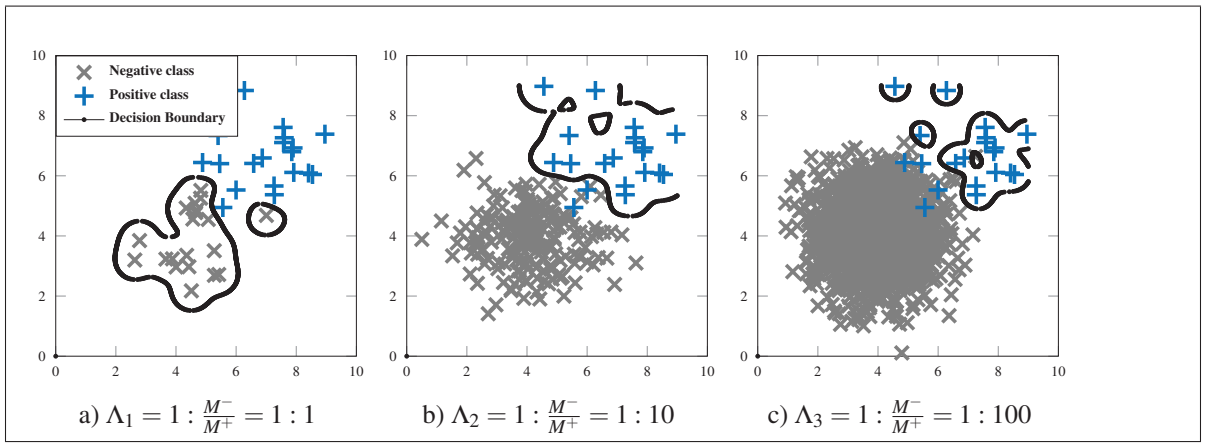


Figure 1.3 Distribution of two 2D classes and the decision boundaries of SVM to classify them: balanced vs. imbalanced cases.

Imbalance between classes hinders classification performance from different perspectives. From design viewpoint, it is a well-established fact that training with imbalanced data result in biased classification performance towards the majority class. On the other hand, the performance of the classification systems may degrade during deployment due to the unpredictable and changing skew level of the input data stream.

In addition to these problems, there are some other imbalance related factors that lead to such degrade in performance. In fact, imbalance and its varying property causes several data intrinsic related drawbacks indirectly such as lack of density, variation in overlap and class separability, dataset shift, within-class imbalance, and small disjuncts. These effects are investigated by López *et al.* (2013) in detail and a summary is presented in the following.

- Lack of density: When the data is imbalanced, the density of the minority class, which is usually the class of interest, is lower than that of the majority class. Therefore, the generated classification model based on these densities is weak (see Figure 1.3(b) and (c)). This effect is highlighted when one-class classifiers are used.

- Overlap and class separability: When the imbalance varies between two distributions of data, the overlap between classes and their separability changes. Figure 1.3(b) and (c) shows examples of two normal distributions of two classes with the same standard deviation. It is observed that with increasing the imbalance, the overlap increases and the separability of the classes decreases.

- Dataset shift: When the imbalance ratio increases the boundary of the classes could change such that it seems that the data set is shifting. If we look back to Figure 1.3(b) and (c) it seems that by increasing the imbalance, the majority class is shifting towards the minority class.

- Within-class imbalance (small disjuncts): Varying imbalance may cause appearance of some small disjuncts in data distribution. This small disjuncts could come from within-class imbalance. In fact within-class imbalance corresponds to the case where a class is

composed of a number of different subclusters and these subclusters do not contain the same number of examples (Japkowicz, 2001). The small disjuncts are solitude subclusters in the feature space. This is the case in user-specific 2-class classification systems for face re-identification. In Figure 1.4 the face captures of 6 individuals are mapped to 2D space. Consider a case when the classifier is designed with ID1 as positive class. By varying imbalance from 1:1 to 1:5 one or more individuals with IDs 2 to 6 are selected as the negative class. Small disjuncts are visible specially in the case of ID2 and ID5. Within-class imbalance is also visible in this application, since the number of samples from each individual is different.

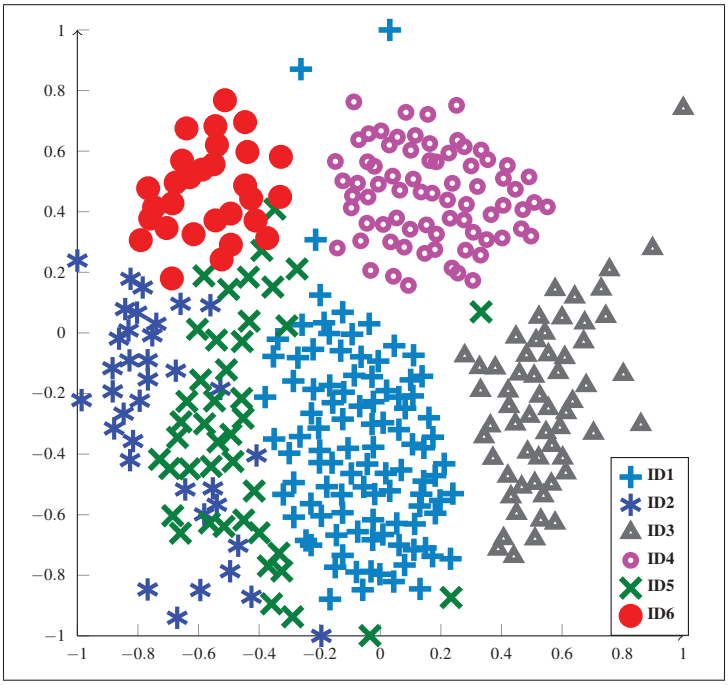


Figure 1.4 2D mapping of face captures of individuals from FIA dataset using Sammon mapping.

Several approaches have been proposed in the literature to design classification systems using imbalanced data. These approaches are divided into data-level and algorithm-level approaches (see Figure 1.5). Data-level approaches either up-sample the target class, under-sample the non-target class or combine up-sampling and under-sampling to re-balance data for the learning

procedure. Algorithm-level methods create or modify learning algorithms to counter the bias towards the non-target class through cost-free techniques or by introducing uneven misclassification costs for the samples from different classes in cost-sensitive approaches. Ensembles of classifiers combine a set of diverse classifiers to design a robust classification system. To handle imbalance, ensembles can exploit one or a combination of aforementioned techniques. In the following, a review of ensemble methods is presented and after that the modified ensembles for imbalance problem are reviewed in Sections 1.2.1.1 and 1.2.1.2.

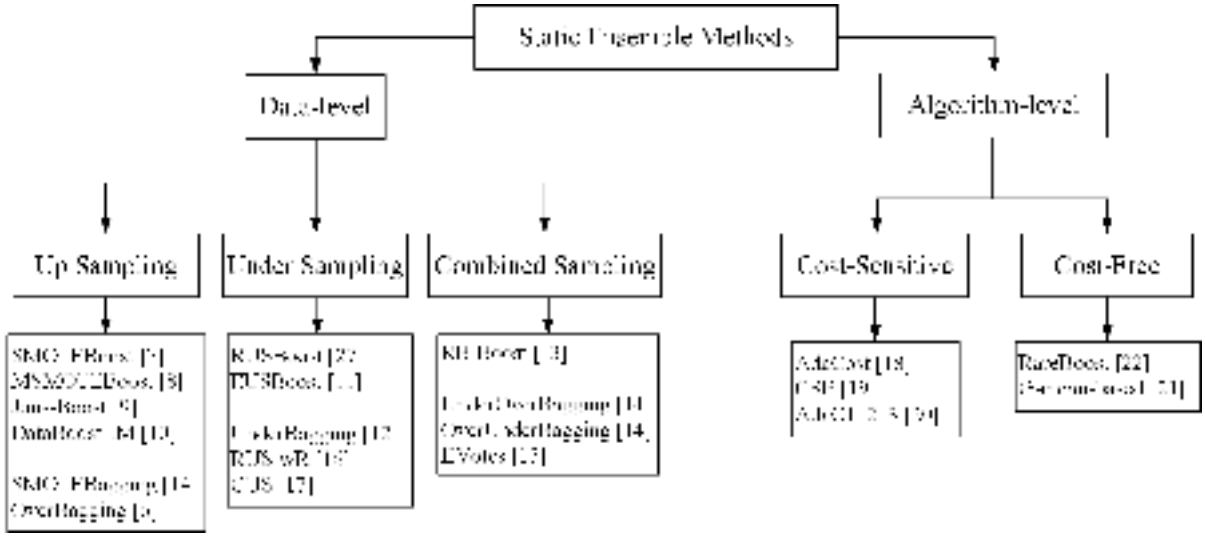


Figure 1.5 A taxonomy of static ensembles learning methods specialized for imbalanced data.

1.2.1 Classifier Ensembles

Ensemble of classifiers improve over the performance of a single classifier by combining a number of classifiers that are diverse and accurate enough to make uncorrelated and accurate predictions. An ensemble of classifiers is created in three stages; generation, selection and combination. In the generation stage a pool of base classifiers is created that are at least more accurate than random guessing and also diverse enough to make different errors. Even though diversity is a key factor in generating and selecting among classifiers, measuring diversity is not straightforward because there is no generally accepted formal definition

(Kuncheva & Whitaker, 2003). Nevertheless, diversity is perceived as the level of disagreement among classifiers in the literature. There are several ensemble generation approaches in the literature to preserve a reasonable level of diversity-accuracy. Diversity could be achieved in several ways. For example, Bagging (Breiman, 1996) and Boosting (Freund, 1995; Freund *et al.*, 1996) algorithms train classifiers on different data subsets. Random sub-space method (RSSM) (Ho, 1995) is another approach that is defined as pseudorandomly drawing subsets of the components of the feature vector to train each base classifier.

The ensemble could be generated from the same base classifiers with different training strategies. For example, in (Connolly *et al.*, 2012) an adaptive classification system (ACS) for video based face recognition is proposed. A pool of fuzzy ARTMAP neural network classifiers is generated. In fact, the authors use the same base classifiers with different hyper-parameters for each classifier in an adaptive manner.

Different types of base classifiers operating in parallel or serially could be used to generate ensemble. For example, in (Zhou & Jiang, 2004) decision tree is integrated with neural network.

The training process of ensemble members may be performed in a single or several iterations. Boosting algorithms (Freund, 1995; Freund *et al.*, 1996) are the most well-known methods that train the base classifiers iteratively.

The relationship between ensemble members can be successive or co-operative. Successive ensembles are created dependently whereas there is an interaction between the learning runs or classification results. The ensembles that are trained iteratively belong to this type of group such as Boosting ensembles. The concept of co-operative relationship between classifiers refers to the ensembles that are created with members acting independently. Bagging and its variations, and the random subspace method belong to this group.

In selection stage of the ensemble members, a subset of the pool is selected and in the combination stage the selected subset is combined to create the ensemble. Two types of ensembles could be defined with respect to their selection and combination strategy based on their de-

pendence on the input data condition. Static ensembles are designed only with training (and validation) data set and are kept the same for any input pattern. Dynamic ensembles, in turn, are adaptive to some criteria or some context extracted from the input data. Such criteria or information from the input data are used for dynamic selection or combination of classifiers in the ensemble (Connolly *et al.*, 2012; Britto Jr *et al.*, 2014).

An important aspect in ensemble methods is to determine how many and which base classifiers should be included in the final ensemble. Ensemble selection has impact on two aspects: efficiency and predictive performance. The reason is that a large ensemble increases the computational cost and too few ensemble members might not be sufficient to increase the accuracy. Some approaches pre-determine the ensemble members by using a controlling parameter such as number of iterations like Bagging and Boosting. Some others try to make the best selection of ensemble members during design, or even testing based on the performance of the classifiers. For example, Prodromidis & Stolfo (2001) suggest ranking classifiers according to their classification performance on a separate validation set and their ability to correctly classify specific classes. Some approaches use greedy search methods and select the ensemble members from a big pool of base classifiers by searching in the space of the different ensembles. The search is guided by either the predictive performance or the diversity of the alternative ensembles. (Ko *et al.*, 2008) propose a dynamic selection of ensembles called K-nearest-oracles (KNORA). For each test sample, KNORA finds its nearest K neighbours in the validation set, figures out which classifiers correctly classify those neighbours in the validation set and uses them as the ensemble for classifying the given pattern in that test set. Skew sensitive Boolean Combination (SSBC) (Radtke *et al.*, 2014; De-la Torre *et al.*, 2015c) select and combine the classifiers based on the estimated imbalance of the input data.

The classification results of classifiers in an ensemble system are combined to make a final single decision for the new unknown data. The combination could be done in two levels, score level or decision level in either static or adaptive manner. With the majority voting method, the final decision is made by counting the votes of ensemble members and selecting the decision with highest number of votes (Kittler *et al.*, 1996). A variant of majority voting

is weighted majority voting that assigns weights to the classifiers based on their performance (Littlestone & Warmuth, 1994; Kuncheva *et al.*, 2001) and is also a decision-level approach that defines a belief function and selects the class that maximizes the belief function based on a basic probability defined for each of the classes. Boosting approaches assign a weight to each ensemble member based on its performance during design. Li *et al.* (2013) assigns the weight of each ensemble member based on the distance between the positive class and the negative class cluster that is used to train that classifier. Dass *et al.* (2005) propose a framework for combining the matching scores from multiple modalities using the likelihood ratio statistic computed using the generalized densities estimated from the genuine and impostor matching scores. Some methods consider the matching scores or decisions as features and train a meta classifier on them. For example, Tao & Veldhuis (2013) concatenate the individual matching scores from different classifiers to construct a new feature vector and classify the feature vector by a Naive Bayes classifier using receiver operating characteristics (ROC). They then obtain Naive likelihood ratio by summing the individual log-likelihood ratios, which can be derived from the slope of the component ROCs. Wolpert (1992) transform the decisions made by the classifiers to a training set and an additional classifier combines the different predictions into a final decision.

Most of the ensemble learning methods to handle imbalance in the literature are static approaches. There are a few adaptive ensembles for imbalanced data classification that allow to adapt the selection and fusion of base classifiers during operations based on the estimated level of skew (Radtke *et al.*, 2014; De-la Torre *et al.*, 2015c). Bagging and Boosting (Freund & Schapire, 1995; Freund *et al.*, 1996) are two common static ensemble methods that have been modified in several ways to learn from imbalanced data in the literature (see the reviews Galar *et al.* (2012); Branco *et al.* (2016); Krawczyk (2016a); Haixiang *et al.* (2016)). Therefore, this section continues by reviewing Bagging and Boosting algorithms in more detail and the effect of imbalance on these algorithms is analyzed. After that the variants of Bagging and Boosting algorithms for imbalance problem are presented.

Bagging:

In the Bagging (see Algo. 1) ensemble generation technique, to train each classifier, a bootstrap of samples $\mathbf{S}'$, containing both target and non-target classes, is drawn randomly with replacement. When data is imbalanced, the non-target samples have higher chance to appear in the bootstraps due to their abundance. Bagging has been modified to handle imbalance as presented in Section 1.2.1.1.

Algorithm 1: Bagging

Input: Training set: $\mathbf{S} = \{(\mathbf{x}_i, y_i); i = 1, ..., M\}$,$y_i \in \{-1, 1\}$
# of iterations: E
Bootstrap size: N_B
Test input : $\mathbf{X}$
Output: Prediction Function: $H(\cdot)$

for $e = 1, .., E$ **do**
  Select N_B samples to create training subset $\mathbf{S}_e^{'}$.
  Train classifier C_e on $\mathbf{S}_e^{'}$.
for $e = 1, .., E$ **do**
  Test C_e classifier on $\mathbf{X}$ and get back $h_e(\cdot)$.
Output the final hypothesis: $H(\cdot) = \sum_{e=1}^{E} h_e(\cdot)$

Boosting:

Boosting algorithms were initiated with AdaBoost (Freund & Schapire, 1995) and improved in AdaBoost.M1 (for 2-class problems) and AdaBoost.M2 (for multiple-class problems) (Freund *et al.*, 1996) to effectively promote a weak learner that performs slightly better than random guessing into a stronger ensemble.

In AdaBoost.M1 (Algo.2) samples are assigned with weights. These weights guide the learning process of a weak classifier to build a stronger ensemble. During Boosting iterations, these weights are used directly or for re-sampling training data, depending on the type of the base classifier being used. When the base classifier is from a type that is not designed to incorporate sample weights in its learning process (like SVMs), training data is re-sampled according to the weights of the samples. This case is considered here to explain the Boosting procedure.

Let's consider a two-class problem with M labelled training samples $\mathbf{S} = \{(\mathbf{x}_i, y_i); i = 1, ..., M\}$ where $y_i \in \{-1, 1\}$ that contains M^+ target samples and M^- non-target samples. All samples in

the dataset are initially associated with the same weight $\mathbf{W_1}(i) = 1/M$, $i = 1, ..., M$. In iteration e, the weights are shown as $\mathbf{W_e}$ and a new training subset is re-sampled into $\mathbf{S}'$ with the weights as $\mathbf{W}'$ to traine classifier C_e. This classifier is tested on all training samples ($\mathbf{S}$) and a loss factor (ε_e) is calculated as the sum of the weights of misclassified samples:

$$\varepsilon_e = \sum_{(i,Y_i):y_i \neq Y_i} \mathbf{W_e}(\mathbf{i}) \tag{1.1}$$

where Y_i is the label associated with $\mathbf{x}_i$ by C_e. If the classifier is too weak ($\varepsilon_e > 0.5$), the classifier is discarded and the training set is re-sampled to train another classifier. The loss factor is then used to define a weight update factor α_e:

$$\alpha_e = \frac{\varepsilon_e}{1 - \varepsilon_e}. \tag{1.2}$$

The weights of the samples are updated as:

$$\mathbf{W}_{\mathrm{e+1}}(i) = \mathbf{W}_{\mathrm{e}}(i)\alpha_e^{\frac{1}{2}|\mathrm{y_i} - \mathrm{Y_i}|}, \tag{1.3}$$

The weight vector is normalized such that the weights of the misclassified samples increase exponentially while the weights of the correctly classified samples decrease. α_e is also used to determine the contribution of the classifier in final predictions (Equation 1.4) so that more accurate classifiers play more important role in identifying the class of the input sample. This process is repeated for a predefined number of times to design E classifiers. Considering $h_e(\mathbf{x})$ as the output of C_e (either a classification score or a label) for an input sample $\mathbf{x}$, final prediction of the ensemble is obtained from:

$$H(\mathbf{x}) = \sum_{e=1}^{E} h_e(\mathbf{x}) \log \frac{1}{\alpha_e} \tag{1.4}$$

AdaBoost is not effective to learn from imbalanced data for two reasons. Non-target samples are the majority and when training data is re-sampled in line 3 of AdaBoost (see Algo. 2), they contribute more in $\mathbf{S}'$. Therefore, C_e is trained biased to correct classification of this

Algorithm 2: AdaBoost.M1 ensemble learning method.

Input: Training set: $\mathbf{S} = \{(\mathbf{x}_i, y_i); i = 1, ..., M\}, y_i \in \{-1, 1\}$
# of iterations: E
Test input : $\mathbf{X}$
Output: Prediction Function: $H(\cdot)$

1 Initialize $\mathbf{W}_1(i) = \frac{1}{M}$ for $i = 1, ..., M$.
2 **for** $e = 1, .., E$ **do**
3 Create new training set $\mathbf{S}'_e$ with weight distribution $\mathbf{W}'_e$.
4 Train classifier C_e on $\mathbf{S}'_e$ with $\mathbf{W}'_e$.
5 Test C_e on $\mathbf{S}$ and get back a label set $\{Y_i, i = 1, ..., M\}$.
6 Calculate the pseudo-loss for $\mathbf{S}$ and $\mathbf{W}_e$: $\varepsilon_e = \sum_{(i,Y_i):y_i \neq Y_i} W_e(i)$.
7 If $\varepsilon_e > 0.5$ **go to** step 3
8 Calculate the weight update parameter: $\alpha_e = \frac{\varepsilon_e}{1-\varepsilon_e}$
9 Update $\mathbf{W}_{e+1}(i) = \mathbf{W}_e(i)\alpha_e^{|y_i - Y_i|/2}$
10 Normalize $\mathbf{W}_{e+1}$ such that: $\sum \mathbf{W}_{e+1} = 1$.
11 **for** $e = 1, .., E$ **do**
12 Test C_e classifier on $\mathbf{X}$ and get back $h_e(\cdot)$.
13 Output the final hypothesis: $H(\cdot) = \sum_{e=1}^{E} h_e(\cdot) \log \frac{1}{\alpha_e}$

class. After that, when C_e is tested on $\mathbf{S}$, loss factor in line 6 is calculated as a weighted error rate of classification. Again, non-target samples contribute more in loss factor calculation and the weight update formula and classifiers contribution in final prediction become biased such that weight of non-target samples increases for the next iteration and classifiers that mostly classify non-target samples correctly get higher importance in final prediction of the ensemble. AdaBoost have been modified to address these two issues through data-level and algorithm-level approaches presented in subsections 1.2.1.1 and 1.2.1.2, respectively.

Another issue related to imbalanced data classification in the literature is that, in this case the most widely used performance metric, classification accuracy, tends to favour the correct classification of the most populated class (or classes) and most of learning algorithms are optimized and compared using unsuitable performance metrics or spaces. Therefore, after presenting the most recent data-level and algorithm-level ensembles from literature in the following subsections, a review of performance metrics and spaces for imbalanced data classification is also presented.

1.2.1.1 Data-Level Ensembles

Class imbalance can be handled in Boosting and Bagging ensembles through upsampling the target class, under-sampling the non-target class or combination of them. A popular up-sampling Boosting approach is SMOTEBoost (Chawla *et al.*, 2003) that integrates Synthetic Minority Over-sampling Technique (SMOTE) into AdaBoost.M2. SMOTE creates synthetic samples by interpolating each target sample with its k-nearest neighbours. MSMOTEBoost (Hu *et al.*, 2009) modify SMOTE algorithm by eliminating noisy samples and oversampling only safe samples. Jous-Boost (Mease *et al.*, 2007) oversamples the target class by duplicating it, instead of creating new samples, and introduces perturbation (jittering) to this data in order to avoid overfitting. DataBoost-IM (Guo & Viktor, 2004) oversample difficult samples from both classes and integrates it into AdaBoost.M2. In OverBagging and SMOTEBagging (Wang & Yao, 2009), Bootstraps are selected after up-sampling the target class.

Up-sampling techniques address the bias of performance in classifiers through balancing class distribution without loss of information. However, up-sampling, in general, increase the number of samples and consequently increase the computation time of learning algorithms, and SMOTE involves additional computations due to interpolating each sample with its k-nearest neighbours to generate synthetic samples.

In under-sampling Boosting category, RUSBoost (Seiffert *et al.*, 2010) integrates random undersampling (RUS) into AdaBoost.M1. RUSBoost is similar to AdaBoost presented in Algo. 2 where in line 3 of this algorithm, $\mathbf{S}'$ contains all target samples and a randomly selected subset of non-target class, often with a size equal to the target class. The subsets of non-target class selected randomly over iterations of RUSBoost could be highly correlated and the classifiers trained on them can lack in diversity, especially when the skew level of training data is high.

The sample selection paradigm in RUSBoost is managed in EUSBoost (Galar *et al.*, 2013b) to create less correlated subsets using evolutionary prototype selection (García & Herrera, 2009). In UnderBagging approach (Barandela *et al.*, 2003), the non-target class is under-sampled with Bagging and the classifiers are trained on balanced subsets of data. Partitioning is also

an UnderBagging approach in which bootstraps are selected without replacement either randomly (Yan *et al.*, 2003), or by clustering (Li *et al.*, 2013).

Some researchers combine SMOTE and RUS in AdaBoost and Bagging ensembles to achieve greater diversity and avoid loss of information as in Random Balance Boosting (RB-Boost)(Díez-Pastor *et al.*, 2015), UnderOverBagging (Wang & Yao, 2009), OverUnderBagging (Wang & Yao, 2009), and IIVotes (Błaszczyński *et al.*, 2010). RB-Boost combines SMOTE and RUS to create training subsets with random and different skew levels in AdaBoost.M1 to increase diversity and robustness to imbalance.

Repetition of sampling in Bagging and Boosting ensembles increase the chance of low correlation between subsets of data that are used for designing classifiers in the ensembles and therefore maintain diversity among them. However, some potentially informative samples may be overlooked from these subsets in under-sampling process.

Partitional approaches (Yan *et al.*, 2003; Li *et al.*, 2013) design classifiers using data subsets sampled from different parts of the feature space and as a result increase diversity of decision boundaries of classifiers. Creating an ensemble from partitions of data can be considered as an UnderBagging approach (Galar *et al.*, 2012) where each bootstrap is drawn from a set of non-target samples that reduces size in each iteration. In other words, after selection of a bootstrap in each iteration, its samples are eliminated from the main set. In random partitioning of non-target samples by Yan et al. (Yan *et al.*, 2003) the non-target data is randomly decomposed into a number of subsets and each subset, combined with the target samples, is used to train a classifier. Li et al. (Li *et al.*, 2013) partition non-target data by clustering it using k-means in the feature space and then create an ensemble from the classifiers trained on each non-target cluster and the target samples. The contribution of the classifiers in the ensemble are then weighted based on the distance between the corresponding non-target cluster and target class.

In contrast to RUSBoost, these partitional approaches use all non-target samples from partitions to design ensembles and avoid loss of information. However, not all samples are informative and using all samples for training may result in unnecessary time and memory usage.

Therefore, enhancing partitional methods with more intelligent sample selection and ensemble learning algorithm (like RUSBoost) can avoid information loss and excessive time usage at the same time. This is the motivation of our proposed algorithms in chapters 3 and 4.

1.2.1.2 Algorithm-Level Ensembles

Using the standard loss factor based on misclassification rate in Boosting ensemble learning algorithms biases their performance towards negative class. In the literature this issue is avoided at the algorithm level using two types of techniques; those that employ two different misclassification cost factors, one for positive and another for negative classes and those that handle this issue without the use of cost factors.

Cost-sensitive Boosting methods including AdaCost (Fan *et al.*, 1999), CSB (Ting, 2000) and AdaC (Sun *et al.*, 2007), embed different misclassification cost factors into loss function or weight update formula of AdaBoost.M2.

For example, CSB (Ting, 2000) introduce two different cost factors for positive and negative classes as $C_{\mathrm{FN}} = 1$ and $C_{\mathrm{FP}} \leq 1$, respectively.

$$\mathbf{W}_{\mathrm{e+1}}(i) = \begin{cases} \mathbf{W}_{\mathrm{e}}(i) C_{\mathrm{FN}} \exp\{-\alpha_e |\mathrm{y_i} - \mathrm{Y_i}|/2\} & \text{for } Y_i = 1 \\ \mathbf{W}_{\mathrm{e}}(i) C_{\mathrm{FP}} exp(-\alpha_e |\mathrm{y_i} - \mathrm{Y_i}|/2\} & \text{for } Y_i = -1 \end{cases} \tag{1.5}$$

In these cost-sensitive approaches by setting the cost of misclassifying the positive class C_{FN} greater than the cost of misclassifying the negative class C_{FP} ,the weights of misclassified samples from positive class increase more than that of the misclassified samples from negative class. In addition, the weights of the classifiers that correctly classify positive class better than the negative class is higher in final decision. Therefore, these cost-sensitive approaches can make up for the usage of standard error rate in Boosting ensembles and allow adapting the performance by selecting proper cost factors based on the application. The drawback of these cost-sensitive approaches is that they require known C_{FN} and C_{FP} that are usually set ad-hoc or by conducting a search in the space of possible costs for a dataset.

Some cost-free approaches have been proposed to deal with the bias of performance caused by using standard error in Boosting ensembles. In RareBoost (Joshi *et al.*, 2001), two different weight update factors α (see Eq. 1.2) are defined for positive and negative classes as:

$$\alpha_e^+ = \frac{1}{2} ln(\frac{TP_e}{FP_e}), \alpha_e^- = \frac{1}{2} ln(\frac{TN_e}{FN_e}) \tag{1.6}$$

where TP_e and TN_e are the true positive and true negative counts, respectively. Then the weight update formula and final classification prediction are modified as:

$$\mathbf{W}_{e+1}(i) = \begin{cases} \mathbf{W}_e(i) \exp\{-\alpha_e^+ |y_i - Y_i|/2\} & \text{for } Y_i = 1 \\ \mathbf{W}_e(i) \exp\{-\alpha_e^- |y_i - Y_i|/2\} & \text{for } Y_i = -1 \end{cases} \tag{1.7}$$

$$H(\mathbf{x}) = sign(\sum_{e:h_e(\mathbf{x}) \geq 0} \alpha_e^+ h_e(\mathbf{x}) + \sum_{e:h_e(\mathbf{x}) < 0} \alpha_e^- h_e(\mathbf{x}))) \tag{1.8}$$

(Kim *et al.*, 2015) also define two different α_es for positive and negative classes as:

$$\alpha_e^+ = \frac{1 - l^+}{l^+}, l^+ = \frac{\sum_{i;y_i=+1} W_e(i)|y_i - Y_i|/2}{\sum_{i;y_i=+1} W_e(i)} \tag{1.9}$$

$$\alpha_e^- = \frac{1 - l^-}{l^-}, l^- = \frac{\sum_{i;y_i=-1} W_e(i)|y_i - Y_i|/2}{\sum_{i;y_i=-1} W_e(i)} \tag{1.10}$$

where l^+ and l^- are pseudo errors of classifier in classifying each class. Finally:

$$\alpha_e = \ln(\sqrt{\mu_i \alpha_e^+ \alpha_e^-}) \, , \tag{1.11}$$

μ_i is a multiplier to control the weight of each sample. There is an issue with the loss factor proposed by Kim *et al.* (2015). If there are no misclassified samples in one class or in both classes (i.e. l^+ and/or $l^- = 0$), α_e becomes undefined.

Cost-free methods enhance the performance of Boosting ensembles without setting any cost factors and guide the learning process using a more suitable loss factor calculation since the use

of weighted standard accuracy, as in original Boosting algorithm, biases the learning process towards correct classification of the negative class. This motivates us to use a suitable loss factor in Boosting ensemble that is presented in chapter 4. For that, a literature of performance evaluation metrics and spaces for class imbalance is presented in section 1.2.3.

1.2.2 State of the Art Deep Learning Methods

Analogously to the traditional classification systems, class imbalance biases the performance of the neural network architectures. The reason is that the gradient component of the majority (negative) class dominates that of the minority (positive) class which affects the weight update process of the back-propagation algorithm (Anand *et al.*, 1993).

Several traditional data-level and algorithm-level methods are applicable in deep learning and there are specialized algorithms that exploit neural network feature learning abilities. A review of the methods in each family is given in the following subsections. The surveys by Buda *et al.* (2018) and Johnson & Khoshgoftaar (2019) compare some of the methods reviewed in this section experimentally and Johnson & Khoshgoftaar (2019) present a discussion that highlights various gaps in deep learning from class imbalanced.

1.2.2.1 Data-Level Methods

Jeatrakul *et al.* (2010) combine SMOTE and Complementary Neural Network (CMTNN). CMTNN (Kraipeerapun *et al.*, 2009) is a technique using a pair of complementary feedforward backpropagation neural networks called Truth Neural Network (Truth NN) and Falsity Neural Network (Falsity NN). Truth NN is a neural network that is trained to predict the degree of the truth memberships, the Falsity NN is trained to predict the degree of false memberships. Falsity NN uses the complement outputs of the Truth NN to train the network. Truth NN and Falsity NN are employed to detect and remove misclassification patterns from a training set.

Masko & Hensman (2015) randomly over sample the positive class and Ahmed *et al.* (2015) randomly undersample the negative class for training deep neural network architectures.

Lee *et al.* (2016) use RUS to rebalance data along with data augmentation (oversampling the positive class) to improve the performance of a deep CNN architecture under imbalance.

In the method proposed by Huang *et al.* (2016), data is clustered and the euclidean distance is optimized in a deep neural network such that the inter-class distance is minimized and intra-class distance is maximized. To this aim, the samples that are most representative of each class, and the samples that lie near the border of each class are preserved and the remaining samples are ignored.

Lee *et al.* (2016) use RUS to train a deep CNN architecture. Then fine-tune it using the whole imbalanced data. This method does not suffer from information loss in contrast to plain RUS, which completely removes potentially useful information from the training set.

A dynamic sampling method is proposed by Pouyanfar *et al.* (2018) to adapt the sampling rate during training according to class-wise performance. The misclassified classes are over-sampled and the correctly classified classes that are under-sampled. The sample size of each class is selected dynamically based on the F-measure after each iteration.

1.2.2.2 Algorithm-Level Methods

In learning process of the conventional deep neural network architectures, the loss function poorly captures the errors from the positive class in cases of high class imbalance. This issue can be addressed by either introducina new loss functions or using cost-sensitive learning methods.

Zhang *et al.* (2016) propose cost-sensitive deep belief network with differential evolution (CSDBN-DE) to automatically learn costs for the cost-sensitive deep learning using a differential evolutionary algorithm. The network is first pre-trained regularly and then the cost matrix is incorporated into the output layer's softmax probabilities during the fine-tuning phase.

Wang *et al.* (2016) propose two new loss functions that are more sensitive to the errors from the minority class for training the deep neural networks on imbalanced data sets. This makes the

learning algorithms more sensitive to better capture errors from the positive class and therefore achieves higher overall classification accuracy under imbalance.

Khan *et al.* (2017) propose cost-sensitive versions of three widely used loss functions in deep neural networks meaning minimum squared error loss, SVM hing loss and cross entropy. They propose an algorithm for jointly optimization of the network parameters and the class-sensitive costs during training.

Yang *et al.* (2017) propose learning ensemble of deep neural networks using AdaBoost algorithm to increase accuracy and robustness to imbalance.

Lin *et al.* (2017) presented the focal loss (FL) which is a modification of the cross entropy (CE) loss. CE is multiplied by a factor that decreases for the easily classified samples and increases for the positive class.

Wang *et al.* (2018) propose a cost-sensitive deep neural network (CSDNN) method by incorporating the cost matrix into the CE loss. The limitation of this method is identifying an ideal cost matrix which often require a domain expert or a grid search procedure.

Zhang *et al.* (2018) propose category centers (CC), that combines transfer learning, deep CNN feature extraction, and a nearest neighbour discriminator. They use the fact that the same class tend to cluster well in CNN deep feature space. However, the decision boundary that is created by the final layer of the CNN tends to be biased towards the majority class when there is high level of imbalance between classes. To avoid this bias of the boundary they propose using high-level features extracted by the CNN to calculate each class's centroid in deep feature space. The new images are then assigned to their nearest deep feature category center. This method is highly dependent on the deep neural network's ability to generate discriminative features that cluster well.

Finally, (Guo & Zhang, 2017) add a loss term to the cross entropy loss of the Softmax to improve the feature extraction performance under imbalance, especially for the persons with very limited training samples.

1.2.3 Performance Evaluation Metrics and Spaces for Class Imbalance

Imbalanced data distributions occur in many real-life applications (Krawczyk, 2016a), often in two-class problems. In these applications, correctly recognizing samples of the positive class is the main requirement. Avoiding excessive misclassification of negative samples can also be more or less important, depending on the application at hand.

In some applications, the above requirements can be expressed in terms of misclassification costs, where a higher cost is assigned to the misclassification of a positive sample. This also allows one to "indirectly" take into account class imbalance. Similarly, in other applications assigning different "fictitious" costs to misclassifications of positive and negative instances can be a (indirect) way to take class imbalance into account. For example, in a medical application like automatic cancer diagnosis the number of positive samples (patients who have cancer) is often much less than negative samples (patients who do not have cancer). In this application, misclassifying a positive sample as a negative i.e., wrongly discharge of a patient who actually has cancer results in a delayed treatment, cost is usually much higher than the one of misclassifying a negative sample, i.e., wrongly suspecting a patient with cancer; the reason is that in the latter case the diagnosis can be corrected in the follow-up tests (Artan *et al.*, 2010). Another example is online video surveillance applications, where the objective is to find images of a suspected individual (positive samples) in a public place over a network of cameras. In this application the misclassification of negative samples is tolerable to some extent, since it would be corrected later by a human operator. However, beyond a certain limit, this could waste too much time of the operator, up to missing the person of interest. In this case, a small number of misclassified positive samples could result in saving a relatively higher misclassifications of negative samples. In such cases, the application requirements cannot be expressed in terms of misclassification costs.

Several performance metrics have been used so far for applications with imbalanced classes, and specific metrics have also been proposed that can be divided into two categories; (1) scalar metrics that are defined in a single operating condition (e.g. decision threshold) of the classifier.

(2) global performance evaluation spaces that consider the performance of the classifier under a range of operating conditions. Some reviews of these metrics can be found in the work by Ferri *et al.* (2009), where the behaviour of some scalar performance metrics are analyzed experimentally for several problems including imbalance to find the correlation between these metrics. Garcıa *et al.* (2010) also compare the scalar metrics for imbalance problem. Landgrebe *et al.* (2006) and Davis & Goadrich (2006) compare ROC and PR spaces and analyze the relationship between them. ROC, PR and cost spaces are compared in a survey by Prati *et al.* (2011). In this section, we focus on reviewing these metrics in terms of their sensitivity to imbalance, specifically global spaces that consider different operating conditions and different preference weights.

1.2.3.1 Scalar performance metrics

We focus here on two-class problems, although some of the existing metrics can also be applied to multi-class problems. We denote the prior probability of the positive and negative class with $P(+)$ and $P(-)$, respectively, and the measure of class skew with $\lambda = {}^{P(-)}/_{P(+)}$ (assuming that the positive class is the minority one, we have $\lambda > 1$).

The performance of a two-class classifier on a given set of labelled samples (e.g., a testing set) can be summarized by its confusion matrix. For a given data set (i.e., a testing set), let us denote with M^+ and M^- the number of positive and negative samples, and with N^+ and N^- the number of samples labelled as positive and negative by the classifier at hand. The confusion matrix reports the number of correctly and wrongly classified samples from both classes: true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN). The true positive and false negative rates, denoted respectively as TPR and FNR, are defined as TP/M^+ and FN/M^+; analogously, the true negative and false positive rates (TNR and FPR) are defined as TN/M^- and FP/M^-. These rates are sample-based estimates of the corresponding probabilities (e.g., TPR estimates the probability of correctly classifying positive samples). In particular, in image or document retrieval applications Precision (Pr) and Recall (Re) metrics are used: Re corresponds to TPR, whereas Pr is defined as $TP/(TP+FP)$ or TP/N^+, i.e.,

the fraction of correctly classified samples among the ones labelled as positive. This metric estimates the probability that a sample labelled as positive is actually positive.

From the above classification rates, several scalar metrics can be defined. The widely used, standard accuracy is defined as the probability of correctly classifying a random sample, which can be estimated as $A = (TP+TN)/(M^+ + M^-)$. However, in case of imbalanced data, accuracy is biased towards the correct classification of the negative class due to the abundance of this class. In particular, the accuracy of the trivial classifier that labels all samples as negatives tends to one as level of imbalance increases. This makes it unsuitable when the correct classification of the positive class is more important, which is often the case in such kind of problem.

A generalization of accuracy (more precisely, of the error probability, defined as $1-A$) is the expected cost (EC), which is used in applications where a cost (in a suitable, application-dependent unit of measurement) can be associated to the classification outcome (either correct or incorrect) of a sample. In the simplest case, the cost of correct classifications is zero, whereas two misclassification costs are associated to the positive and negative classes, denoted respectively as C_{FN} and C_{FP}. EC is then defined as the expected classification cost of a random sample, which amounts to:

$$EC = \mathrm{FNR} \cdot P(+) \cdot C_{\mathrm{FN}} + \mathrm{FPR} \cdot P(-) \cdot C_{\mathrm{FP}} \tag{1.12}$$

When data is imbalanced, misclassifying a positive sample is usually more costly than misclassifying a negative one, i.e., $C_{\mathrm{FN}} > C_{\mathrm{FP}}$ (see, e.g., the above example of medical diagnosis). This avoids the bias of classification accuracy toward the negative class. Accordingly, EC can be used also in applications with imbalanced classes where misclassification costs are not precisely known, or even difficult to define: one can design a classifier focused on correctly recognizing the positive class by suitably defining C_{FN} and C_{FP}, and using EC as the objective function to be minimized. On the other hand, as the ratio $C_{\mathrm{FN}}/C_{\mathrm{FP}}$ increases, minimizing EC

allows to increase TPR at the expense of increasing FPR, which may be undesirable in some applications.

In information retrieval applications, misclassifications are usually not associated to a cost, and thus EC is not used. To evaluate the effectiveness of a retrieval system, Pr and Re are the most widely used metrics, instead. Note that they measure complementary aspects of retrieval effectiveness: for a given retrieval system, Pr can usually be increased only at the expense of a lower Re, and vice versa. For instance, when a two-class classifier is used to label input samples (e.g., images or documents) as relevant or non-relevant to a given query, changing its decision threshold results in increasing one of the two metrics and in decreasing the other.

In particular, in the case of class imbalance, Pr takes into account both TP and FP, and drops severely when correct classification of positive class is attained at the expense of a high number of misclassified negative samples. This can be seen more clearly by rewriting Pr as follows:

$$Pr = \frac{TPR}{TPR + \lambda FPR} \,. \tag{1.13}$$

It is now evident that, as the λ increases, any given increase in FPR results in a higher reduction in Pr. This is an interesting feature compared to EC mentioned above for class imbalance problems.

To obtain a single scalar metric, the F-measure has been proposed by Van Rijsbergen (1979). It is defined as the weighted harmonic mean of Pr and Re:

$$F_\alpha = \frac{1}{\alpha \frac{1}{\mathrm{Pr}} + (1-\alpha)\frac{1}{\mathrm{Re}}} \,, \tag{1.14}$$

where $0 < \alpha < 1$ is the weight parameter. Note that another form of the F-measure is more commonly used: it is obtained by rewriting the weight as $\alpha = (1+\beta^2)^{-1}$, with $\beta \in [0, +\infty)$, which leads to

$$F_\beta \quad = \quad \frac{(1+\beta^2)\mathrm{Pr} \cdot \mathrm{Re}}{\beta^2 \mathrm{Pr} + \mathrm{Re}} \tag{1.15}$$

$$= \frac{(1+\beta^2)\text{TP}}{(1+\beta^2)\text{TP}+\text{FP}+\beta^2\text{FN}} . \tag{1.16}$$

It is easy to see that for $\alpha \to 0$, $F_\alpha \to$ Re, whereas for $\alpha \to 1$, $F_\alpha \to$ Pr; setting $\alpha = 1/2$ one gets the unweighted harmonic mean of Pr and Re, i.e., both metrics are equally weighted. One interesting feature of the F-measure is that its sensitivity to the correct classification of the positive and the negative classes can be adjusted by tuning α. This measure can be better than the expected cost to compare classifiers when data is imbalanced, especially in information retrieval, since it weighs the relative importance of TPR and Pr rather than TPR and FPR directly as in obtaining expected cost. This inspires us to use this metric to optimize ensembles of classifiers in terms of F-measure in chapters 3, 4 and 5.

In general, note that each of the metrics TPR, FPR, TNR and FNR, that are directly extracted from the confusion matrix, focuses on the performance of each class individually, and does not allow to evaluate the effect of class imbalance. However, any metric that uses values from both rows of this matrix, like Pr, will be inherently sensitive to imbalance (He & Garcia, 2009).

We finally mention other existing scalar metrics that have been exploited, and new ones that have been specifically proposed, for class imbalance problems, although they are currently less used than EC and the F-measure (some of them include parameters that can be tuned to weigh the correct classification of both classes). A more extensive review of these metrics can be found in (Ferri *et al.*, 2009; Garcıa *et al.*, 2010): Matthews correlation coefficient (Matthews, 1975), defined as the correlation between the true and the predicted label of a random sample; the geometrical mean (Kubat *et al.*, 1998) either between TNR and Re, or between Pr and Re (the former values the correct classification of both classes equally, whereas the latter is more sensitive to imbalance due to the use of Pr); the arithmetic average of TPR and TNR (macro-averaged accuracy) (Ferri *et al.*, 2009), which values the correct classification of both classes equally; its variant mean-class-weighted accuracy (Cohen *et al.*, 2006), that includes a weight to increase the relative importance of the two classes; optimized precision (Ranawana & Palade, 2006), that combines Re and TNR and accounts for the relative

number of positive and negative samples; the adjusted geometric mean (Batuwita & Palade, 2009) between Re and TNR, that aims at increasing Re while keeping the reduction of TNR to a minimum; and the index of balanced accuracy (Garcıa *et al.*, 2010), aimed at reducing the effect of the difference in TPR and TNR in metrics that combine them. Variants of the above performance metrics have also been proposed to account for $P(+)$ (Garcıa *et al.*, 2010; Flach & Kull, 2015).

1.2.3.2 Global Evaluation Curves

In many real-world applications, there is no precise knowledge of the operating condition where the classification system will be deployed, i.e., the misclassification costs (when they can be applied) or the relative importance of Pr and Re (the weight α of the F-measure), and the class priors. In this case, it is desirable that the classifier performs well over a wide range of operating conditions, and it is thus useful to evaluate its performance over such a range.

Global curves depict the trade-offs between different evaluation metrics in a multidimensional space under different operating conditions, rather than reducing these aspects to a single scalar measure which gives an incomplete picture of prediction performance. However, global methods are not as easy to interpret and analyze as single scalar values and pose some difficulties in conducting experiments where many data sets are used.

A well known and widely used tool for two-class problems is the Receiver-Operating Characteristic (ROC) curve, which plots TPR vs FPR for a classifier (typically as a function of its decision threshold), allowing to evaluate the trade-off between these measures for different operating conditions, as well as to compare different classifiers. Each classifier with a specific threshold corresponds to a point in the ROC space, and a potentially optimal classifier lies on the convex hull of the available set of points, regardless of operating conditions (misclassification costs and class skew). The performance of a classifier in ROC space can be summarized by a scalar metric defined as the area under the ROC curve (AUC), which equals the probability that a random positive sample is given a higher score than a random negative sample.

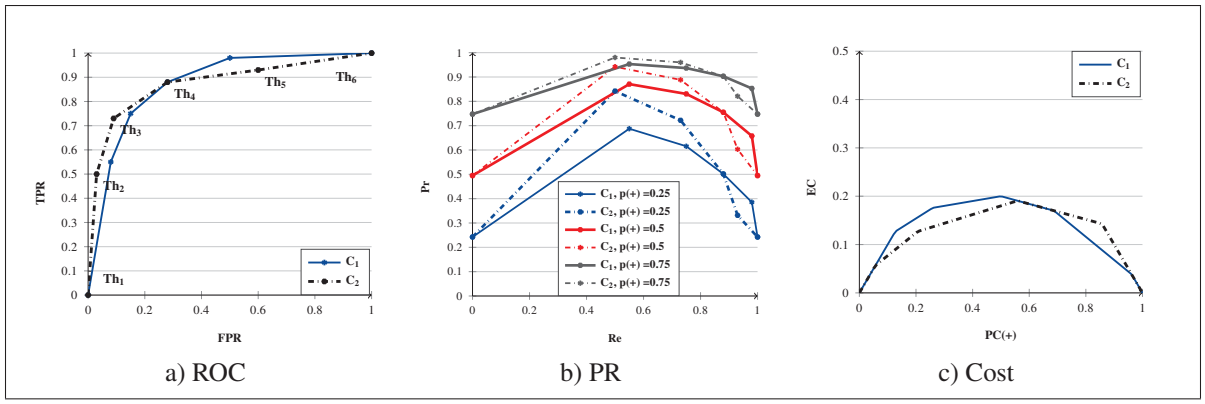
In problems with class imbalance, a drawback of the ROC space is that it does not reflect the impact of imbalance since for a given classifier, the TPR and FPR values do not depend on class priors (Fawcett, 2006) (see Figure 1.6(a)). However, it is possible to estimate the expected performance of the classifier in ROC space for a given skew level of data in terms of EC. In ROC space, each operating condition corresponds to a set of "isoperformance" lines that have the same slope. An optimal classifier for a specific operating condition is found by intersecting the ROC convex hull (ROCCH) with the upper left isoperformance line.

To make this process easier, Cost Curves (CC) (Drummond & Holte, 2006) have been proposed to better visualize the performance of the classifiers over a range of misclassification costs and/or skew levels in terms of EC (see Figure 1.6(c)). This space is further investigated in section 1.2.3.3.

In contrast to TPR and FPR, precision (Pr), is sensitive to class imbalance. When Pr and Re, the well-known metrics in information retrieval, are used as the performance metrics, their trade-off across different choices of the classifier's decision threshold can be evaluated by the precision-recall (PR) curve, which is obtained by plotting Pr as a function of Re. Contrary to the ROC curve, the PR curve is sensitive to class imbalance, given its dependence on Pr. However, for different operating conditions (skew levels), different curves are obtained in this space, which makes it difficult to compare the classifiers when a range of operating conditions are considered (see Figure 1.6(b)). A disadvantage with respect to the ROC space is that the convex hull of a set of points in PR space, and the area under the PR curve, have no clear meaning, despite they are used by several authors Flach & Kull (2015).

In PR space, a given Pr and Re pair, under a specific operating condition (skew level and desired preference of Re to Pr), can be summarized into a single value metric; F-measure (similarly to EC in ROC space). F-measure isometrics (sets of points that correspond to the same value of F-measure) in PR space are hyperbolic (Hanczar & Nadif, 2013; Flach & Kull, 2015). In fact, it is not easy to visualize the performance of a classifier in terms of F-measure over a range of decision thresholds, skew level, and preference of Pr to Re at the same time

in PR space. This is analogous to difficulty of visualizing the performance in terms of the expected cost in ROC space. In the case of the expected cost, this problem has been addressed by proposing the cost (and Brier) curves visualization tools, alternative to the ROC space, described in Section 1.2.3.3. Inspired by them, we will propose in Chapter 5 an analogous visualization tool for the F-measure.



a) ROC b) PR c) Cost

Figure 1.6 Performance of two soft classifiers in ROC, PR and Cost spaces.

1.2.3.3 Expected Costs Visualization Tools

There are at least three or four distinct sets of samples during the design and deployment of a classification system that each might have a different proportion of positive class samples. $P_{train}(+)$ is the proportion of positive samples in the dataset used to learn the classification system. Depending on the type and the structure of the classification system, a validation stage may or may not exist. $P_{validation}(+)$ is the proportion of positive samples in the dataset used to evaluate and tune the parameters of the classification system during the validation stage. After the classification system is designed, it is tested with another set of data of which $P_{test}(+)$ is the proportion of positive samples. $P_{deploy}(+)$ is the proportion of the positive samples during deployment. The ROC curve plots TPR versus FPR computed from the class-conditional probabilities, which are assumed to remain constant for different $P_{deploy}(+)$. However, because we do not necessarily know $P_{deploy}(+)$ at the time we are learning or evaluating the classifier we would like to visualize the classifier's performance across all possible values of $P_{deploy}(+)$.

Cost curves (Drummond & Holte, 2006) do precisely that and estimates the classifier performance in terms of EC across all possible values of $P(+)$. Therefore, it is $P_{deploy}(+)$ that should be used for $P(+)$ because it is the classifier performance during deployment that we wish to estimate.

With $P(-) = 1 - P(+)$, and FNR $= 1 -$ TPR, EC (Eq. (1.12)) is normalized to take the maximum value of 1 as:

$$PC(+) = \frac{P(+) \cdot C_{\text{FN}}}{P(+) \cdot C_{\text{FN}} + (1 - P(+))C_{\text{FP}}} \tag{1.17}$$

$$NEC = (1 - \text{TPR} - \text{FPR})PC(+) + \text{FPR} \tag{1.18}$$

Cost curve depicts the normalized expected cost NEC (Eq. (1.18)) versus $PC(+)$

$$NEC = \begin{cases} \text{FPR} & \text{if } PC(+) = 0 \\ 1 - \text{TPR} & \text{if } PC(+) = 1 \end{cases} \tag{1.19}$$

The always positive and always negative classifiers are shown with two lines in the cost space connecting (1,0) to (0,1), and (0,0) to (1,1), respectively. The operating range of a classifier is the set of operating points, for which the classifier dominates both these lines (Drummond & Holte, 2006). Note that in the rest of the thesis, we use EC instead of NEC.

There is a point-line duality between cost curves and ROC space. A point in ROC space is represented by a line in the cost space and a point in cost space is represented by a line in the ROC space. The lower envelope of cost lines in cost space shapes the CC corresponding to the convex hall of all pairs of (TPR and FPR) points in the ROC space (ROCCH). There are some advantages in cost space to visualize the performance of the classifiers compared to ROC space. For reading quantitative performance information from an ROC plot for specific operating conditions, one deals with some geometric constructions using the iso-performance lines and it is difficult to analyze them when inspecting ROC curves with naked eye (Drummond & Holte, 2006). In contrast, the classifiers can be analyzed easily by a quick visual inspection for given

operating conditions. This property of cost space helps the user to easily compare the classifiers to the trivial classifiers, to select between them, or to measure the difference in performance between classifiers for the given operating conditions (Drummond & Holte, 2006).

Brier curves (BC) Ferri *et al.* (2011) visualize classifier performance assuming that the classifier scores are estimates of the posterior class probabilities, without requiring optimal decision threshold for a given operating condition. Similarly to the cost space BC plots classification error versus C_{FN}, C_{FP} and $P(+)$ assuming a fixed decision threshold equal to cost proportion $c = C_{\text{FN}}/(C_{\text{FN}} + C_{\text{FP}})$ or skew that is defined as $z = cP(+)/(cP(+) + (1-c)(1-P(+)))$.

Having in mind the advantages of CC and BC to visualize the expected cost of the classifiers, no performance visualization tools analogous to CC and BC exist for the F-measure, and investigating such space is the subject of Chapter 5 which, to our knowledge, has not been addressed in the literature. There is a first step by (Flach, 2014) "towards linking threshold choice methods and performance metrics for loss functions based on F-measure." derived as F-cost curves. F-cost curves plot a non-linear transformation of the F-measure, defined as $2(1-F_\alpha)/F_\alpha$, as a function of a parameter that corresponds to $(1-\alpha)$. In this space, the curves which correspond to a single TPR, FPR point are straight lines as in CC. So F-cost curves are visually similar to CC. However, this does not allow one to see the behavior of the F-measure as a function of class skew, which is the (different) goal of the proposed F-measure space in Chapter 5.

CHAPTER 2

EXPERIMENTAL METHODOLOGY

In this chapter, the experimental methodology used throughout this thesis is presented. We start this chapter by introducing the datasets and then continue with the experimental protocol. Then the performance evaluation used in our experiments are presented. For the experiments in this thesis, we use SVM with RBF kernel (Chang & Lin, 2011) as the base classifier where $K(x',x'') = \exp\{\frac{\|x'-x''\|^2}{2\kappa^2}\}$. The kernel parameter κ is set as the average of the mean minimum distance between any two training samples and the scatter radius of the training samples in the input space (Li *et al.*, 2008). The scatter radius is calculated by selecting the maximum distance between the training samples and a point corresponding to the mean of training samples. We used the LibSVM implementation of (Chang & Lin, 2011).

2.1 Datasets

As explained in Section 1.2 , one of the main finding of López *et al.* (2013) is that variations in imbalance affects overlap and class separability and imbalance becomes an issue specially when there is overlap between classes. In addition, different classification systems may perform differently when they are trained and tested on different imbalance levels of data. Therefore, in this Ph.D. project, we generate synthetic datasets to control and vary these factors and evaluate the proposed and baseline classification systems under different conditions. The proposed and baseline classification systems are also evaluated on video datasets for face re-identification application.

2.1.1 Synthetic Dataset

2D synthetic datasets used in experiments of Chapters 3 and 4 are generated to emulate a nonlinear classification problem similar to face re-identification classification problem. The data is generated to emulate both binarization of a multi-class classification problem when the classification strategy is one versus all and binary classification problems where there is no

prior knowledge of optimal sub-clusters of data. The samples of both positive and negative classes are generated from a mixture of Gaussian distributions. The samples from one normal distribution are considered as positive class and all other samples are considered as negative class.

To generate the 2D synthetic data, $M^+ = 100$ positive class samples are generated with a normal distribution as $N(m_+, \sigma_+)$, where $m_+ = (0,0)$ and $\sigma_+ = \left[\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right]$ indicate its mean and covariance matrix, respectively. Then, $n_T^- = 100$ points are generated randomly from a uniform distribution around m_+. These points ($m_{-,j},\ j = 1,...,n_T^-$) are generated as the mean of n_T^- Normal distributions ($N(m_{-,j}, \sigma_-),\ j = 1,\ldots,n_T^-$) for negative class where $\sigma_- = \sigma_+$. Each normal distribution contains $M^+ = 100$ samples and is considered as an ideal sub-cluster of negative class. The mean of these clusters ($m_{-,j}$s) keep a margin distance δ from m_+. This margin δ is used to control the level of overlap between positive and negative classes.

For the experiments in Chapters 3 and 4, we selected the parameter δ as 0.1 (maximum overlap) and 0.2 (medium overlap). For each overlap level, each normal distribution is randomly divided into two subsets for design and testing. Then the design subsets are divided into 5 folds considering one fold for validation and 4 folds for training. Five replications are carried out by alternating the validation fold in each iteration and by reversing the role of design and testing subsets for a total of 10 replications.

Two skew levels and two overlap levels of training data have been considered for the experiments, which have been combined into the three settings shown in Table 2.1. $\Lambda_{\text{train}} = 1 : \lambda$, $\lambda = {}^{M^-}/_{M^+}$ is set to 1:50 in two setting and to 1:20 in the other (or $P_{train}(+) = \{0.04, 0.02\}$ in terms of prior probability).

In a similar way, four imbalance levels ($\Lambda_{\text{test}} = \{1:1, 1:20, 1:50, 1:100\}$) are considered in Chapters 3 and 4 for testing to evaluate the robustness of the classification algorithms over varying skew levels of data during operation.

Examples of synthetic training and test data corresponding to different settings are presented in Figures 2.1 and 2.2.

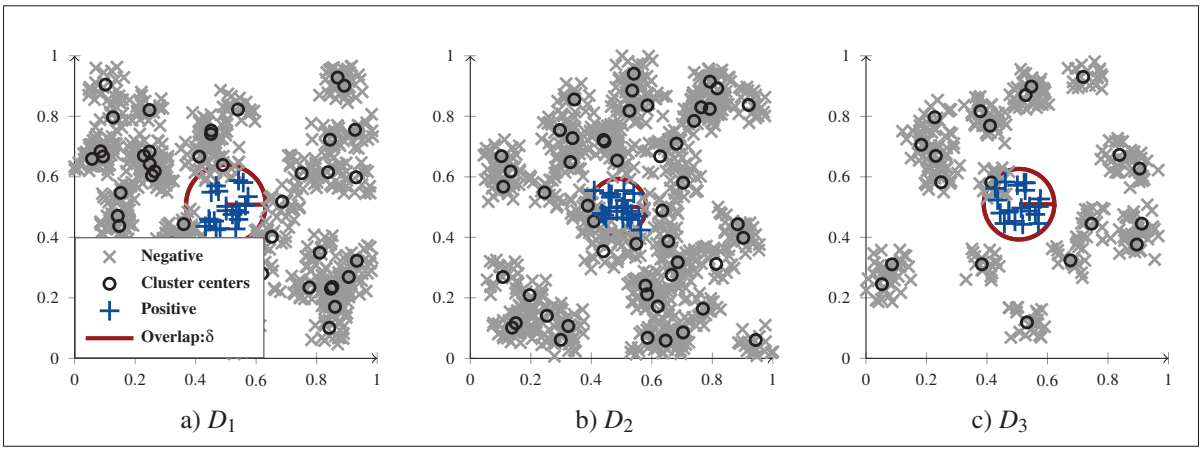


Figure 2.1 Examples of synthetic training data generated under different settings D_1, D_2 and D_3 used in the experiments of Chapters 3 and 4.

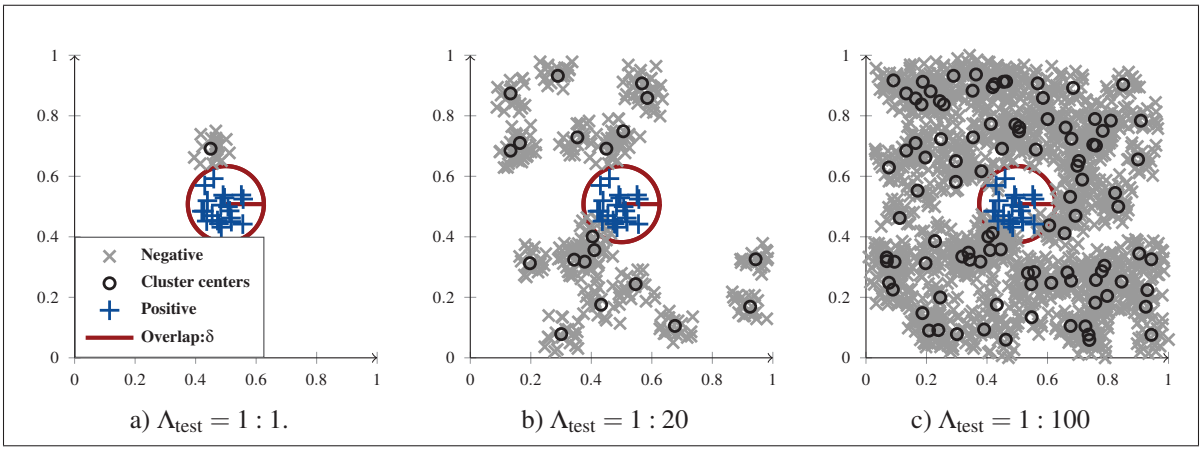


Figure 2.2 Examples of synthetic test data generated with $\delta = 0.2$ and different skew levels λ_{test} used in the experiments of Chapters 3 and 4.

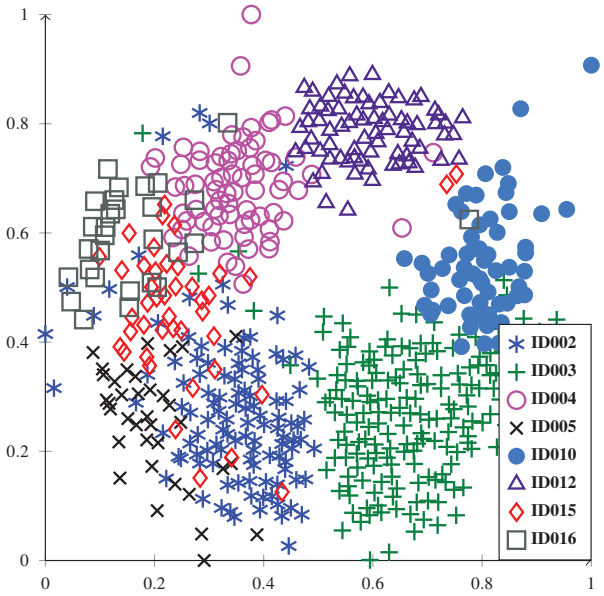
Table 2.1 Settings used for data generation.

	D_1	D_2	D_3
Λ_{train}	1:50	1:50	1:20
δ	0.2	0.1	0.2

2.1.2 Video Dataset

In addition to Synthetic datasets, the experiments in this thesis are carried out on two datasets for video-based face recognition are considered: FIA video database (Goh *et al.*, 2005) and the COX Face dataset (Huang *et al.*, 2015).

FIA video database (Goh *et al.*, 2005) contains video sequences that emulate a passport checking scenario. The video streams are collected from 221 participants under different capture conditions such as pose, illumination and expression in both indoor and outdoor environments. Videos were collected over three sessions where second and third sessions are three months later than the previous one. The participants are present before 6 cameras for about 5 seconds, resulting in total of 18 video sequences per person.



a) Examples of 2D mapping of LBP feature vectors belonging to 8 individuals using Sammon mapping (Sammon, 1969).

b) examples of 70 × 70 pixels ROIs along a trajectory captures with camera 3, during session one for ID010 with their frame numbers.

Figure 2.3 Representation of FIA dataset.

Figure 2.4 Examples of 20 ROIs captured in a trajectory for individual ID 20110319 – 0010 and camera 3 (video 2) of the COX dataset.

For experiments in this paper using FIA dataset, only the faces captured with frontal camera in indoor environment are used for both design and testing. ROIs are extracted using Viola Jones algorithm (Viola & Jones, 2001) from the video. Then, they are converted to gray-scale and rescaled to 70×70 pixels. Some examples of ROIs from this data set are presented in Figure 2.3. The number of face captures from each individual is different from the other in this dataset.

The COX Face dataset for face recognition in video surveillance (Huang *et al.*, 2015) contains videos from 1000 participants captured with 4 cameras under different capture conditions. The faces are tracked and resized such that for each frame with a face detected, an image patch centered at the head of the subject is cropped out with a size of 66×66. In this dataset, there is 25 images for each individual. Some examples of ROIs from this data set are presented in Figure 2.4.

For experiments with video data, multi-resolution gray-scale and rotation invariant Local Binary Patterns (LBP) (Ojala *et al.*, 2002) histograms have been extracted as features. The local image texture for LBP has been characterized with 8 neighbours on a 1 radius circle centred on each pixel. Finally, a feature vector with the length of 59 has been obtained for each ROI.

In these experiments, face captures from one individual (that is randomly selected as target) are considered as the positive class and the face captures from a number of randomly selected

individuals are considered as the negative class[1]. For the experiments in Chapters 3 and 4, one video is used for both design and testing whereas in Chapter 5, one video is used for design and one video is used for testing. 2×5-fold cross validation is used to split the data for design and testing. The design set is divided into 5 folds, and for each round one fold is considered for validation and remaining 4 folds are considered for training. Then the roles of design and testing sets is reversed. Therefore, for each target individual, three independent sets are collected from these face patterns for training, validation and testing. Thus, there is 10 rounds of experiments for each target individual. In the experiments in this thesis, 10 target individuals are randomly selected to carry out and average the results of the experiments that yields overall $10 \times 10 = 100$ rounds of experiments.

For the experiments in Chapters 3 and 4 two imbalance levels ($\Lambda_{\text{train}} = 1:50$ and 100) and four different imbalance levels $\Lambda_{\text{test}} = \{1:1, 1:20, 1:50, 1:100\}$ are considered for selecting the training and testing negative class samples for each positive individual, respectively. This is to evaluate the performance of different classification algorithms when they are trained on different imbalance levels, and to evaluate the robustness of the classification algorithms over varying skew levels during operations. When $\Lambda_{\text{train}} = 1:50$, for each positive individual, $T^- = 50$ individuals are used as the negative class from the training set that was collected for that positive individual. Therefore, when $\Lambda_{\text{test}} = 1:100$, there are 50 negative individuals in the testing set that were not included in training the classification systems and the skew level of test data is higher than the skew level of training data. When $\Lambda_{\text{test}} < 1:50$, most of the negative individuals that were used for training do not appear in testing data. When $\Lambda_{\text{train}} = \Lambda_{\text{test}} = 100$, the maximum imbalance level of testing data is the same as the imbalance level of training data. Therefore, all individuals are seen in both training and testing. However, in this case a high level of imbalance exists in both training and testing stages that makes both learning and classification more difficult. It is worth mentioning that in all settings, the skew level of the validation data is selected to be the same as testing data.

[1] The ROIs of each individual in FIA and COX datasets are already grouped.

In Chapter 5, two imbalance levels are considered for train, validation and testing. For $P_{\text{train}}(+) = P_{\text{validation}}(+) = P_{\text{test}}(+) = 0.1$ and $P_{\text{train}}(+) = P_{\text{validation}}(+) = P_{\text{test}}(+) = 0.04$, 9 and 24 negative class individuals are selected, respectively.

2.2 Performance Evaluation

As reviewed in section 1.2.3, global performance evaluation curves such as ROC and precision-recall, show the trade off between two metrics for different operational settings. For classifiers that output scores or probability estimates, this setting is usually the choice of decision threshold. Area under the curve, shows the global performance of the classifier over a range of possible decision thresholds, where local evaluation metric such as F-measure show the performance for a specific decision threshold.

When data is imbalanced, the same change in the number of true positives and false positives reflects more significantly in TPR than FPR. Therefore, precision is preferred to FPR because it magnifies FPR by the skew level of data for the given TPR. Consequently, AUPR (area under precision-recall curve) is preferred to AUC (area under ROC curve) when data is imbalanced. However, positive class is often the class of interest which makes TPR (or recall) very important. In this case, the F-measure is a more suitable metric to compare the performance of the classification systems.

F_β-measure shows the harmonic mean of precision and recall(Re) when a higher importance is given to recall (in other words changes in precision is eased by β^2). Therefore, in the experiments in Chapters 3 and 4, AUPR is used to compare the performance of ensembles globally which shows the average value of precision for different values of recall (or TPR) giving them equal importance. F-measure is used for giving a higher importance to recall and G-mean is used to evaluates the performance of the classification systems giving equal weight to the TPR and $TNR(= 1 - FPR)$.

In experiments of Chapter 4, the performance of the proposed ensembles in both Chapters 3 and 4 are also compared to state of the art ensembles in terms of computational complexity.

In Chapter 5, the performance is compared in ROC, Precision-Recall, Cost and the proposed F-measure spaces.

CHAPTER 3

CLASSIFIER ENSEMBLES WITH TRAJECTORY UNDER-SAMPLING FOR FACE RE-IDENTIFICATION

In this chapter, a new method of sample selection called trajectory under-sampling (TUS) is proposed to design individual-specific ensembles for face re-identification under imbalance. TUS is an application-specific sample selection method that uses contextual information from data to group samples. TUS strategy is inspired by the characteristics of the application (face re-identification) considered in this thesis i.e. face captures that are assigned to a trajectory by the tracker create a natural sub-cluster of data and sample selection strategies can be adapted to select such sub-clusters instead of general purpose sampling methods like random sampling. This sample selection technique is then used in two ensemble design strategies called random trajectory under-sampling (RTUS) and sorted trajectory under-sampling (STUS). The proposed ensemble design strategies benefit from training the base two-class classifiers of the ensembles on data subsets that have different data imbalances and complexities. Training subsets contain samples from target trajectory and a growing selection of samples from non-target trajectories to minimize the risk of information loss and to increase diversity among the classifiers trained on them as well as increasing the robustness to varying imbalance levels.

Starting from one non-target trajectory for the first subset, the level of imbalance (and decision bound complexity) is increased for the next subsets by adding a number of non-target trajectories to the previous ones. In RTUS ensemble, the trajectories are selected randomly to create these data subsets and the contribution of each classifier is weighted based on its accuracy measured on a validation set. In STUS ensemble, non-target trajectories are selected based on their proximity to the target trajectory. In another step, the proposed ensemble method is further extended by shrinking the training subsets to the Support Vectors prior to adding the next trajectories and training the next classifier in the ensemble on them. With this extended under-sampling of training subsets, only the most informative samples which are support vectors are preserved and the more relatively redundant samples are discarded. Consequently the

diversity is maintained while the classifiers are trained on less imbalanced data which in turn results in increasing their accuracy.

The accuracy and diversity of the proposed ensembles are compared to reference methods from the literature including UnderBagging approach (Barandela *et al.*, 2003), Partitional UnderBagging using random under-sampling without replacement (Yan *et al.*, 2003), SeEn-SVM (Li *et al.*, 2013), and RUSBoost (Seiffert *et al.*, 2010). Both synthetic data and videos in the FIA and COX datasets are used for the experiments. The evaluation is done with different data distribution complexities by varying imbalance level of training and test data as well as overlap between classes. The content of this chapter is an extended version of a conference paper published in the international conference in pattern recognition applications and methods (ICPRAM2016).

3.1 Trajectory Under-Sampling

It is well-known that classifier ensembles can increase accuracy and robustness over a single classifier by combining uncorrelated classifiers (Rokach, 2010). The optimal accuracy-diversity trade-off is a key factor in designing an accurate and effective ensemble of classifiers (Rokach, 2009). Even though there is no straightforward definition of diversity in the literature, base classifiers are usually deemed diverse when their misclassification events are not correlated (Rokach, 2009). Diversity in ensembles that are specialized to handle imbalance can be obtained by training base classifiers on target samples and different subsets of the non-target data. These subsets of non-target data may be collected by randomly selecting subsets from the whole set of non-target samples with replacement (Random under-sampling (RUS)) for example in UnderBagging approach (Barandela *et al.*, 2003) or RUSBoost (Seiffert *et al.*, 2010). The subsets of non-target data may also be collected by partitioning the whole set of non-target data. Partitioning can be done by random selection of bootstraps without replacement (Yan *et al.*, 2003) or by clustering the non-target set in the feature space (Li *et al.*, 2013). In sample-based partitional approaches like random under-sampling without replacement (RUSwR) (Yan *et al.*, 2003) the samples are treated independently, while in cluster-based under sampling (CUS)

techniques (Li *et al.*, 2013), the samples are under-sampled based on their relative distribution and their proximity to each other. When compared in the feature space, clustering generates non-overlapping (uncorrelated) partitions, whereas randomly selected subsets of samples both with and without replacement are highly correlated. The classifiers trained on data subsets that are uncorrelated in the feature space maintain higher diversity of opinions, however may have low accuracy in recognition of the overall data distribution. In contrast, the classifiers trained on data subsets that are correlated in the feature space and resemble the overall data distribution maintain higher accuracy. An ensemble of classifiers outperforms its components when they are accurate and make uncorrelated errors at the same time (Rokach, 2009). Therefore, a sample selection scheme is preferred to the above mentioned ones if it results in training classifiers that maintain optimal accuracy-diversity trade-off in order to design efficient ensembles.

In many applications, the data consists of subclusters inherently. In fact grouping all samples that belong to the same subject form a set of natural subclusters of the subjects. This is the case when a multi-class classification problem is converted to a one versus all binary classification problem. In this chapter we propose and investigate using such natural subclusters to under-sample data to design ensembles of classifiers for imbalanced data classification specifically for face re-identification application.

In many video surveillance applications a tracker is used to follow and regroup objects in a camera's field of view according to trajectories for spatio-temporal recognition. Spatio-temporal recognition methods may do tracking and recognition as individual tasks and combine the results (Matta & Dugelay, 2007; Mazzon *et al.*, 2012; Tao & Veldhuis, 2009; Pagano *et al.*, 2014; De-la Torre *et al.*, 2015c) or merge tracking and recognition to a single task (Saeed *et al.*, 2006; Zhou *et al.*, 2004). The tracker could follow the position of each person observed in the scene over consecutive frames, and the facial regions of interest (ROIs) of the same person are collected into a trajectory. These trajectories of the individuals can be representative of natural sub-clusters in this application.

Facial samples in a trajectory are captured under different operating conditions and consequently, they are dispersed in the feature space compared to the samples from a cluster of data that is obtained from conventional clustering methods. In fact, using this contextual information to group samples, or in other words using the natural subclusters of the data, could provide a better modelling of data from different people. For instance, the facial regions captured within trajectories are defined by different geometric and environmental conditions, so the facial region of interest (ROI) patterns in a trajectory may exhibit multi-modal distribution that correlate in some regions of the feature space. Accordingly, a pool of 2-class classifiers that are trained on samples selected based on the trajectories (trajectory under-sampling (TUS)) may provide more variability, separability, and diversity and provide better decision bounds.

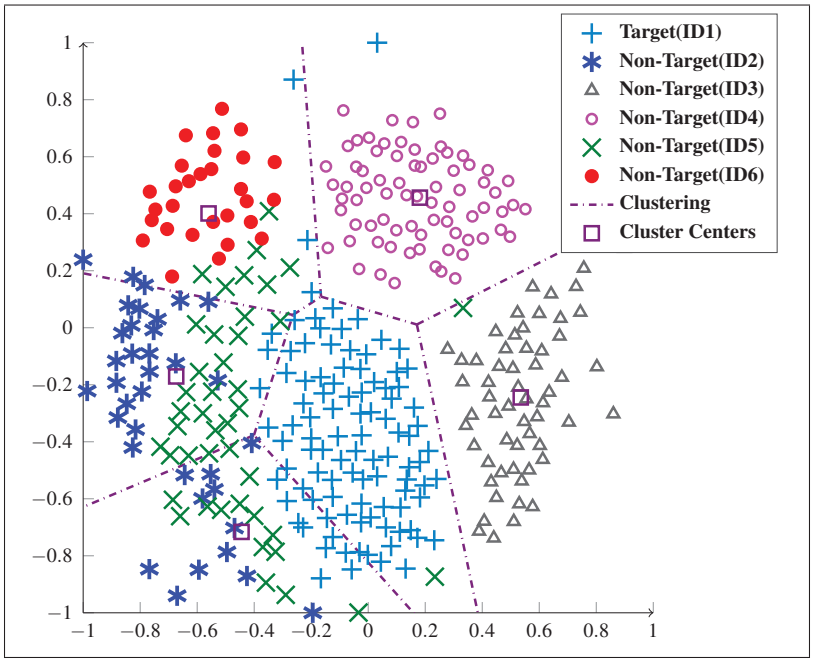


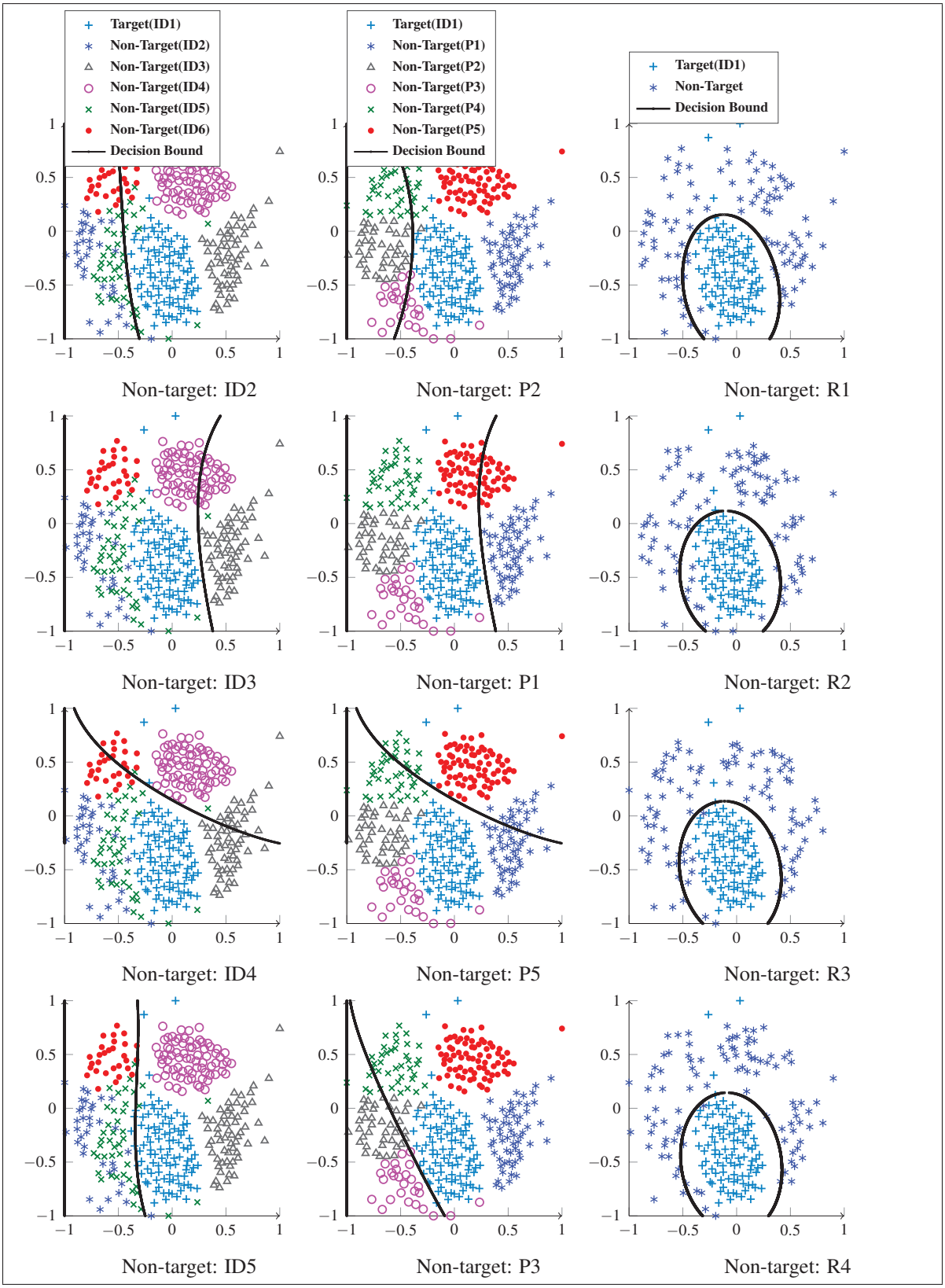
Figure 3.1 Sammon mapping of a target trajectory and 5 non-target trajectories clustered using k-means with $k = 5$

An example is shown in Figures 3.1 and 3.2 to show above-mentioned differences between RUS, CUS and TUS. In Figure 3.1, the face samples of 6 individuals (collected in trajectories) are mapped to 2D space using Sammon mapping. These face samples are also clustered using k-means clustering. It is observed that samples that belong to different trajectories exhibit

multi-modal distribution that correlate in some regions of the feature space and by clustering the whole data, three clusters contain samples from 3 different individuals. In Figure 3.2, five SVMs are trained on five data subsets selected using TUS (first column), CUS (second column), and RUS (third column). It is observed that the decision boundaries of classifiers that are trained using RUS are very similar and more accurate than CUS and TUS since RUS takes samples from all parts of the feature space. However, training on data subsets using TUS and CUS result in classifiers with diverse decision boundaries that are not accurate to classify the whole data on their own, but complement each other. This difference is presented in terms of Kappa diversity measure (see section 3.3 for more on this measure) in Figure 3.3, when the classifiers are tested on 4 different imbalance levels (shown on x-axis). When CUS and TUS are compared in Figure 3.2, the decision boundary is similar when the cluster contains the same samples in a trajectory (rows 2, 3). When they contain different samples the classifiers that are trained on trajectories are more accurate in rows 1 and 4, and the classifier that is trained on the cluster in row 5 is more accurate.

3.2 Ensembles with Trajectory Under-Sampling

This approach is specialized for video surveillance applications like person re-identification, where faces or soft biometrics are captured and regrouped in terms of trajectories. A tracker assigns a track ID to each different person appearing in the scene. During consecutive frames, the tracker follows the positions of persons and regroups the face captures along each track into trajectories. Consider the faces captured in training video streams as $\mathbf{S}_{\text{tr}} = \{(\mathbf{x}_i, y_i, \text{ID}_i); i = 1,...,M_{\text{tr}}\}$ where $y_i \in \{1,0\}$ indicates the class label, i.e. target (1) or non-target (0) classes, and ID_i is the track ID assigned by the tracker to the face. Let ID^+ be the track ID assigned by tracker to the target face. All target samples are grouped into a trajectory $\mathbf{t}^+ = \{(\mathbf{x}_p, y_p) \in \mathbf{S}_{\text{tr}} | \text{ID}_p = \text{ID}^+\}$. In the same way, the non-target samples that are assigned the same track ID are grouped into a non-target trajectory as $\mathbf{t}_j^-$. By collecting all non-target trajectories into a set $\mathbf{T}^- = \{\mathbf{t}_j^-; j = 1,..,n_T^-\}$, the overall non-target set can be under-sampled by selecting a number of $\mathbf{t}_j^-$s from this set.


Target(ID1)
Non-Target(ID2)
Non-Target(ID3)
Non-Target(ID4)
Non-Target(ID5)
Non-Target(ID6)
Decision Bound
Non-target: ID2
Target(ID1)
Non-Target(P1)
Non-Target(P2)
Non-Target(P3)
Non-Target(P4)
Non-Target(P5)
Decision Bound
Non-target: P2
Target(ID1)
Non-Target
Decision Bound
Non-target: R1
Non-target: ID3
Non-target: P1
Non-target: R2
Non-target: ID4
Non-target: P5
Non-target: R3
Non-target: ID5
Non-target: P3
Non-target: R4

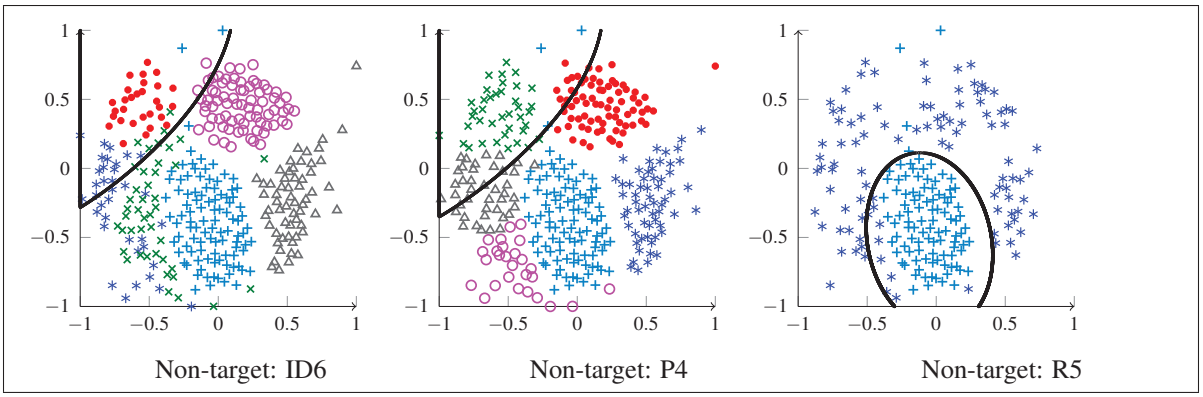


Figure 3.2 Decision boundaries of 5 SVMs trained on 2D mapping of FIA data set using TUS (first column), CUS (second column) and RUS (third column). Each figure shows the decision boundary of the an SVM trained on a subset of the negative class. The corresponding subsets are shown below each figure.

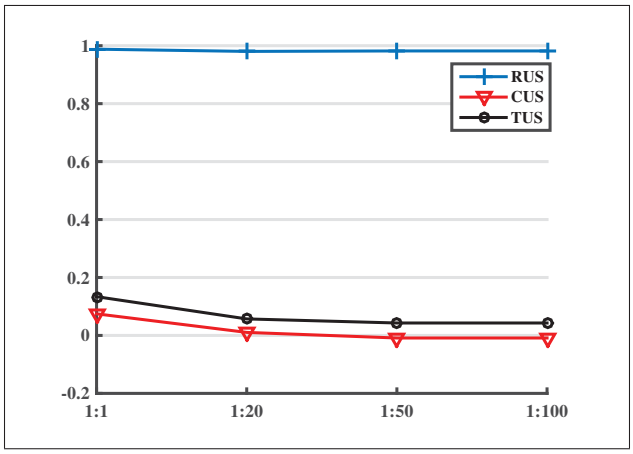


Figure 3.3 Kappa diversity measure of classifiers presented in Figure 3.2.

In this chapter, an ensemble method is proposed that is based on gradually increasing the skew level of training data subsets that are used for training the base classifiers of the ensemble. With this method of training, the ensemble becomes more robust to varying imbalance levels. In addition, training classifiers on different imbalance levels means training classifiers on data subsets with different data complexities which result in different decision boundaries and increase the diversity between the base classifiers of the ensemble. Another advantage of this method is that there won't be any information loss during under-sampling of non-target class

during design of the ensemble because all the non-target samples are used to train at least one classifier in the ensemble.

To generate design data with several skew levels, non-target samples in each subset are selected by accumulating trajectories incrementally. Two important concerns arise with this ensemble generation technique. First is the suitable selection of the ensemble size and the number of trajectories for each classifier and second is that the performance of base classifiers in the ensemble can be affected by the order of trajectories.

TUS-ensemble can be designed by training a pool of classifiers equal to the number of non-target trajectories. In other words, the skew level of design data for each base classifier is approximately one level higher than the previous one. However, the ensemble size can be limited by using larger skew steps between base classifiers. In addition, bigger difference between skew levels of classifiers in the ensemble result in greater diversity among them.

To select the ensemble size, we determine the steps between skew levels based on the overall imbalance level of design data. The level of imbalance in a data distribution is typically calculated as the proportion of overall number of non-target samples to the overall number of target ones (M^-/M^+). In this chapter, the skew level is indicated in a different way based on the number of trajectories. Letting $\mathrm{n_T}^-$ be the number of non-target trajectories, $\mathrm{n_T}^+$ be the number of target trajectories (typically $\mathrm{n_T}^+ = 1$ in a single video sequence), and n_s as the desired skew level difference between two consecutive classifiers in the ensemble, the number of imbalanced sets to design classifiers in the ensemble n_E, is determined from:

$$n_E = \left\lfloor \frac{\mathrm{n_T}^-}{n_s \mathrm{n_T}^+} \right\rfloor \tag{3.1}$$

Considering the balanced case in addition to imbalanced ones, there are $E = n_E + 1$ classifiers in the ensemble. Defining the skew level of e^{th} classifier in the ensemble as λ_e, skew levels of data subsets in the ensemble are determined from the set: $\Lambda = \{1 : \lambda_e | \lambda_1 = 1, \lambda_{e+1} = n_s \times$

$e, e = 1,3,...,n_E\}$. As an example, if $n_s = 5$ and $\mathrm{n_T}^- / 5\mathrm{n_T}^+ = 5.2$ for a dataset, the number of classifiers in the ensemble will be $E = 6$, with skew levels $\Lambda = \{1{:}1, 1{:}5, 1{:}10, 1{:}15, 1{:}20, 1{:}25\}$.

Regarding the second concern, the order of selecting non-target trajectories can be random, or predefined. Therefore, two ensembles are proposed in this chapter based on these trajectory selection schemes RTUS (random TUS) and STUS (sorted TUS). These ensembles are described in sections 3.2.1 and 3.2.2.

3.2.1 Sorted Trajectory Under-Sampling (STUS)

Some non-target trajectories are more relevant than others, and can play a critical role in defining accurate class boundary. Samples of non-target trajectories that are closer to the target class are more relevant to define good classifier decision bounds (Stefanowski & Wilk, 2008). To generate ensembles with STUS (Algorithm 3), first the non-target trajectories are sorted based on their proximity to the target class using Hausdorff distance (Edgar, 2007) that measures the distance between two sets of samples as the maximum of the minimum distances between pairs of elements from two sets. The Hausdroff distance between all non-target trajectories and target trajectory is calculated as:

$$\mathrm{HD}_j = \max\{\min \parallel \mathbf{x}^+ - \mathbf{x}^- \parallel \mid \mathbf{x}^+ \in \mathbf{t}^+, \mathbf{x}^- \in \mathbf{t}_j^-\}. \tag{3.2}$$

There are two ways to sort and select the non-target trajectories. With the first method that we call STUSA (sorted trajectory under sampling in ascending order), the trajectories are sorted from the closest to the farthest. In other words, given $D = \{\mathrm{HD}_j; j = 1,...,\mathrm{n_T}^- | \mathrm{HD}_j \leq \mathrm{HD}_{j+1}\}$, the non-target trajectories are sorted into $\mathbf{T}_s^- = \{\mathbf{t}_j^-; j = 1,..,\mathrm{n_T}^-\}$ in the same order as D. Then, for training the first classifier in the ensemble $\mathbf{t}_1^-$ is selected from $\mathbf{T}_s^-$ and for the next e-th classifiers ($e = 2,...,E$), $\{\mathbf{t}_k^-; k = 1,...,\lambda_e\}$) are used. With this method, the decision boundaries of classifiers in the ensemble get closer and closer to the target class, as the imbalance level of the training subsets increases. Therefore, the classifiers make less error in correct classification of non-target class. However, if the overlap between target and non-target class be higher, less

error in correct classification of non-target class would be in expense of more error in correct classification of target class.

The other way proposed here to sort and select the non-target trajectories is called STUSD (sorted trajectory sampling in descending order). With this method the distance vector mentioned above changes to $D = \{\mathrm{HD}_j; j = 1,...,\mathrm{n_T}^- | \mathrm{HD}_j \geq \mathrm{HD}_{j+1}\}$. The benefit of this method is that the classifiers in the ensemble make less error in correct classification of the target class. However, this might be in expense of making more error in correct classification of the non-target class if the overlap between the target and non-target class is low.

Algorithm 3: STUS Ensemble Algorithm.

Input:

- $\mathbf{S}_{\mathrm{tr}} = \{(\mathbf{x}_i, y_i; i = 1,...,M_{\mathrm{tr}}\}$: Training set
- $y_i \in \{1,0\}$: Class label of samples
- t^+: Target trajectory
- $\mathbf{T}^- = \{\mathbf{t}_j^-; j = 1,..,\mathrm{n_T}^-\}$: Non-target trajectories
- $\mathbf{X}$: Input probe sample

Output: $H(\cdot)$: Prediction function

The Hausdorff distance (Eq. 3.2) between all $\mathbf{t}_j^-$s and $\mathbf{t}^+$ are sorted into $D = \{\mathrm{HD}_j; j = 1,...,\mathrm{n_T}^- | \mathrm{HD}_j \geq \mathrm{HD}_{j+1}\}$ (or $D = \{\mathrm{HD}_j; j = 1,...,\mathrm{nT}^- | \mathrm{HD}_j \leq \mathrm{HD}_{j+1}\}$ in STUSA)

Non-target trajectories are sorted based on D into $\mathbf{T}_s^- = \{\mathbf{t}_j^-; j = 1,..,\mathrm{nT}^-\}$

The number of base classifiers E and their skew levels λ_e for $e = 1,..,E$:

$n_E = \lfloor \mathrm{n_T}^- / n_s \mathrm{n_T}^+ \rfloor$, $E = n_E + 1$, $\Lambda = \{\lambda_e | \lambda_1 = 1, \lambda_{e+1} = n_s \cdot e, e = 2,..,E\}$

for $e = 1,..,E$ **do**

  Collect a subset of $\mathbf{T}_s^-$ into $\mathbf{T}_{s,e}^- = \{\mathbf{t}_k^-,\ k = 1,...,\lambda_e\}$

  Train a classifier C_e on $\mathbf{T}_{s,e}^-$ and $\mathbf{t}^+$

for $e = 1,..,E$ **do**

  $h_e(\cdot) \leftarrow$ Prediction of C_e (classification score or decision) on $\mathbf{X}$.

Combine the predictions of classifiers: $H(\cdot) = \mathrm{sign}\left(\sum_{e=1}^{E}(h_e(\cdot))\right)$

3.2.2 Random Trajectory Under-Sampling (RTUS)

In contrast to sorted selection of trajectories, in random selection of trajectories, the performance of the classifiers in the ensemble is not predictable. In order to avoid weak classifiers in the final ensemble, more importance is given to the component classifiers with better performance in classifying imbalanced data.

In Boosting ensembles, a weight is assigned to each classifier based on its error rate. In the case of classifying imbalanced data distributions, accuracy is not an appropriate measure to evaluate the performance of a classifier. Therefore, in the proposed ensemble, the weight of each base classifier is set based on its performance measured using the F-measure, because this is a more suitable metric when classifying imbalanced data.

The pseudo code of RTUS ensemble is presented in Algorithm 4. When a classifier is trained for the ensemble it is tested with a validation subset to determine its weight. This validation subset should have the same level of imbalance (λ_e) as the training subset. The performance F_e of the $e-$th classifier in the ensemble is measured in terms of the F-measure and its weight is assigned using:

$$w_e = \log\left(\frac{F^e}{1-F^e}\right). \tag{3.3}$$

This weight is then used for weighted combination of the ensemble.

Unlike STUS, this method of selecting trajectories may not suffer from the issue of correctly classifying one class in expense of misclassifying the other. However, the change in the decision boundary of the classifiers is not predictable, as the skew level of training data subsets increases. Nevertheless, assigning weights to classifiers in the final prediction of the ensemble lowers the impact of the weak classifiers.

3.2.3 Under-Sampling Trajectories to Support Vectors

As explained in section 1.2, training classifiers on imbalanced data moves their decision boundary towards the target class. Under-sampling the non-target class in order to balance the number

Algorithm 4: RTUS Ensemble Algorithm.

Input:

- $\mathbf{S}_{\text{tr}} = \{(\mathbf{x}_i, y_i); i = 1, ..., M_{\text{tr}}\}$: Training set
- $\mathbf{S}_{\text{val}} = \{(\mathbf{x}_i, y_i); i = 1, ..., M_{\text{val}}\}$: Validation set
- $y_i \in \{+1, -1\}$: Class label of samples
- t^+: Target trajectory
- $\mathbf{T}^- = \{\mathbf{t}_j^-; j = 1, .., N^-\}$: Non-target trajectories
- $\mathbf{X}$: Input probe sample

Output: $H(\cdot)$: Prediction function.

Non-target trajectories are randomly shuffled into $\mathbf{T}_r^- = \{\mathbf{t}_j^-; j = 1, .., n_T^-\}$
The number of base classifiers E and their skew levels λ_e for $e = 1, 2, ..., E$:
$n_E = \lfloor n_T^- / n_s n_T^+ \rfloor$, $E = n_E + 1$, $\Lambda = \{1 : \lambda_e | \lambda_1 = 1, \lambda_{e+1} = n_E e\}$
for $e = 1, .., E$ **do**
    Collect a subset of $\mathbf{T}_r^-$ into
    $\mathbf{T}_{r,e}^- = \{\mathbf{t}_k^-; k = 1, ..., \lambda_e\}$
    Train a classifier C_e on $\mathbf{T}_{r,e}^-$ and $\mathbf{t}^+$
    Under-sample the validation set $\mathbf{S}_{\text{val}}$ to λ_e level
    $F_e \leftarrow$ the F-measure attained by C_e on the validation subset
    Set the weight of C_e as: $w_e = \log(\frac{F_e}{1-F_e})$
for $e = 1, .., E$ **do**
    $h_e(\cdot) \leftarrow$ Prediction of C_e (classification score or decision) on $\mathbf{X}$.
Combine the predictions of classifiers: $H(\cdot) = \text{sign}\left(\sum_{e=1}^{E}(w_e h_e(\cdot))\right)$

of samples from both classes resolves this issue, however, cause information loss and higher misclassification rate of the non-target class. Combining classifiers trained on data subsets with different imbalance levels avoids both of these biases and result in a classification system that is more robust to imbalance and perform well in classifying both classes. However, the bias of performance in classifiers that are trained on imbalanced subsets of data can be further reduced by under-sampling the non-target trajectories to the support vectors. Tang *et al.* (2009) utilize SVM for data cleaning/under-sampling because support vectors are the training samples that define classification model of an SVM classifier and other samples can be safely removed without substantially affecting classification. Therefore, we modify the proposed RTUS and STUS algorithms in another step as presented in Algorithms 5 and 6. The reason is that not all the samples are informative and eliminating non-informative samples from the progressive training

of the base classifiers in the ensemble decreases the computation and memory use. In addition, the accuracy of the base classifiers increases since they are training on less imbalanced training subsets. For the first classifier in the ensemble, one target and one non-target trajectories are used to train an SVM. The non-target support vectors of this classifier is stored and then it is added to E non-target trajectories to train the second classifier in the ensemble. The non-target support vectors of this classifier is again stored, and the same procedure is repeated to train remaining classifiers in the ensemble. Note that in contrast to the original versions of the RTUS and STUS ensembles, this modification is only applicable if the base classifier of the ensemble is an SVM classifier. RTUS and STUS ensembles are applicable with any discriminative classifier.

Algorithm 5: STUS Ensemble Algorithm using SVs.

Input:

- $\mathbf{S}_{\text{tr}} = \{(\mathbf{x}_i, y_i; i = 1, ..., M_{\text{tr}}\}$: Training set
- $y_i \in \{1, 0\}$: Class label of samples
- t^+: Target trajectory
- $\mathbf{T}^- = \{\mathbf{t}_j^-; j = 1, .., \text{nT}^-\}$: Non-target trajectories
- $\mathbf{X}$: Input probe sample

Output: $H(\cdot)$: Prediction function.

The Hausdorff distance (Eq. 3.2) between all $\mathbf{t}_j^-$s and $\mathbf{t}^+$ are sorted into $D = \{\text{HD}_j; j = 1, ..., \text{nT}^- | \text{HD}_j \geq \text{HD}_{j+1}\}$ (or $D = \{\text{HD}_j; j = 1, ..., \text{nT}^- | \text{HD}_j \leq \text{HD}_{j+1}\}$ in STUSA)
Non-target trajectories are sorted based on D into $\mathbf{T}_s^- = \{\mathbf{t}_j^-; j = 1, .., \text{nT}^-\}$
The number of base classifiers E and their skew levels λ_e for $e = 1, .., E$:
$n_E = \lfloor \text{nT}^- / n_s \text{nT}^+ \rfloor$, $E = n_E + 1$, $\Lambda = \{\lambda_e | \lambda_1 = 1, \lambda_{e+1} = n_s \cdot e\}$
Collect a subset of $T_s^- = t_1^-$.
Train an SVM classifier C_1 on T_s^- and t^+
Store the non-target support vectors of C_1 into a set SV_1
for $e = 2, .., E$ **do**
    Collect a subset of $\mathbf{T}_s^-$ into $\mathbf{T}_{s,e}^- = \{SV_{\text{e}-1}, \mathbf{t}_k^-,\ k = e-1, ..., \lambda_e, e = 2, .., E\}$
    Train a classifier C_e on $\mathbf{T}_{s,e}^-$ and $\mathbf{t}^+$
for $e = 1, .., E$ **do**
    $h_e(\cdot) \leftarrow$ Prediction of C_e (classification score or decision) on $\mathbf{X}$.
Combine the predictions of classifiers: $H(\cdot) = \text{sign}\left(\sum_{e=1}^{E} (h_e(\cdot))\right)$

Algorithm 6: RTUS Ensemble Algorithm using SVs.

Input:

- $\mathbf{S}_{\text{tr}} = \{(\mathbf{x}_i, y_i); i = 1, ..., M_{\text{tr}}\}$: Training set
- $\mathbf{S}_{\text{val}} = \{(\mathbf{x}_i, y_i); i = 1, ..., M_{\text{val}}\}$: Validation set
- $y_i \in \{+1, -1\}$: Class label of samples
- t^+: Target trajectory
- $\mathbf{T}^- = \{\mathbf{t}_j^-; j = 1, .., N^-\}$: Non-target trajectories

Output: $H(\mathbf{X})$: Prediction function.

Non-target trajectories are randomly shuffled into $\mathbf{T}_r^- = \{\mathbf{t}_j^-; j = 1, .., n_T^-\}$
The number of base classifiers E and their skew levels λ_e for $e = 1, 2, ..., E$:
$n_E = \lfloor n_T^- / n_s n_T^+ \rfloor$, $E = n_E + 1$, $\Lambda = \{1 : \lambda_e | \lambda_1 = 1, \lambda_{e+1} = n_E e\}$
Collect a subset of $T_r^- = t_1^-$.
Train an SVM classifier C_1 on T_r^- and t^+
Store the non-target support vectors of C_1 into a set SV_1
for $e = 2, .., E$ **do**
    Collect a subset of $\mathbf{T}_r^-$ into $\mathbf{T}_{r,e}^- = \{SV_{\text{e}-1}, \mathbf{t}_k^-,\ k = e-1, ..., \lambda_e, e = 2, .., E\}$
    Train a classifier C_e on $\mathbf{T}_{r,e}^-$ and $\mathbf{t}^+$
for $e = 1, .., E$ **do**
    $h_e(\cdot) \leftarrow$ Prediction of C_e (classification score or decision) on $\mathbf{X}$.
Combine the predictions of classifiers: $H(\cdot) = \text{sign}\left(\sum_{e=1}^{E}(h_e(\cdot))\right)$

3.3 Experimental Methodology

In the experiments of this chapter, different versions of the proposed RTUS and STUS ensembles are compared to ensemble methods from the literature in the same family of methods including UnderBagging approach (Barandela *et al.*, 2003), Partitional UnderBagging using random under-sampling without replacement (Yan *et al.*, 2003), SeEn-SVM (Li *et al.*, 2013), and RUSBoost (Seiffert *et al.*, 2010). The experiments are carried out on both synthetic and video datasets described in Chapter 2 and an RBF-SVM classifier is used as the base classifier. The classification systems are compared in terms of both accuracy and diversity.

The accuracy is measured in terms of AUPR (as a global performance metric), the F-measure by setting $\beta = 2$, and the G-mean (to measure the performance for both classes at the same time disregarding the imbalance). Therefore, the ensembles are combined in two ways. The

F-measure and G-mean scalar metrics are obtained by combining decisions of the classifiers and AUPR is obtained by combining scores of the classifiers.

Several measures of ensemble diversity have been introduced in the literature (Kuncheva & Whitaker, 2003) Q-statistics and Kappa diversity measures are the most commonly used metrics among them. Measuring diversity between several classifiers in terms of Q-statistics involves calculating Q-statistics between pairs of classifiers on a given validation set of data and averaging them. The diversity between several classifiers in terms of Kappa measure can be calculated both pairwise and non-pairwise.

Considering d_i and d_k as decisions of a pair of classifiers on the validation data, the diversity matrix in Table 3.1 is obtained and the pairwise Q-statistics for the i^{th} and k^{th} classifiers is calculated as:

$$q_{i,k} = \frac{D^{cc}D^{ww} - D^{cw}D^{wc}}{D^{cc}D^{ww} + D^{cw}D^{wc}}. \tag{3.4}$$

The overall Q_{av} measure for E classifiers is calculated as:

$$Q_{av} = \frac{2}{E(E-1)} \sum_{i=1}^{E-1} \sum_{k=i+1}^{E} q_{i,k}. \tag{3.5}$$

Non-pairwise kappa diversity measure (κ) is calculated as follows:

$$\bar{\rho} = \frac{1}{EM} \sum_{j=1}^{M} \sum_{n=1}^{E} Y_e^j, \tag{3.6}$$

$$\kappa = 1 - \frac{\frac{1}{E}\sum_{j=1}^{M} L_j(E - L_j)}{M(E-1)\bar{\rho}(1-\bar{\rho})}. \tag{3.7}$$

where $Y_e^j \in \{0,1\}$ is the classification result of the e^{th} classifier out of E classifiers on the j^{th} validation sample x_j out of overall M samples. $L_j = \sum_{e=1}^{E} Y_e^j$ is the number of classifiers that correctly classify x_j. Smaller values of κ and Q indicate high diversity, $\kappa = 0$ and $Q = -1$ indicates independent classifiers, and $\kappa = Q = 1$ indicates identical classifiers citekuncheva2003measures. Take note that these measures also rely on the accuracy of the classifiers and in fact can be used

to evaluate the base classifiers of an ensemble in terms of both accuracy and diversity at the same time. Q and κ are greater if the classifiers are more accurate and less diverse. In turn if Q and κ be low, it could either mean that the decisions of most of the classifiers are not correlated, or that the classifiers are not accurate in general.

Table 3.1 Diversity measure matrix.

	d_i correct	d_i wrong
d_j correct	D^{cc}	D^{cw}
d_j wrong	D^{wc}	D^{ww}

Different versions of RTUS and STUS are implemented whose properties are explained in the following:

- RTUS-1, STUSA-1, STUSD-1:

  RTUS, STUSA (sorted TUS with distances in ascending order), STUSD (sorted TUS with distances in descending order) when the difference between two consecutive training subsets is one ($n_s = 1$), meaning that the imbalance is increased by adding one trajectory to the previous ones at a time. The skew levels of data subsets used to train classifiers in these ensembles are determined from the set: $\Lambda = \{1 : \lambda_e | \lambda_1 = 1, \lambda_{e+1} = n_s \times e, e = 1,3,...,n_E\}$.

- RTUS-5, STUSA-5, STUSD-5: RTUS, STUSA, STUSD when the difference between two consecutive training subsets is five ($n_s = 5$). The skew levels of data subsets used to train classifiers in these ensembles are determined from the set: $\Lambda = \{1 : \lambda_e | \lambda_1 = 1, \lambda_{e+1} = n_s \times e, e = 1,3,...,n_E\}$.

- RTUS-1sv, STUSA-1sv, STUSD-1sv, RTUS-5sv, STUSA-5sv, STUSD-5sv: RTUS-1, STUSA-1, STUSD-1, RTUS-5, STUSA-5, STUSD-5 when the trajectories are under-sampled by preserving support vectors.

From the literature, the following systems are implemented and compared. UnderBagging approach (Barandela *et al.*, 2003) uses random under-sampling with replacement and we show it by RUS in this chapter. Partitional UnderBagging using random under-sampling without replacement (Yan *et al.*, 2003) is shown by RUSwR. In addition to SeEn-SVM (Li et al., 2013), another cluster under-sampling ensemble is implemented which is shown as CUS, that is similar to RUS with the difference that the number of clusters in this ensemble is found based on Dunn index (Dunn, 1973). Dunn index depends on the proportion of the intra-cluster to the inter-cluster distance between samples. To select the number of clusters, it is varied over a range of possible values and the value of Dunn index is calculated for each case using a validation set. Finally, the optimal number of clusters, is selected when Dunn index takes its maximum value. In addition, these methods are modified and implemented by growing imbalance (similarly to RTUS-1 explained above) to analyze the effect of varying imbalance of training subsets. These modified versions are shown as RUSG, RUSwRG, and CUSG.

In the experimental protocol, the results of the algorithms have been averaged by alternating the target individual among overall 10 target individuals and using 2×5-fold cross-validation as described in Chapter 2. The test data subsets with different skew levels $\Lambda_{\text{test}} = \{1:1, 1:20, 1:50, 1:100\}$ are used to evaluate the robustness of each approach over varying skew levels during operation.

In summary, the experiments are carried out to analyze the following aspects in terms of accuracy and diversity:

1. Effect of varying imbalance of training subsets.
2. Comparing TUS versus RUS, RUSwR and CUS.
3. Comparing the proposed RTUS and STUS ensembles versus the literature.
   - RTUS versus STUS (ordered descending and ascending).
   - Effect of steps of increasing imbalance (1 versus 5).
   - Effect of under-sampling trajectories using support vectors.

4. Comparing the robustness of the ensembles to varying skew level in the testing data.

5. Comparing the performance of the ensembles for different levels of overlap and skew between classes in the training data.

3.4 Results and Discussion

3.4.1 Results of experiments with synthetic datasets

Tables 3.2, 3.3 and 3.4 show the results of experiments on synthetic data-sets. As expected, increasing the level of imbalance in test data result in degradation of performance for all classification systems in terms of AUPR and F-measure but not in terms of G-mean. Higher level of overlap between classes for the same skew level of training data (D_2 and D_1) as well as lower imbalance level of training data for the same level of overlap (D_3 and D_1) lower the performance of the classification systems.

It is observed that modifying RUS, RUSwR, and CUS by using training data subsets with different skew levels improves their performance in terms of F-measure and G-mean. However, the performance does not improve in terms of AUPR. The same applies when TUS is compared to the variations of RTUS and STUS.

The performance of TUS ensemble is very similar to that of CUS ensemble and both outperform RUS and RUSwR in terms of AUPR. However, the TUS and CUS ensembles degrade in performance in terms of F-measure and G-mean. These ensembles are better used by classification score combination rather than decision combination.

By increasing the difference of imbalance between classifiers from one to five, the performance of RTUS and STUSA ensembles improve or stay the same in terms of AUPR, F-measure and G-mean with datasets D_1, D_2, and D_3. The performance of STUSD also improves in terms of F-measure and G-mean. With highly overlapped data D_2, the performance of STUSD improves in terms of AUPR, however, it degrades with less overlapped data D_1 and D_3. In summary, the

Table 3.2 Average of AUPR performance of baseline techniques with and without F-measure loss factor on synthetic data over different levels of skew and overlap in test data.

Ensembles	**Train Data**	D_1 (Λ_{train} =1:50, $\delta = 0.2$)				D_2 (Λ_{train} =1:50, $\delta = 0.1$)				D_3 (Λ_{train} =1:20, $\delta = 0.2$)			
	Λ_{test}	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**
RUSBoost		0.98 ± 0.00	0.83 ± 0.08	0.73 ± 0.14	0.68 ± 0.15	0.63 ± 0.31	0.44 ± 0.24	0.37 ± 0.30	0.19 ± 0.15	0.96 ± 0.01	0.75 ± 0.06	0.75 ± 0.06	0.37 ± 0.07
Seg-EoSVMs		0.98 ± 0.00	0.04 ± 0.00	0.02 ± 0.00	0.01 ± 0.00	0.98 ± 0.00	0.04 ± 0.00	0.02 ± 0.00	0.01 ± 0.00	0.98 ± 0.00	0.05 ± 0.00	0.02 ± 0.00	0.01 ± 0.00
RUS		0.98 ± 0.00	0.93 ± 0.02	0.90 ± 0.02	0.87 ± 0.03	0.94 ± 0.02	0.66 ± 0.05	0.66 ± 0.05	0.38 ± 0.02	0.98 ± 0.00	0.88 ± 0.03	0.88 ± 0.03	0.37 ± 0.08
RUSwR		0.98 ± 0.00	0.94 ± 0.03	0.90 ± 0.02	0.87 ± 0.04	0.91 ± 0.05	0.61 ± 0.08	0.61 ± 0.08	0.35 ± 0.04	0.97 ± 0.01	0.85 ± 0.06	0.85 ± 0.06	0.34 ± 0.07
RUSG		0.98 ± 0.00	0.93 ± 0.03	0.90 ± 0.03	0.54 ± 0.21	0.98 ± 0.00	0.90 ± 0.02	0.90 ± 0.02	**0.43** ± 0.05	0.98 ± 0.00	0.92 ± 0.03	0.74 ± 0.11	0.25 ± 0.08
RUSwRG		0.98 ± 0.00	0.93 ± 0.03	0.90 ± 0.03	0.54 ± 0.21	0.98 ± 0.00	0.91 ± 0.01	**0.91** ± 0.01	0.42 ± 0.05	0.98 ± 0.00	0.92 ± 0.02	0.77 ± 0.09	0.24 ± 0.05
CUS		0.98 ± 0.00	**0.97** ± 0.01	**0.93** ± 0.02	**0.90** ± 0.02	0.98 ± 0.00	0.90 ± 0.03	0.90 ± 0.03	0.42 ± 0.05	0.98 ± 0.00	**0.95** ± 0.02	0.93 ± 0.03	**0.68** ± 0.06
CUSG		0.98 ± 0.01	0.91 ± 0.07	0.87 ± 0.09	0.64 ± 0.22	0.98 ± 0.00	0.90 ± 0.02	0.89 ± 0.02	0.41 ± 0.06	0.98 ± 0.00	0.93 ± 0.02	0.85 ± 0.08	0.37 ± 0.15
TUS		0.98 ± 0.00	**0.97** ± 0.00	**0.94** ± 0.01	**0.90** ± 0.01	0.98 ± 0.00	0.87 ± 0.03	0.87 ± 0.03	0.42 ± 0.05	0.98 ± 0.00	**0.96** ± 0.01	**0.94** ± 0.02	0.60 ± 0.07
RTUS-1		0.98 ± 0.00	0.94 ± 0.02	0.91 ± 0.03	0.56 ± 0.23	0.98 ± 0.00	**0.93** ± 0.00	0.89 ± 0.01	0.38 ± 0.06	0.98 ± 0.00	0.82 ± 0.03	0.49 ± 0.10	0.11 ± 0.01
RTUS-1s		0.98 ± 0.00	**0.97** ± 0.00	**0.94** ± 0.00	**0.92** ± 0.01	0.98 ± 0.00	0.77 ± 0.03	0.77 ± 0.03	0.42 ± 0.02	0.98 ± 0.00	**0.95** ± 0.02	**0.95** ± 0.02	**0.75** ± 0.04
RTUS-5		0.98 ± 0.00	0.94 ± 0.02	0.91 ± 0.03	0.51 ± 0.24	0.98 ± 0.00	**0.93** ± 0.00	0.89 ± 0.01	0.38 ± 0.06	0.98 ± 0.00	0.85 ± 0.04	0.56 ± 0.10	0.13 ± 0.02
RTUS-5s		0.98 ± 0.00	0.92 ± 0.02	0.84 ± 0.06	0.73 ± 0.14	0.98 ± 0.00	0.48 ± 0.21	0.19 ± 0.15	0.09 ± 0.06	0.98 ± 0.00	0.37 ± 0.06	0.31 ± 0.06	0.12 ± 0.03
STUSA-1		0.97 ± 0.00	0.92 ± 0.03	0.89 ± 0.04	0.47 ± 0.22	0.98 ± 0.00	**0.92** ± 0.01	**0.91** ± 0.00	0.38 ± 0.03	0.98 ± 0.00	0.87 ± 0.06	0.61 ± 0.11	0.14 ± 0.03
STUSA-1s		0.97 ± 0.01	0.90 ± 0.04	0.87 ± 0.05	0.44 ± 0.22	0.98 ± 0.00	**0.93** ± 0.01	**0.91** ± 0.00	0.33 ± 0.01	0.97 ± 0.01	0.83 ± 0.05	0.63 ± 0.11	0.11 ± 0.02
STUSA-5		0.97 ± 0.00	0.92 ± 0.03	0.89 ± 0.04	0.48 ± 0.22	0.98 ± 0.00	0.91 ± 0.01	**0.91** ± 0.01	0.39 ± 0.03	0.98 ± 0.00	0.90 ± 0.04	0.65 ± 0.13	0.16 ± 0.03
STUSA-5s		0.98 ± 0.00	0.91 ± 0.03	0.88 ± 0.04	0.45 ± 0.21	0.98 ± 0.00	**0.93** ± 0.01	**0.92** ± 0.00	0.33 ± 0.01	0.97 ± 0.01	0.90 ± 0.04	0.74 ± 0.13	0.14 ± 0.03
STUSD-1		0.98 ± 0.00	**0.96** ± 0.01	**0.93** ± 0.01	0.86 ± 0.04	0.98 ± 0.00	0.83 ± 0.02	0.83 ± 0.02	**0.43** ± 0.04	0.98 ± 0.00	0.94 ± 0.02	0.92 ± 0.04	0.66 ± 0.03
STUSD-1s		0.98 ± 0.00	0.95 ± 0.01	**0.93** ± 0.01	0.86 ± 0.04	0.98 ± 0.00	0.83 ± 0.02	0.83 ± 0.02	**0.43** ± 0.04	0.98 ± 0.00	0.94 ± 0.02	0.92 ± 0.04	**0.68** ± 0.03
STUSD-5		0.98 ± 0.00	0.95 ± 0.01	0.92 ± 0.01	0.81 ± 0.08	0.98 ± 0.00	0.90 ± 0.02	0.90 ± 0.02	**0.44** ± 0.06	0.98 ± 0.00	0.93 ± 0.02	0.87 ± 0.06	0.45 ± 0.06
STUSD-5s		0.98 ± 0.00	0.95 ± 0.02	0.92 ± 0.02	0.80 ± 0.08	0.98 ± 0.00	0.90 ± 0.02	0.90 ± 0.02	**0.44** ± 0.06	0.98 ± 0.00	0.93 ± 0.03	0.84 ± 0.08	0.50 ± 0.05

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

performance of TUS ensembles improves or stays the same by increasing the difference of imbalance between training subsets from one to five even though the size of the ensemble

Table 3.3 Average of F_2-measure performance of baseline techniques with and without F-measure loss factor on synthetic data over different levels of skew and overlap in test data.

Ensembles	**Train Data**	D_1 (Λ_{train} =1:50, $\delta = 0.2$)				D_2 (Λ_{train} =1:50, $\delta = 0.1$)				D_3 (Λ_{train} =1:20, $\delta = 0.2$)			
	Λ_{test}	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**
RUSBoost		0.92 ± 0.04	0.24 ± 0.02	0.13 ± 0.01	0.07 ± 0.01	0.67 ± 0.00	0.40 ± 0.07	0.15 ± 0.05	0.07 ± 0.02	0.67 ± 0.00	0.37 ± 0.01	0.29 ± 0.02	0.12 ± 0.01
Seg-EoSVMs		0.76 ± 0.06	0.10 ± 0.00	0.05 ± 0.00	0.02 ± 0.00	0.67 ± 0.00	0.10 ± 0.00	0.05 ± 0.00	0.02 ± 0.00	0.67 ± 0.00	0.11 ± 0.01	0.05 ± 0.00	0.02 ± 0.00
RUS		**0.98** ± 0.02	0.33 ± 0.01	0.19 ± 0.01	0.10 ± 0.01	0.67 ± 0.00	0.50 ± 0.00	0.25 ± 0.01	0.10 ± 0.00	0.71 ± 0.02	0.42 ± 0.01	0.33 ± 0.02	0.16 ± 0.02
RUSwR		**1.00** ± 0.00	0.42 ± 0.02	0.27 ± 0.02	0.16 ± 0.01	0.67 ± 0.01	0.50 ± 0.01	0.32 ± 0.01	0.13 ± 0.01	0.76 ± 0.03	0.46 ± 0.01	0.42 ± 0.01	0.22 ± 0.02
RUSG		0.94 ± 0.03	0.84 ± 0.04	0.78 ± 0.03	0.47 ± 0.07	**0.99** ± 0.01	0.87 ± 0.03	0.86 ± 0.04	0.37 ± 0.02	**0.99** ± 0.02	0.82 ± 0.02	0.53 ± 0.06	0.25 ± 0.02
RUSwRG		0.94 ± 0.03	**0.85** ± 0.04	0.80 ± 0.02	0.48 ± 0.07	**0.99** ± 0.01	0.88 ± 0.02	0.86 ± 0.04	0.38 ± 0.02	0.98 ± 0.02	0.84 ± 0.02	0.56 ± 0.07	0.27 ± 0.03
CUS		0.67 ± 0.03	0.09 ± 0.00	0.04 ± 0.00	0.02 ± 0.00	0.67 ± 0.00	0.10 ± 0.01	0.04 ± 0.00	0.02 ± 0.00	0.67 ± 0.00	0.11 ± 0.00	0.05 ± 0.00	0.02 ± 0.00
CUSG		0.95 ± 0.05	0.75 ± 0.17	0.64 ± 0.22	0.38 ± 0.17	0.86 ± 0.14	0.71 ± 0.20	0.54 ± 0.26	0.24 ± 0.12	0.98 ± 0.03	0.65 ± 0.20	0.46 ± 0.21	0.22 ± 0.09
TUS		0.67 ± 0.00	0.09 ± 0.00	0.04 ± 0.00	0.02 ± 0.00	0.67 ± 0.00	0.09 ± 0.00	0.04 ± 0.00	0.02 ± 0.00	0.67 ± 0.00	0.11 ± 0.00	0.05 ± 0.00	0.02 ± 0.00
RTUS-1		0.93 ± 0.04	**0.86** ± 0.04	0.81 ± 0.03	0.47 ± 0.09	**0.98** ± 0.01	**0.90** ± 0.01	0.49 ± 0.03	0.23 ± 0.01	0.95 ± 0.03	0.62 ± 0.03	0.39 ± 0.04	0.17 ± 0.01
RTUS-1s		**1.00** ± 0.00	0.81 ± 0.04	0.70 ± 0.05	0.50 ± 0.03	0.85 ± 0.02	0.62 ± 0.01	0.58 ± 0.02	0.27 ± 0.01	**0.99** ± 0.01	0.80 ± 0.03	**0.74** ± 0.02	**0.44** ± 0.02
RTUS-5		0.93 ± 0.04	**0.86** ± 0.04	0.81 ± 0.03	0.48 ± 0.10	**0.98** ± 0.01	**0.91** ± 0.01	0.74 ± 0.02	0.33 ± 0.02	0.94 ± 0.02	0.68 ± 0.02	0.42 ± 0.02	0.18 ± 0.01
RTUS-5s		0.76 ± 0.02	0.11 ± 0.00	0.05 ± 0.00	0.02 ± 0.00	**0.98** ± 0.03	0.13 ± 0.01	0.06 ± 0.00	0.03 ± 0.00	**1.00** ± 0.00	0.16 ± 0.00	0.06 ± 0.00	0.03 ± 0.00
STUSA-1		0.92 ± 0.05	**0.86** ± 0.06	**0.83** ± 0.05	**0.54** ± 0.10	**0.98** ± 0.01	**0.90** ± 0.01	**0.89** ± 0.02	**0.42** ± 0.03	0.94 ± 0.03	**0.86** ± 0.03	0.61 ± 0.10	**0.28** ± 0.04
STUSA-1s		0.90 ± 0.06	0.84 ± 0.07	**0.82** ± 0.07	0.52 ± 0.13	0.93 ± 0.02	0.88 ± 0.02	**0.87** ± 0.03	**0.43** ± 0.02	0.92 ± 0.03	0.84 ± 0.03	**0.66** ± 0.10	0.27 ± 0.04
STUSA-5		0.92 ± 0.06	**0.86** ± 0.06	**0.83** ± 0.06	**0.54** ± 0.11	**0.98** ± 0.01	**0.90** ± 0.01	**0.89** ± 0.01	**0.43** ± 0.02	0.94 ± 0.03	**0.87** ± 0.02	0.62 ± 0.09	**0.28** ± 0.04
STUSA-5s		0.90 ± 0.06	**0.85** ± 0.07	**0.82** ± 0.07	**0.53** ± 0.13	0.92 ± 0.02	0.88 ± 0.02	**0.87** ± 0.02	**0.43** ± 0.02	0.93 ± 0.03	0.84 ± 0.03	**0.66** ± 0.10	0.27 ± 0.04
STUSD-1		0.89 ± 0.04	0.26 ± 0.00	0.14 ± 0.00	0.07 ± 0.00	0.67 ± 0.00	0.42 ± 0.01	0.17 ± 0.00	0.08 ± 0.00	0.67 ± 0.00	0.29 ± 0.01	0.21 ± 0.01	0.09 ± 0.00
STUSD-1s		0.89 ± 0.03	0.26 ± 0.01	0.14 ± 0.00	0.07 ± 0.00	0.67 ± 0.00	0.43 ± 0.01	0.17 ± 0.00	0.08 ± 0.00	0.67 ± 0.00	0.29 ± 0.01	0.21 ± 0.01	0.09 ± 0.00
STUSD-5		0.95 ± 0.02	0.31 ± 0.00	0.17 ± 0.00	0.09 ± 0.00	0.67 ± 0.00	0.46 ± 0.01	0.20 ± 0.01	0.09 ± 0.00	0.67 ± 0.00	0.38 ± 0.01	0.32 ± 0.01	0.16 ± 0.01
STUSD-5s		0.95 ± 0.01	0.30 ± 0.00	0.16 ± 0.00	0.09 ± 0.00	0.67 ± 0.00	0.46 ± 0.01	0.19 ± 0.01	0.08 ± 0.00	0.67 ± 0.00	0.36 ± 0.01	0.29 ± 0.01	0.14 ± 0.01

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

reduces in this case. The reason is that by increasing the difference of imbalance between data subsets from one to five the diversity between classifiers trained on them increases.

Table 3.4 Average of G-mean performance of baseline techniques with and without F-measure loss factor on synthetic data over different levels of skew and overlap in test data.

Ensembles	**Train Data**	D_1 (Λ_{train} =1:50, $\delta = 0.2$)				D_2 (Λ_{train} =1:50, $\delta = 0.1$)				D_3 (Λ_{train} =1:20, $\delta = 0.2$)			
	Λ_{test}	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**
RUSBoost		0.91 ± 0.06	0.83 ± 0.02	0.86 ± 0.01	0.86 ± 0.01	0.00 ± 0.00	0.92 ± 0.03	0.87 ± 0.05	0.83 ± 0.06	0.05 ± 0.09	0.91 ± 0.01	0.95 ± 0.00	0.93 ± 0.01
Seg-EoSVMs		0.55 ± 0.23	0.37 ± 0.06	0.43 ± 0.04	0.43 ± 0.03	0.00 ± 0.00	0.31 ± 0.03	0.44 ± 0.03	0.44 ± 0.03	0.00 ± 0.00	0.39 ± 0.07	0.43 ± 0.04	0.45 ± 0.05
RUS		**0.97** ± 0.02	0.89 ± 0.01	0.91 ± 0.01	0.91 ± 0.01	0.00 ± 0.00	0.95 ± 0.00	0.94 ± 0.00	0.91 ± 0.00	0.43 ± 0.09	0.93 ± 0.00	0.96 ± 0.00	0.95 ± 0.01
RUSwR		**1.00** ± 0.00	0.93 ± 0.01	**0.94** ± 0.01	**0.94** ± 0.01	0.06 ± 0.13	0.95 ± 0.00	0.96 ± 0.00	0.93 ± 0.00	0.59 ± 0.08	0.94 ± 0.00	**0.97** ± 0.00	**0.96** ± 0.00
RUSG		0.94 ± 0.03	0.93 ± 0.03	**0.94** ± 0.03	0.93 ± 0.03	**0.99** ± 0.01	**0.99** ± 0.01	**0.99** ± 0.01	**0.98** ± 0.01	**0.99** ± 0.02	**0.98** ± 0.01	**0.97** ± 0.02	0.96 ± 0.02
RUSwRG		0.94 ± 0.03	**0.94** ± 0.03	**0.94** ± 0.03	0.93 ± 0.03	**0.99** ± 0.01	**0.99** ± 0.01	**0.99** ± 0.01	**0.98** ± 0.01	0.98 ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02	**0.96** ± 0.02
CUS		0.06 ± 0.18	0.15 ± 0.13	0.16 ± 0.11	0.16 ± 0.11	0.00 ± 0.00	0.29 ± 0.13	0.26 ± 0.10	0.25 ± 0.11	0.00 ± 0.00	0.42 ± 0.05	0.43 ± 0.03	0.45 ± 0.03
CUSG		0.95 ± 0.05	**0.94** ± 0.05	**0.94** ± 0.05	0.93 ± 0.05	0.72 ± 0.36	0.96 ± 0.02	0.96 ± 0.03	0.94 ± 0.03	0.98 ± 0.03	0.95 ± 0.06	0.95 ± 0.07	0.94 ± 0.06
TUS		0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.03 ± 0.01	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.42 ± 0.01	0.42 ± 0.01	0.44 ± 0.00
RTUS-1		0.94 ± 0.04	0.93 ± 0.04	0.93 ± 0.04	0.93 ± 0.04	**0.98** ± 0.01	**0.97** ± 0.01	0.96 ± 0.01	0.95 ± 0.01	0.95 ± 0.03	0.93 ± 0.03	0.92 ± 0.03	0.91 ± 0.03
RTUS-1s		**1.00** ± 0.00	**0.99** ± 0.00	**0.99** ± 0.00	**0.99** ± 0.00	0.80 ± 0.03	**0.97** ± 0.00	**0.98** ± 0.00	**0.97** ± 0.00	**0.99** ± 0.01	**0.98** ± 0.01	**0.99** ± 0.01	**0.98** ± 0.01
RTUS-5		0.94 ± 0.04	0.93 ± 0.04	0.93 ± 0.04	0.93 ± 0.04	**0.98** ± 0.01	**0.97** ± 0.01	0.97 ± 0.01	0.96 ± 0.01	0.94 ± 0.02	0.93 ± 0.02	0.92 ± 0.02	0.91 ± 0.02
RTUS-5s		0.59 ± 0.08	0.42 ± 0.01	0.49 ± 0.01	0.46 ± 0.01	**0.98** ± 0.03	0.59 ± 0.04	0.64 ± 0.01	0.64 ± 0.01	**1.00** ± 0.00	0.67 ± 0.00	0.63 ± 0.01	0.66 ± 0.00
STUSA-1		0.93 ± 0.05	0.92 ± 0.05	0.92 ± 0.05	0.92 ± 0.05	**0.98** ± 0.01	**0.97** ± 0.01	**0.98** ± 0.01	**0.97** ± 0.01	0.95 ± 0.03	0.94 ± 0.03	0.94 ± 0.03	0.92 ± 0.03
STUSA-1s		0.91 ± 0.06	0.91 ± 0.06	0.91 ± 0.06	0.90 ± 0.06	0.93 ± 0.02	0.93 ± 0.02	0.93 ± 0.02	0.92 ± 0.02	0.93 ± 0.03	0.92 ± 0.03	0.92 ± 0.03	0.90 ± 0.03
STUSA-5		0.92 ± 0.05	0.92 ± 0.05	0.92 ± 0.05	0.92 ± 0.05	**0.98** ± 0.01	**0.97** ± 0.01	0.97 ± 0.01	0.96 ± 0.01	0.94 ± 0.03	0.94 ± 0.03	0.93 ± 0.03	0.92 ± 0.03
STUSA-5s		0.91 ± 0.06	0.91 ± 0.06	0.91 ± 0.06	0.90 ± 0.06	0.93 ± 0.02	0.92 ± 0.02	0.93 ± 0.02	0.92 ± 0.01	0.93 ± 0.03	0.93 ± 0.02	0.92 ± 0.03	0.91 ± 0.03
STUSD-1		0.86 ± 0.05	0.85 ± 0.00	0.87 ± 0.00	0.87 ± 0.00	0.00 ± 0.00	0.93 ± 0.00	0.90 ± 0.00	0.87 ± 0.00	0.00 ± 0.00	0.87 ± 0.01	0.92 ± 0.00	0.90 ± 0.00
STUSD-1s		0.86 ± 0.04	0.85 ± 0.01	0.87 ± 0.00	0.87 ± 0.00	0.00 ± 0.00	0.93 ± 0.00	0.90 ± 0.00	0.87 ± 0.00	0.00 ± 0.00	0.87 ± 0.01	0.92 ± 0.00	0.90 ± 0.00
STUSD-5		0.94 ± 0.02	0.88 ± 0.00	0.89 ± 0.00	0.90 ± 0.00	0.00 ± 0.00	0.94 ± 0.00	0.92 ± 0.00	0.89 ± 0.00	0.07 ± 0.07	0.92 ± 0.00	0.96 ± 0.00	0.95 ± 0.00
STUSD-5s		0.94 ± 0.02	0.88 ± 0.00	0.89 ± 0.00	0.89 ± 0.00	0.00 ± 0.00	0.94 ± 0.00	0.91 ± 0.00	0.88 ± 0.00	0.00 ± 0.00	0.91 ± 0.01	0.95 ± 0.00	0.94 ± 0.00

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

Under-sampling the trajectories to the support vectors improves RTUS-1 substantially in terms of AUPR, F-measure and G-mean. However, RTUS-5 degrade except for RTUS-5 in terms

of F-measure and G-mean. The performance of STUSD-1, and STUSD-5 remain the same after under-sampling the trajectories to the support vectors and the performance of STUSA-1 and STUSA-5 go down slightly. The decision boundary of classifiers in RTUS-5 change more abruptly than RTUS-1, and most of the classifiers in the ensemble are trained on higher imbalance levels which means there is higher bias of performance in base classifiers towards the non-target class. Therefore, under-sampling the trajectories to the support vectors is useful when the SVMs are accurate enough to select the proper support vectors.

In Table 3.2, when the overlap between classes is lower (D_1) CUS, TUS, RTUS-1s, and STUSD-1outperform the others in terms of AUPR. For the highly imbalanced data with high level of overlap, STUSD-1, STUSD-1s, STUSD-5, and STUSD-5s perform the best in terms of AUPR. In Table 3.3, STUSA-1, STUSA-1s, STUSA-5, and STUSA-5s outperform the others with all three cases of data complexities (D_1, D_2, and D_3) in terms of F-measure. RTUS-1s outperforms the others substantially when D_3 (lower imbalance level of training data and lower overlap between classes) is used for the experiments. In Table 3.4, RUSG, RUSwRG, and RTUS-1s outperform the other methods in terms of G-mean. STUSA-1 outperforms the other methods when the overlap between classes is higher and RUSwR outperform the other when the overlap between classes is lower.

3.4.2 Results of experiments with video datasets

Tables 3.5, 3.6 and 3.7 show the results of experiments on the FIA video dataset and Tables 3.8, 3.9 and 3.10 show the results of experiments on the COX video dataset . In Table 3.5, it is observed that RUSBoost, SeRn-SVM, and CUS are outperformed by all the other ensembles in terms of AUPR. However, the results of experiments on the FIA video dataset in terms of the AUPR in Table 3.8 show that RUSG, RUSwRG, STUSA-1, STUSA-1s, STUSA-5 and STUSA-5s outperform the other methods over all ranges of imbalance during design and testing.

Table 3.5 Average of AUPR performance of baseline and proposed techniques on FIA data set over different levels of skew in training and test data.

Ensembles	**Train Data**	Λ_{train}=1:50				Λ_{train}=1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
RUSBoost		0.98 ± 0.01	0.96 ± 0.02	0.95 ± 0.03	0.94 ± 0.03	0.98 ± 0.01	0.95 ± 0.04	0.94 ± 0.05	0.92 ± 0.05
Seg-EoSVMs		0.98 ± 0.01	0.96 ± 0.02	0.93 ± 0.05	0.91 ± 0.06	0.98 ± 0.01	0.95 ± 0.03	0.93 ± 0.04	0.91 ± 0.05
RUS		0.98 ± 0.01	0.96 ± 0.02	0.95 ± 0.03	0.94 ± 0.03	0.98 ± 0.01	0.96 ± 0.03	0.95 ± 0.03	0.93 ± 0.04
RUSwR		0.98 ± 0.01	0.97 ± 0.02	0.95 ± 0.02	0.95 ± 0.03	0.98 ± 0.01	0.96 ± 0.02	0.95 ± 0.03	0.93 ± 0.04
RUSG		0.98 ± 0.01	0.97 ± 0.02	0.96 ± 0.02	0.96 ± 0.02	0.98 ± 0.01	0.97 ± 0.02	0.96 ± 0.02	0.94 ± 0.03
RUSwRG		0.98 ± 0.01	0.97 ± 0.01	0.96 ± 0.02	0.96 ± 0.02	0.98 ± 0.01	0.97 ± 0.02	0.96 ± 0.02	0.94 ± 0.03
CUS		0.97 ± 0.05	0.95 ± 0.11	0.93 ± 0.13	0.92 ± 0.13	0.98 ± 0.01	0.96 ± 0.05	0.95 ± 0.06	0.93 ± 0.08
CUSG		0.97 ± 0.04	0.96 ± 0.08	0.95 ± 0.08	0.94 ± 0.08	0.98 ± 0.01	0.97 ± 0.02	0.96 ± 0.02	0.94 ± 0.03
TUS		0.98 ± 0.01	0.97 ± 0.02	0.96 ± 0.02	0.95 ± 0.03	0.98 ± 0.01	0.96 ± 0.02	0.95 ± 0.03	0.94 ± 0.04
RTUS-1		0.98 ± 0.01	0.97 ± 0.01	0.96 ± 0.02	0.95 ± 0.03	0.98 ± 0.01	0.97 ± 0.01	0.95 ± 0.03	0.94 ± 0.03
RTUS-1s		0.98 ± 0.01	0.97 ± 0.01	0.96 ± 0.02	0.95 ± 0.03	0.98 ± 0.01	0.97 ± 0.01	0.96 ± 0.02	0.94 ± 0.03
RTUS-5		0.98 ± 0.01	0.97 ± 0.01	0.96 ± 0.02	0.95 ± 0.03	0.98 ± 0.01	0.97 ± 0.01	0.95 ± 0.02	0.94 ± 0.03
RTUS-5s		0.98 ± 0.01	0.96 ± 0.02	0.95 ± 0.03	0.94 ± 0.03	0.98 ± 0.01	0.96 ± 0.03	0.94 ± 0.05	0.92 ± 0.05
STUSA-1		0.98 ± 0.01	0.97 ± 0.01	0.96 ± 0.02	0.95 ± 0.02	0.98 ± 0.01	0.97 ± 0.02	0.96 ± 0.02	0.94 ± 0.03
STUSA-1s		0.98 ± 0.01	0.97 ± 0.01	0.96 ± 0.02	0.96 ± 0.02	0.98 ± 0.01	0.97 ± 0.02	0.96 ± 0.02	0.94 ± 0.03
STUSA-5		0.98 ± 0.01	0.97 ± 0.01	0.96 ± 0.02	0.95 ± 0.02	0.98 ± 0.01	0.97 ± 0.02	0.96 ± 0.02	0.94 ± 0.03
STUSA-5s		0.98 ± 0.01	0.97 ± 0.01	0.96 ± 0.02	0.96 ± 0.02	0.98 ± 0.01	0.97 ± 0.02	0.96 ± 0.02	0.94 ± 0.03
STUSD-1		0.98 ± 0.01	0.96 ± 0.02	0.95 ± 0.03	0.94 ± 0.03	0.98 ± 0.01	0.96 ± 0.03	0.95 ± 0.03	0.93 ± 0.04
STUSD-1s		0.98 ± 0.01	0.96 ± 0.02	0.95 ± 0.03	0.94 ± 0.03	0.98 ± 0.01	0.96 ± 0.03	0.95 ± 0.03	0.93 ± 0.04
STUSD-5		0.98 ± 0.01	0.96 ± 0.02	0.95 ± 0.03	0.94 ± 0.03	0.98 ± 0.01	0.96 ± 0.02	0.95 ± 0.03	0.93 ± 0.04
STUSD-5s		0.98 ± 0.01	0.96 ± 0.02	0.95 ± 0.03	0.95 ± 0.03	0.98 ± 0.01	0.96 ± 0.02	0.95 ± 0.03	0.93 ± 0.04

It is observed that modifying RUS, RUSwR, CUS and TUS using training data subsets with different skew levels improves their performance in terms of F-measure substantially. Modifying RUS and RUSwR does not improve their performance in terms of G-mean. However, the performance of CUS and TUS improve substantially in terms of G-mean.

Table 3.6 Average of F_2-measure performance of baseline and proposed techniques on FIA data set over different levels of skew in training and test data.

Ensembles	Train Data	Λ_{train}=1:50				Λ_{train}=1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
RUSBoost		**0.98** ± 0.02	0.89 ± 0.08	0.76 ± 0.16	0.65 ± 0.18	**0.98** ± 0.02	0.86 ± 0.10	0.71 ± 0.15	0.57 ± 0.17
Seg-EoSVMs		**0.99** ± 0.01	0.69 ± 0.16	0.48 ± 0.21	0.32 ± 0.21	**0.99** ± 0.02	0.68 ± 0.15	0.45 ± 0.18	0.31 ± 0.18
RUS		**0.98** ± 0.02	0.88 ± 0.08	0.73 ± 0.16	0.61 ± 0.18	**0.98** ± 0.02	0.85 ± 0.09	0.70 ± 0.14	0.54 ± 0.15
RUSwR		**0.98** ± 0.02	0.92 ± 0.05	0.83 ± 0.11	0.74 ± 0.12	**0.98** ± 0.02	0.90 ± 0.08	0.81 ± 0.12	0.67 ± 0.14
RUSG		0.96 ± 0.03	**0.95** ± 0.03	**0.95** ± 0.03	**0.94** ± 0.04	0.96 ± 0.04	**0.95** ± 0.04	**0.94** ± 0.04	0.89 ± 0.06
RUSwRG		0.96 ± 0.03	**0.95** ± 0.03	**0.95** ± 0.03	**0.94** ± 0.03	0.95 ± 0.04	**0.95** ± 0.04	**0.94** ± 0.04	**0.90** ± 0.05
CUS		0.89 ± 0.07	0.32 ± 0.18	0.18 ± 0.15	0.10 ± 0.09	0.89 ± 0.06	0.30 ± 0.17	0.16 ± 0.14	0.09 ± 0.09
CUSG		0.96 ± 0.02	**0.95** ± 0.03	**0.94** ± 0.04	**0.93** ± 0.04	0.97 ± 0.03	**0.95** ± 0.03	**0.93** ± 0.05	0.87 ± 0.08
TUS		0.90 ± 0.06	0.32 ± 0.20	0.18 ± 0.17	0.10 ± 0.10	0.90 ± 0.06	0.30 ± 0.17	0.16 ± 0.15	0.09 ± 0.09
RTUS-1		0.96 ± 0.03	**0.96** ± 0.03	**0.95** ± 0.03	0.92 ± 0.05	0.97 ± 0.03	**0.96** ± 0.03	0.92 ± 0.05	0.85 ± 0.08
RTUS-1s		0.94 ± 0.04	0.94 ± 0.04	**0.94** ± 0.04	**0.93** ± 0.04	0.95 ± 0.04	0.94 ± 0.04	**0.93** ± 0.05	**0.91** ± 0.06
RTUS-5		0.96 ± 0.03	**0.96** ± 0.03	**0.95** ± 0.03	**0.93** ± 0.04	0.96 ± 0.03	**0.96** ± 0.04	0.92 ± 0.05	0.87 ± 0.07
RTUS-5s		0.95 ± 0.06	0.48 ± 0.24	0.30 ± 0.24	0.19 ± 0.20	0.96 ± 0.05	0.46 ± 0.19	0.27 ± 0.19	0.15 ± 0.14
STUSA-1		0.96 ± 0.03	**0.95** ± 0.03	**0.95** ± 0.03	**0.93** ± 0.04	0.96 ± 0.04	**0.95** ± 0.04	**0.93** ± 0.05	0.87 ± 0.07
STUSA-1s		0.96 ± 0.03	**0.95** ± 0.03	**0.95** ± 0.03	**0.94** ± 0.04	0.96 ± 0.04	**0.95** ± 0.04	**0.93** ± 0.05	0.88 ± 0.07
STUSA-5		0.96 ± 0.03	**0.95** ± 0.03	**0.95** ± 0.03	**0.93** ± 0.04	0.96 ± 0.04	**0.95** ± 0.04	**0.93** ± 0.05	0.88 ± 0.07
STUSA-5s		0.96 ± 0.03	**0.95** ± 0.03	**0.95** ± 0.03	**0.94** ± 0.04	0.96 ± 0.04	**0.95** ± 0.04	**0.93** ± 0.05	0.89 ± 0.06
STUSD-1		0.97 ± 0.02	0.93 ± 0.05	0.86 ± 0.12	0.82 ± 0.13	0.97 ± 0.03	0.89 ± 0.09	0.80 ± 0.14	0.70 ± 0.17
STUSD-1s		0.97 ± 0.02	0.93 ± 0.05	0.88 ± 0.11	0.85 ± 0.12	0.97 ± 0.03	0.90 ± 0.08	0.81 ± 0.13	0.73 ± 0.16
STUSD-5		0.97 ± 0.02	0.93 ± 0.05	0.87 ± 0.11	0.83 ± 0.12	0.97 ± 0.03	0.91 ± 0.08	0.83 ± 0.12	0.74 ± 0.15
STUSD-5s		0.97 ± 0.02	0.94 ± 0.04	0.89 ± 0.10	0.86 ± 0.11	0.97 ± 0.03	0.91 ± 0.08	0.84 ± 0.11	0.77 ± 0.14

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

The performance of TUS ensemble is very similar to that of CUS ensemble and both outperform RUS and RUSwR in terms of AUPR. However, the TUS and CUS ensembles degrade in

Table 3.7 Average of G-mean performance of baseline and proposed techniques on FIA data set over different levels of skew in training and test data.

Ensembles	**Train Data**	Λ_{train}=1:50				Λ_{train}=1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
RUSBoost		0.84 ± 0.34	0.84 ± 0.34	0.84 ± 0.34	0.84 ± 0.34	0.89 ± 0.25	0.91 ± 0.24	0.90 ± 0.24	0.90 ± 0.24
Seg-EoSVMs		**0.98** ± 0.03	0.93 ± 0.05	0.92 ± 0.07	0.92 ± 0.07	**0.97** ± 0.06	0.93 ± 0.05	0.92 ± 0.06	0.92 ± 0.06
RUS		**0.97** ± 0.04	**0.97** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02	0.95 ± 0.08	**0.97** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02
RUSwR		**0.98** ± 0.03	**0.98** ± 0.02	**0.98** ± 0.02	**0.98** ± 0.02	**0.98** ± 0.03	**0.98** ± 0.02	**0.98** ± 0.02	**0.98** ± 0.02
RUSG		0.96 ± 0.02	0.96 ± 0.02	0.96 ± 0.02	0.96 ± 0.02	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03
RUSwRG		0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.95 ± 0.04	0.96 ± 0.04	0.96 ± 0.04	0.95 ± 0.04
CUS		0.50 ± 0.38	0.55 ± 0.27	0.55 ± 0.27	0.55 ± 0.26	0.53 ± 0.36	0.54 ± 0.25	0.54 ± 0.25	0.53 ± 0.24
CUSG		0.95 ± 0.14	0.95 ± 0.14	0.95 ± 0.14	0.95 ± 0.14	**0.97** ± 0.03	**0.97** ± 0.03	**0.97** ± 0.03	**0.97** ± 0.03
TUS		0.52 ± 0.38	0.54 ± 0.28	0.54 ± 0.27	0.54 ± 0.26	0.55 ± 0.38	0.54 ± 0.26	0.53 ± 0.25	0.53 ± 0.25
RTUS-1		**0.97** ± 0.02	**0.97** ± 0.03	**0.97** ± 0.03	0.96 ± 0.03	**0.97** ± 0.03	**0.97** ± 0.03	0.96 ± 0.03	0.96 ± 0.03
RTUS-1s		0.94 ± 0.04	0.94 ± 0.04	0.94 ± 0.04	0.94 ± 0.04	0.95 ± 0.04	0.95 ± 0.04	0.95 ± 0.04	0.95 ± 0.04
RTUS-5		0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03
RTUS-5s		0.74 ± 0.38	0.71 ± 0.29	0.70 ± 0.29	0.70 ± 0.28	0.79 ± 0.32	0.75 ± 0.22	0.74 ± 0.23	0.72 ± 0.24
STUSA-1		0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.04	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03
STUSA-1s		0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.04	0.96 ± 0.04	0.96 ± 0.04	0.96 ± 0.04
STUSA-5		0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.04	0.96 ± 0.04	0.96 ± 0.04	0.96 ± 0.04
STUSA-5s		0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.04	0.96 ± 0.04	0.96 ± 0.04	0.96 ± 0.04
STUSD-1		**0.97** ± 0.03	**0.97** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02	0.95 ± 0.08	**0.97** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02
STUSD-1s		**0.97** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02	0.95 ± 0.07	**0.97** ± 0.03	0.96 ± 0.03	**0.97** ± 0.03
STUSD-5		**0.97** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02	0.95 ± 0.06	**0.97** ± 0.03	**0.97** ± 0.03	**0.97** ± 0.03
STUSD-5s		**0.97** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02	0.95 ± 0.06	0.96 ± 0.03	0.96 ± 0.03	**0.97** ± 0.03

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

performance in terms of F-measure and G-mean. These ensembles are better used by classification score combination followed by threshold optimization rather than decision combination.

Table 3.8 Average of AUPR performance of baseline and proposed techniques on COX data set over different levels of skew in training and test data.

Ensembles	**Train Data**	Λ_{train}=1:50				Λ_{train}=1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
RUSBoost		0.90 ± 0.04	0.80 ± 0.11	0.79 ± 0.12	0.78 ± 0.13	0.90 ± 0.03	0.81 ± 0.13	0.79 ± 0.13	0.78 ± 0.15
Seg-EoSVMs		0.91 ± 0.02	0.81 ± 0.10	0.76 ± 0.13	0.72 ± 0.17	0.91 ± 0.01	0.83 ± 0.11	0.77 ± 0.14	0.73 ± 0.17
RUS		0.90 ± 0.04	0.82 ± 0.11	0.81 ± 0.11	0.80 ± 0.12	0.90 ± 0.03	0.83 ± 0.11	0.82 ± 0.11	0.81 ± 0.13
RUSwR		0.90 ± 0.04	0.82 ± 0.11	0.81 ± 0.12	0.80 ± 0.12	0.90 ± 0.03	0.83 ± 0.11	0.82 ± 0.12	0.80 ± 0.13
RUSG		0.90 ± 0.03	0.85 ± 0.08	**0.85** ± 0.09	**0.84** ± 0.09	0.91 ± 0.02	0.86 ± 0.08	**0.85** ± 0.09	**0.82** ± 0.12
RUSwRG		0.90 ± 0.03	**0.86** ± 0.08	**0.85** ± 0.09	**0.85** ± 0.09	0.91 ± 0.02	0.86 ± 0.08	**0.85** ± 0.09	**0.82** ± 0.12
CUS		0.90 ± 0.05	0.82 ± 0.12	0.81 ± 0.12	0.81 ± 0.12	0.90 ± 0.04	0.82 ± 0.14	0.81 ± 0.15	0.79 ± 0.16
CUSG		0.91 ± 0.02	0.85 ± 0.08	**0.84** ± 0.09	**0.84** ± 0.09	0.91 ± 0.03	0.84 ± 0.13	0.83 ± 0.14	0.80 ± 0.16
TUS		0.90 ± 0.05	0.84 ± 0.09	0.83 ± 0.10	0.82 ± 0.10	0.90 ± 0.03	0.84 ± 0.09	0.84 ± 0.10	**0.82** ± 0.12
RTUS-1		0.91 ± 0.02	**0.86** ± 0.07	**0.84** ± 0.09	0.82 ± 0.11	0.91 ± 0.01	**0.87** ± 0.08	0.83 ± 0.10	0.79 ± 0.13
RTUS-1s		0.91 ± 0.02	**0.87** ± 0.07	**0.85** ± 0.08	0.83 ± 0.10	0.92 ± 0.01	**0.88** ± 0.06	**0.85** ± 0.07	**0.82** ± 0.11
RTUS-5		0.91 ± 0.02	**0.86** ± 0.07	**0.84** ± 0.09	0.82 ± 0.11	0.91 ± 0.01	**0.87** ± 0.07	0.84 ± 0.10	0.80 ± 0.13
RTUS-5s		0.91 ± 0.02	0.80 ± 0.13	0.78 ± 0.14	0.76 ± 0.16	0.91 ± 0.02	0.81 ± 0.14	0.78 ± 0.16	0.76 ± 0.18
STUSA-1		0.91 ± 0.03	**0.86** ± 0.09	**0.85** ± 0.09	**0.84** ± 0.10	0.91 ± 0.02	0.86 ± 0.07	**0.85** ± 0.08	**0.82** ± 0.12
STUSA-1s		0.91 ± 0.03	**0.86** ± 0.09	**0.84** ± 0.09	**0.84** ± 0.10	0.91 ± 0.02	0.86 ± 0.07	**0.85** ± 0.08	**0.82** ± 0.12
STUSA-5		0.91 ± 0.03	**0.86** ± 0.09	**0.85** ± 0.09	**0.84** ± 0.10	0.91 ± 0.02	**0.87** ± 0.07	**0.85** ± 0.08	**0.82** ± 0.12
STUSA-5s		0.91 ± 0.03	**0.86** ± 0.08	**0.85** ± 0.09	**0.84** ± 0.10	0.91 ± 0.02	**0.87** ± 0.07	**0.86** ± 0.08	**0.82** ± 0.12
STUSD-1		0.90 ± 0.04	0.82 ± 0.11	0.81 ± 0.12	0.80 ± 0.12	0.90 ± 0.03	0.84 ± 0.10	0.83 ± 0.11	**0.81** ± 0.13
STUSD-1s		0.90 ± 0.04	0.82 ± 0.11	0.81 ± 0.12	0.81 ± 0.12	0.90 ± 0.03	0.84 ± 0.10	0.83 ± 0.11	**0.81** ± 0.12
STUSD-5		0.90 ± 0.04	0.82 ± 0.11	0.81 ± 0.11	0.81 ± 0.12	0.90 ± 0.03	0.84 ± 0.10	0.83 ± 0.11	**0.81** ± 0.12
STUSD-5s		0.90 ± 0.04	0.82 ± 0.11	0.81 ± 0.11	0.81 ± 0.12	0.90 ± 0.03	0.84 ± 0.10	0.84 ± 0.10	**0.81** ± 0.12

By increasing the difference of imbalance between classifiers from one to five, the performance of RTUS, STUSA and STUSD ensembles improve or stay the same in terms of AUPR, F-measure and G-mean.

Table 3.9 Average of F_2-measure performance of baseline and proposed techniques on COX data set over different levels of skew in training and test data.

Ensembles	**Train Data**	Λ_{train}=1:50				Λ_{train}=1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
RUSBoost		0.89 ± 0.10	0.41 ± 0.17	0.36 ± 0.18	0.33 ± 0.20	0.87 ± 0.12	0.42 ± 0.22	0.36 ± 0.23	0.29 ± 0.25
Seg-EoSVMs		**0.95** ± 0.06	0.62 ± 0.16	0.45 ± 0.19	0.34 ± 0.20	**0.95** ± 0.07	0.64 ± 0.18	0.46 ± 0.19	0.34 ± 0.20
RUS		0.90 ± 0.10	0.46 ± 0.17	0.41 ± 0.17	0.38 ± 0.18	0.91 ± 0.10	0.52 ± 0.20	0.45 ± 0.20	0.37 ± 0.23
RUSwR		0.90 ± 0.10	0.47 ± 0.17	0.42 ± 0.17	0.39 ± 0.18	0.90 ± 0.10	0.53 ± 0.20	0.46 ± 0.20	0.38 ± 0.23
RUSG		0.92 ± 0.08	**0.90** ± 0.09	**0.90** ± 0.09	**0.89** ± 0.09	0.92 ± 0.09	**0.90** ± 0.10	**0.89** ± 0.10	**0.80** ± 0.18
RUSwRG		0.92 ± 0.08	**0.91** ± 0.08	**0.90** ± 0.09	**0.90** ± 0.09	0.92 ± 0.09	**0.91** ± 0.09	**0.90** ± 0.09	**0.82** ± 0.17
CUS		0.68 ± 0.03	0.11 ± 0.03	0.05 ± 0.02	0.03 ± 0.02	0.68 ± 0.03	0.11 ± 0.03	0.06 ± 0.02	0.03 ± 0.01
CUSG		0.92 ± 0.09	0.85 ± 0.14	0.84 ± 0.15	0.83 ± 0.16	0.92 ± 0.09	0.84 ± 0.16	0.82 ± 0.18	0.72 ± 0.24
TUS		0.68 ± 0.04	0.11 ± 0.04	0.06 ± 0.03	0.04 ± 0.02	0.69 ± 0.06	0.12 ± 0.05	0.07 ± 0.04	0.04 ± 0.03
RTUS-1		0.92 ± 0.08	**0.91** ± 0.09	**0.90** ± 0.09	0.84 ± 0.16	0.93 ± 0.08	**0.91** ± 0.09	0.80 ± 0.15	0.69 ± 0.26
RTUS-1s		0.85 ± 0.12	0.84 ± 0.12	0.84 ± 0.12	0.84 ± 0.13	0.87 ± 0.11	0.87 ± 0.11	0.84 ± 0.11	**0.80** ± 0.18
RTUS-5		0.92 ± 0.08	**0.91** ± 0.09	**0.90** ± 0.09	0.85 ± 0.13	0.92 ± 0.08	**0.91** ± 0.09	0.80 ± 0.15	0.70 ± 0.25
RTUS-5s		**0.95** ± 0.06	0.35 ± 0.15	0.23 ± 0.13	0.17 ± 0.13	**0.96** ± 0.05	0.35 ± 0.15	0.23 ± 0.12	0.16 ± 0.13
STUSA-1		0.92 ± 0.08	0.89 ± 0.09	0.88 ± 0.10	0.88 ± 0.10	0.92 ± 0.09	0.89 ± 0.10	0.88 ± 0.10	**0.80** ± 0.17
STUSA-1s		0.92 ± 0.08	0.89 ± 0.09	**0.89** ± 0.09	**0.89** ± 0.10	0.92 ± 0.09	0.89 ± 0.10	0.88 ± 0.10	**0.80** ± 0.17
STUSA-5		0.91 ± 0.08	0.89 ± 0.09	**0.89** ± 0.09	**0.89** ± 0.09	0.92 ± 0.09	0.89 ± 0.10	**0.89** ± 0.10	**0.82** ± 0.16
STUSA-5s		0.92 ± 0.08	0.89 ± 0.09	**0.89** ± 0.09	**0.89** ± 0.09	0.92 ± 0.09	**0.90** ± 0.09	**0.89** ± 0.09	**0.82** ± 0.17
STUSD-1		**0.94** ± 0.07	0.67 ± 0.23	0.63 ± 0.25	0.60 ± 0.26	0.93 ± 0.08	0.64 ± 0.24	0.58 ± 0.26	0.48 ± 0.30
STUSD-1s		**0.94** ± 0.07	0.69 ± 0.23	0.65 ± 0.24	0.62 ± 0.26	0.93 ± 0.08	0.67 ± 0.24	0.61 ± 0.26	0.52 ± 0.31
STUSD-5		**0.94** ± 0.07	0.70 ± 0.23	0.66 ± 0.24	0.63 ± 0.26	0.93 ± 0.08	0.71 ± 0.22	0.66 ± 0.23	0.56 ± 0.29
STUSD-5s		**0.94** ± 0.07	0.71 ± 0.22	0.67 ± 0.24	0.65 ± 0.25	0.93 ± 0.08	0.71 ± 0.22	0.66 ± 0.24	0.56 ± 0.30

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

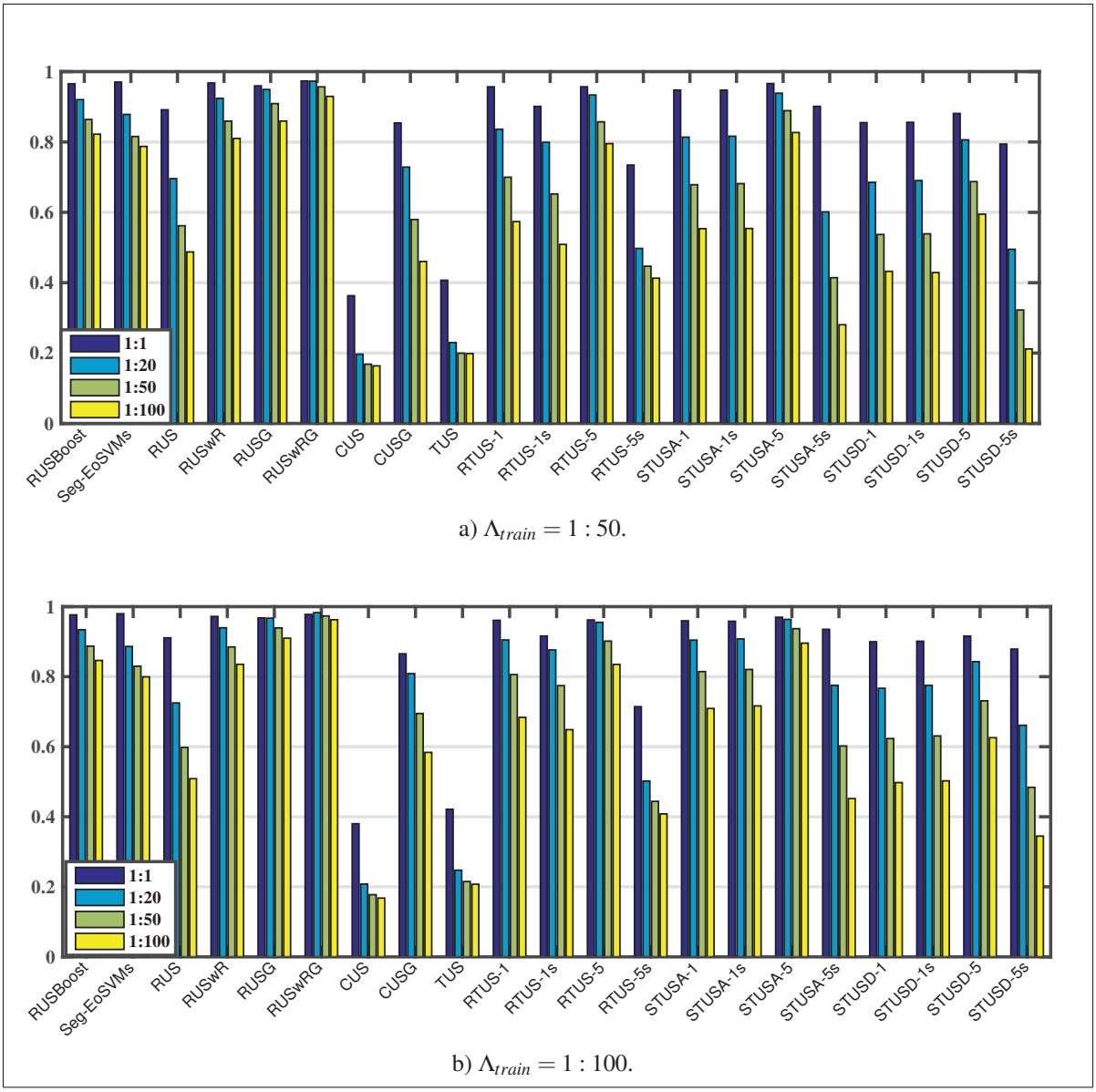
Under-sampling the trajectories to the support vectors improves RTUS-1 in terms of F-measure, but not in terms of G-mean. The performance of RTUS-5 degrades similarly to the experiments

Table 3.10 Average of G-mean performance of baseline and proposed techniques on COX data set over different levels of skew in training and test data.

Ensembles	**Train Data**	Λ_{train}=1:50				Λ_{train}=1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
RUSBoost		0.68 ± 0.38	0.71 ± 0.36	0.74 ± 0.38	0.76 ± 0.38	0.76 ± 0.30	0.84 ± 0.19	0.88 ± 0.19	0.89 ± 0.19
Seg-EoSVMs		**0.95** ± 0.06	**0.93** ± 0.05	**0.93** ± 0.06	0.92 ± 0.07	**0.95** ± 0.07	**0.93** ± 0.07	0.92 ± 0.07	0.92 ± 0.07
RUS		0.87 ± 0.17	0.90 ± 0.06	**0.94** ± 0.05	**0.95** ± 0.05	0.88 ± 0.15	0.92 ± 0.06	**0.94** ± 0.05	**0.95** ± 0.05
RUSwR		0.87 ± 0.16	0.91 ± 0.06	**0.94** ± 0.05	**0.95** ± 0.05	0.88 ± 0.15	0.91 ± 0.07	**0.94** ± 0.06	**0.94** ± 0.06
RUSG		0.93 ± 0.07	**0.93** ± 0.07	**0.93** ± 0.07	0.93 ± 0.07	0.93 ± 0.08	**0.93** ± 0.08	0.93 ± 0.08	0.93 ± 0.08
RUSwRG		0.93 ± 0.07	**0.93** ± 0.07	**0.93** ± 0.07	0.93 ± 0.07	0.93 ± 0.08	**0.93** ± 0.08	0.93 ± 0.08	0.93 ± 0.08
CUS		0.13 ± 0.20	0.27 ± 0.25	0.35 ± 0.28	0.44 ± 0.28	0.11 ± 0.20	0.25 ± 0.24	0.39 ± 0.29	0.35 ± 0.27
CUSG		0.93 ± 0.08	**0.93** ± 0.08	**0.93** ± 0.08	0.93 ± 0.08	0.90 ± 0.17	0.90 ± 0.17	0.91 ± 0.17	0.91 ± 0.17
TUS		0.18 ± 0.23	0.34 ± 0.25	0.44 ± 0.27	0.54 ± 0.25	0.18 ± 0.26	0.40 ± 0.24	0.56 ± 0.23	0.53 ± 0.23
RTUS-1		0.92 ± 0.07	0.92 ± 0.07	0.92 ± 0.07	0.92 ± 0.07	0.93 ± 0.08	**0.93** ± 0.08	0.93 ± 0.07	0.92 ± 0.08
RTUS-1s		0.86 ± 0.10	0.86 ± 0.10	0.86 ± 0.10	0.86 ± 0.10	0.88 ± 0.09	0.88 ± 0.09	0.88 ± 0.09	0.88 ± 0.09
RTUS-5		0.92 ± 0.07	0.92 ± 0.07	0.92 ± 0.07	0.92 ± 0.07	0.93 ± 0.08	**0.93** ± 0.08	0.92 ± 0.07	0.92 ± 0.08
RTUS-5s		**0.95** ± 0.06	0.85 ± 0.08	0.88 ± 0.08	0.89 ± 0.07	**0.96** ± 0.05	0.86 ± 0.08	0.89 ± 0.07	0.88 ± 0.08
STUSA-1		0.92 ± 0.08	0.92 ± 0.07	**0.93** ± 0.07	0.93 ± 0.07	0.92 ± 0.08	0.92 ± 0.08	0.92 ± 0.08	0.92 ± 0.08
STUSA-1s		0.93 ± 0.08	**0.93** ± 0.07	**0.93** ± 0.07	0.93 ± 0.07	0.92 ± 0.08	0.92 ± 0.08	0.93 ± 0.08	0.93 ± 0.08
STUSA-5		0.92 ± 0.08	0.92 ± 0.07	0.92 ± 0.07	0.92 ± 0.07	0.92 ± 0.08	0.92 ± 0.08	0.92 ± 0.08	0.92 ± 0.08
STUSA-5s		0.92 ± 0.08	**0.93** ± 0.07	**0.93** ± 0.07	0.93 ± 0.07	0.92 ± 0.08	0.92 ± 0.08	0.92 ± 0.08	0.92 ± 0.08
STUSD-1		**0.94** ± 0.07	**0.93** ± 0.06	**0.94** ± 0.06	**0.95** ± 0.06	0.92 ± 0.10	**0.93** ± 0.07	**0.94** ± 0.06	**0.94** ± 0.06
STUSD-1s		**0.94** ± 0.07	**0.93** ± 0.06	**0.94** ± 0.06	**0.95** ± 0.06	0.92 ± 0.09	**0.93** ± 0.06	**0.94** ± 0.06	**0.94** ± 0.06
STUSD-5		**0.94** ± 0.07	**0.93** ± 0.06	**0.94** ± 0.06	**0.95** ± 0.06	0.93 ± 0.09	**0.94** ± 0.07	**0.95** ± 0.06	**0.94** ± 0.07
STUSD-5s		**0.94** ± 0.07	**0.93** ± 0.06	**0.94** ± 0.06	**0.95** ± 0.06	0.93 ± 0.08	**0.93** ± 0.06	**0.95** ± 0.06	**0.94** ± 0.06

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

with the synthetic data. The performance of STUSD-1, STUSD-5, STUSA-1 and STUSA-5 either improve or stay the same.

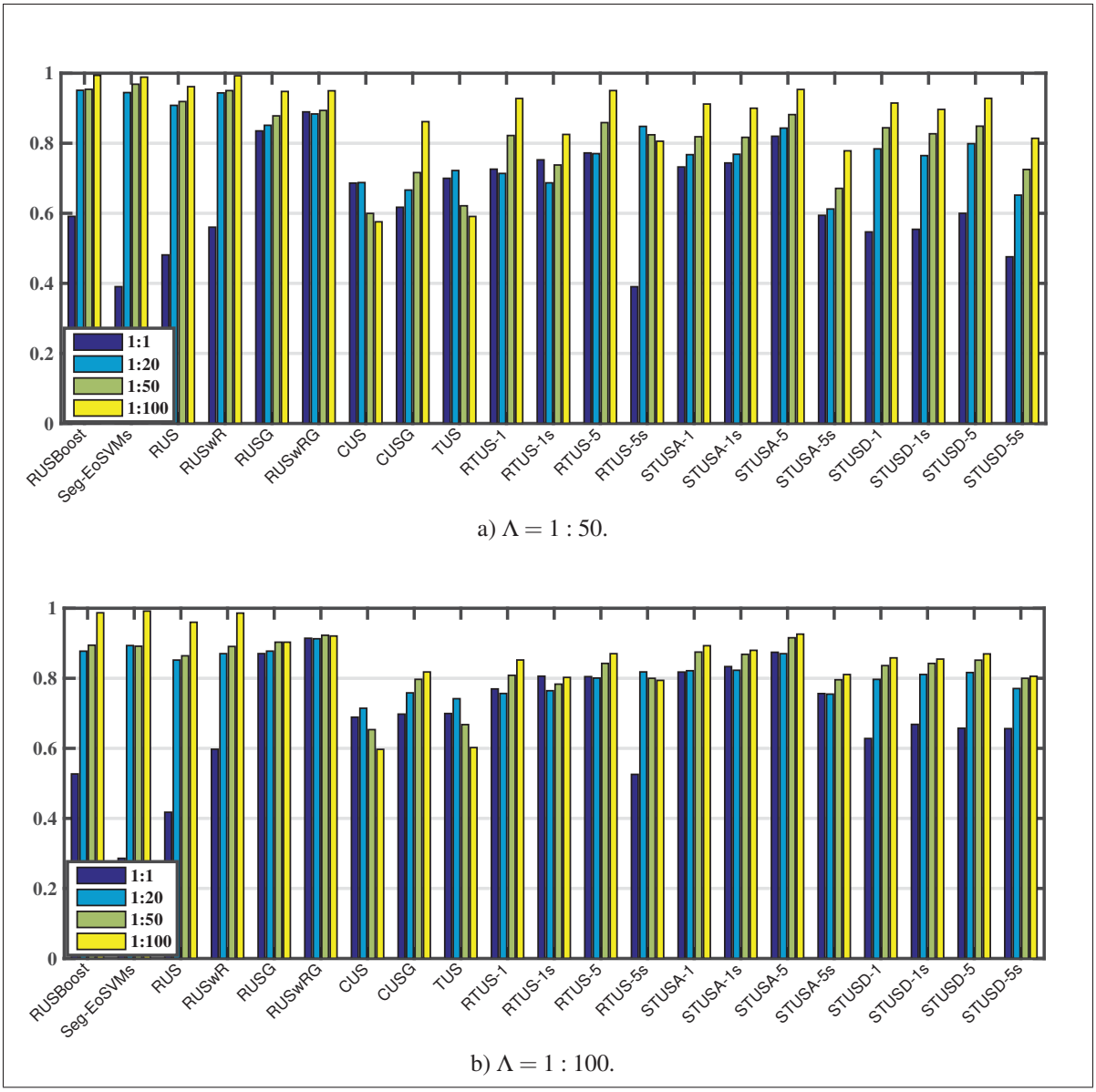


a) $\Lambda_{train} = 1:50$.

b) $\Lambda_{train} = 1:100$.

Figure 3.4 Average of diversity of proposed and baseline techniques in terms of kappa measure on the FIA video dataset.

Figures 3.4 and 3.5, show the diversity among classifiers of the baseline and proposed ensembles in terms of kappa measure and Q-statistics when the classifiers are trained on two different imbalance levels. Diversity between the classifiers of the CUS ensemble is the greatest in terms of both kappa and Q measures. Growing imbalance in RUS and RUSwR either increase the diversity or keeps it the same. However, growing imbalance in CUS and TUS decrease the



a) $\Lambda = 1:50$.

b) $\Lambda = 1:100$.

Figure 3.5 Average of diversity of proposed and baseline techniques in terms of Q-statistics on the FIA video dataset.

value of Q and kappa measures except for variations of STUSD that show greater diversity. As explained in section 3.3, increase in Q and kappa measures could be the result of improving the accuracy of classifiers in an ensemble. Since the accuracy of both CUS and TUS ensembles improve in Tables 3.5, 3.6 and 3.7, it is justified that growing imbalance in CUS and TUS does not decrease the diversity. In variations of TUS ensembles increasing the difference of

imbalance between training data subsets reduce the diversity. Under-sampling the trajectories to support vectors either increase the diversity or keeps it the same in all variations of TUS ensembles.

In summary, the experiments on both synthetic and video data sets show that growing imbalance in training classifiers in the ensemble improves the performance. Trajectory under-sampling in this application improves the accuracy-diversity trade-off between classifiers compared to the general-purposed sampling methods and the performance of the proposed ensembles either improve or stay the same by reducing the trajectories to support vectors. The proposed STUSA outperforms the state of the art methods in terms of F-measure. It is worth mentioning that, analogous to any other face classification system, the capture conditions like illumination and pose as well as the accuracy of the extracted ROIs (if the face is fully included and well fitted in the ROI) affects the quality of the feature vectors (how well they represent the face) and therefore the performance of the proposed classification system designed by them.The accuracy of the tracking module can affect the performance of the proposed ensembles only if the positive class trajectory contains samples from the negative class or the negative class trajectories contain samples from the positive class.

3.5 Conclusion

In this chapter, a novel technique was proposed for the design of individual-specific ensembles to address the class imbalance problem in person re-identification applications. In ensembles with trajectory under-sampling, training subsets contain samples from target trajectory and a growing selection of samples from non-target trajectories to minimize the risk of information loss. Instead of using general-purpose under-sampling techniques such as random or cluster-based under-sampling, contextual information (i.e., trajectory structure) is exploited to under-sample from an abundance of non-target data to design diverse ensembles of 2-class classifiers. The proposed under-sampling technique can be applicable in any multiclass classification application where the data consists of sub-clusters inherently and the one-versus-all strategy is used. Starting from one target and non-target trajectory for the first subset, the level of im-

balance and decision bound complexity is increased for the next subsets by adding non-target trajectories to the previous ones. Variants of these ensembles can give more importance to the most efficient classifiers in recognizing target samples, or define efficient and diverse decision boundaries by starting selection of trajectories from the farthest ones to the target class. Experimental results obtained using both synthetic and video data sets indicate that the proposed ensembles outperform several baseline techniques over a range of test set imbalance levels. Although using all non-target trajectories eliminate the risk of information loss, not all samples are informative and yield better generalization. Therefore, in a next step the trajectories are further under-sampled to support vectors that improve these ensembles' performance and efficiency.

CHAPTER 4

PROGRESSIVE BOOSTING FOR CLASS IMBALANCE AND ITS APPLICATION TO FACE RE-IDENTIFICATION

Most of the ensemble learning methods to handle imbalance in the literature are static approaches. Boosting (Freund & Schapire, 1995; Freund *et al.*, 1996) is a common static ensemble method that has been modified in several ways to learn from imbalanced data (see the reviews by (Galar *et al.*, 2012; Branco *et al.*, 2016; Krawczyk, 2016b; Haixiang *et al.*, 2016)). In data-level Boosting approaches, training data is rebalanced by up-sampling positive class, under-sampling negative class, or using both up-sampling and under-sampling (Chawla *et al.*, 2003; Hu *et al.*, 2009; Mease *et al.*, 2007; Guo & Viktor, 2004; Seiffert *et al.*, 2010; Galar *et al.*, 2013b; Díez-Pastor *et al.*, 2015). Up-sampling methods like SMOTEBoost (Chawla *et al.*, 2003) are often more accurate, but they are computationally complex. In contrast, random under-sampling (RUS) (Seiffert *et al.*, 2010) is more computationally efficient, but suffers from information loss. Partitional approaches (Yan *et al.*, 2003; Li *et al.*, 2013) avoid information loss by splitting the negative class to uncorrelated subsets and training classifiers using all of these subsets.

Another issue with Boosting-based ensembles is that they may suffer from the bias of performance towards negative class because the loss factor, which guides their learning process, is obtained based on weighted accuracy. In cases of imbalance, weighted accuracy reflects the ability for correct classification of negative samples more than positive ones. This issue can be avoided by adopting a cost-sensitive approach (Fan *et al.*, 1999; Ting, 2000; Sun *et al.*, 2007), that defines different misclassification costs for different classes and integrates these cost factors into Boosting learning process. The drawback of these cost-sensitive techniques is that they rely on the suitable selection of cost factors which is often estimated by searching a range of possible values. In contrast, cost-free techniques modify learning algorithms by enhancing loss factor calculation without considering cost factors (Joshi *et al.*, 2001; Kim *et al.*, 2015).

In this chapter we address the two above mentioned issues of Boosting-based ensembles in imbalanced problems, and in particular in face re-identification applications: the computational complexity of up-sampling methods, the loss of information of under-sampling ones; and the bias of the standard loss factor towards the negative class. To this aim we propose the Progressive Boosting (PBoost) algorithm to design static classifier ensembles that can maintain a high level of performance over a range of possible levels of imbalance and complexity in the data encountered during operations.

PBoost uses a partitioning method inspired by face re-identification applications, Trajectory Under-Sampling (TUS), that we proposed in Chapter 3. TUS uses partitions of the negative class based on tracking information to design ensembles of classifiers. In particular, samples from the negative class are regrouped into disjoint partitions, and over iterations, these partitions are gradually accumulated into a temporary design subset.

However, samples from the newly added partition and the important samples from previous iterations have an equally higher probability of being selected than the other negative samples in the temporary set (those that were correctly classified in the previous iteration). The base classifier is then validated on the whole temporary subset. As with traditional Boosting ensembles, the samples that are misclassified are considered as the most important samples and their weights increase. With the sample selection scheme proposed in this Chapter, loss of information is considerably reduced, correlation among subsets of negative class is low, and only important samples tend to appear in more than one training subset. Therefore, the diversity and accuracy of Boosting ensembles tend to increase. In addition, to avoid biasing the performance towards the negative class, we propose employing a loss factor based on the F_β-measure in the proposed PBoosting algorithm.

The diverse pool of classifiers generated with PBoost allows to globally model a range of different levels of imbalance and decision boundary complexities for the data. Therefore, the static ensembles produced using PBoost are robust to possible variations in data processed during operations because base classifiers are validated on a growing number of negative samples

(imbalance level). In addition, the number of samples used per iteration to design (train and validate) a classifier in this ensemble is smaller than Boosting methods in the literature, which translates to a lower computational complexity for design.

The contributions of the proposed PBoost algorithm for face re-identification is summarized as follows:

- A new sample selection process for designing Boosting ensembles where negative class samples enter the Boosting process in uncorrolated partitions to avoid loss of information. Specifically for face re-identification application, the disjoint partitions are selected using tracking information.
- Modifying the validation step in Boosting learning such that base classifiers are validated on growing number of negative samples to increase robustness to imbalance and decrease computation complexity;
- Proposing a specific loss factor (F-measure) in Boosting algorithm to avoid bias of performance towards the majority class.

The PBoost algorithm has been compared to state of the art Boosting ensembles on synthetic and video datasets, that emulate face re-identification application, in terms of both accuracy and computational complexity. The content of this chapter has been partially published in a conference paper presented in ICPR2016 and the full content has been published as an article in Expert Systems with Applications Journal.

The rest of the chapter is structured as follows. In Section 1, the proposed PBoost algorithm is described. The experimental methodology and results are presented in Sections 2 and 3, respectively.

4.1 Progressive Boosting for Learning Ensembles from Imbalanced Data

The Progressive Boosting (PBoost) learning method is proposed to sustain a high level of performance over a range of imbalance and complexity levels in the data seen during operations. This method follows a static approach, and learns ensembles based on a combination of under-sampling and cost-free adjustment of Boosting ensemble learning.

With the PBoost algorithm, negative class is partitioned into disjoint subsets. These partitions are accumulated into a temporary design set progressively as learning iterations proceed. In each iteration, a subset of this temporary set is used for training a classifier such that the most important samples plus samples from the new partition are given an equally high opportunity to be used in training a base classifier. Loss of information is therefore avoided and ensemble diversity is increased. The trained classifier is then validated on the temporary set that contains all positive samples and only those negative partitions that have already been used in previous training iterations. As the temporary set grows, its imbalance level increases and therefore, the ensemble's robustness to diverse levels of skew and decision boundary complexities during operations is increased. In PBoost, the error of the classifier is determined based on its ability to correctly classify both positive and negative classes. This loss factor plays an important role in determining the contribution of classifiers in final prediction, and in selection criteria of samples for designing the next classifiers.

There are several possible ways to partition the negative samples into disjoint subsets in the literature (Xu & Wunsch, 2005) e.g., prototype-based methods like k-means and GMM algorithms, affinity-based methods like spectral, normalized-cut and sub-space algorithms to represent the negatives, and thus define partitions (number of clusters and association of data to clusters). Two general-purpose partitioning techniques have been used in the literature to partition data to learn ensembles from imbalanced data: Random Under-Sampling without replacement (we call RUSwR in this Chapter) (Yan *et al.*, 2003) , and Cluster Under-Sampling (CUS) (Li *et al.*, 2013). In some applications the data is already partitioned, like binarization

of multi-class classification problems using one-vs-all strategy. In some others, the data may be grouped based on some contextual or application-based knowledge of data.

Trajectory Under-Sampling (TUS) is applicable in video surveillance applications where Regions Of Interest (ROIs), which are faces in face re-identification, are regrouped into a set called a trajectory with a high quality face tracker (as proposed in the previous chapter).

A high quality tracking system finds the trajectories by efficiently following the location of the ROIs that belong to the same individual over consecutive video frames. This application-based under-sampling method appeared to be more effective than the general-purposed under-sampling methods in designing classifier ensembles for face re-identification in terms of diversity and accuracy of opinions.

The progressive Boosting method is presented in Algo. 7 and Figure 4.1. Its main steps are explained in the following. The negative samples are regrouped to E disjoint partitions $\mathbf{P}_e$, where $e = 1,..,E$, one per classifier in the ensemble (line 1). E and the number of negative samples in each partition N_e^P varies and depends on the partitioning method and the data distribution. In the case of random under-sampling without replacement E is preselected and N_e^P takes a fixed random value $N_e^P \in [M^+/2, 2M^+]$ such that $\sum_{e=1}^{E} N_e^P = M^-$. In the case of CUS and TUS, E and N_e^P depend on the number of samples that are assigned to each partition by the clustering algorithm and the tracker, respectively.

Given a training data set $\mathbf{S}$, one partition $\mathbf{P}_e$ is selected in each iteration and added to a temporary set $\mathbf{S}_e^{\text{tmp}}$ (line 7) which initially contains the positive samples. The same initial weight w_{ini} is assigned to the samples in the new partition creating a weight vector $\mathbf{W}_e^p$ (line 6) which is also added to a temporary weight set $\mathbf{W}_e^{\text{tmp}}$ (line 7). In the next step (line 9), N_e^P samples from the temporary set $\mathbf{S}_e^{\text{tmp}}$ are selected through random under-sampling to create a new subset $\mathbf{S}_e^{'}$ with the weight distribution of $\mathbf{W}_e^{'}$. A classifier C_e is trained on $\mathbf{S}_e^{'}$ (line 10). Then it is tested on the whole temporary set $\mathbf{S}_e^{\text{tmp}}$ that has an imbalance level of $\Lambda_e = 1 : \sum_{f=1}^{e} {}^{N_f}/_{M^+}$ (line 11). Therefore, the classifiers in this ensemble are in fact validated on data subsets with a growing level of imbalance and complexity.

After that, a new loss factor is calculated that adapts Boosting algorithm for classifying imbalanced data based on the F_β-measure (line 19). To calculate the loss factor, we first split the temporary weight vector $\mathbf{W}_e^{\text{tmp}}$ to two weight matrices for target $\mathbf{W}_e^{\text{tmp},+}$ and non-target $\mathbf{W}_e^{\text{tmp},-}$ classes. The size of $\mathbf{W}_e^{\text{tmp},+}$ is M^+ and the size of $\mathbf{W}_e^{\text{tmp},-}$ is $\sum_{f=1}^{e} N_f$, and:

$$\mathbf{W}_e^{\text{tmp},+} = \{\mathbf{W}_e^{\text{tmp}}(j), j = 1,...,(M^+ + \sum_{f=1}^{e} N_f) | y_j = 1\} \ , \tag{4.1}$$

$$\mathbf{W}_e^{\text{tmp},-} = \{\mathbf{W}_e^{\text{tmp}}(j), j = 1,...,(M^+ + \sum_{f=1}^{e} N_f) | y_j = -1\} \ . \tag{4.2}$$

Then, weighted versions of true positive, false positive, true negative and false negative counts are defined as:

$$\text{TP}_e = \sum_{k:Y_k=1} \mathbf{W}_e^{\text{tmp},+}(k), k = 1,...,M^+, \tag{4.3}$$

$$\text{FP}_e = \sum_{k:Y_k=1} \mathbf{W}_e^{\text{tmp},-}(k), k = 1,..., \sum_{f=1}^{e} N_f, \tag{4.4}$$

$$\text{TN}_e = \sum_{k:Y_k=-1} \mathbf{W}_e^{\text{tmp},-}(k), k = 1,..., \sum_{f=1}^{e} N_f, \tag{4.5}$$

$$\text{FN}_e = \sum_{k:Y_k=-1} \mathbf{W}_e^{\text{tmp},+}(k), k = 1,...,M^+. \tag{4.6}$$

To measure the error of the classifiers, the corresponding loss factor is defined as:

$$L_e = 1 - A_F = \frac{\text{FP}_e + \beta^2 \text{FN}_e}{(1+\beta^2)\text{TP}_e + \text{FP}_e + \beta^2 \text{FN}_e} \ . \tag{4.7}$$

After calculation of α_e (line 22) from:

$$\alpha_e = \frac{L_e}{1 - L_e}, \tag{4.8}$$

the weights in the temporary set $\mathbf{W}_e^{\text{tmp}}$ are updated (line 23) as:

$$\mathbf{W}_{\text{e+1}}^{\text{tmp}}(j) = \mathbf{W}_{\text{e}}^{\text{tmp}}(j)\ \alpha_e^{|\text{y}_\text{j} - \text{Y}_\text{j}|/2}\ . \tag{4.9}$$

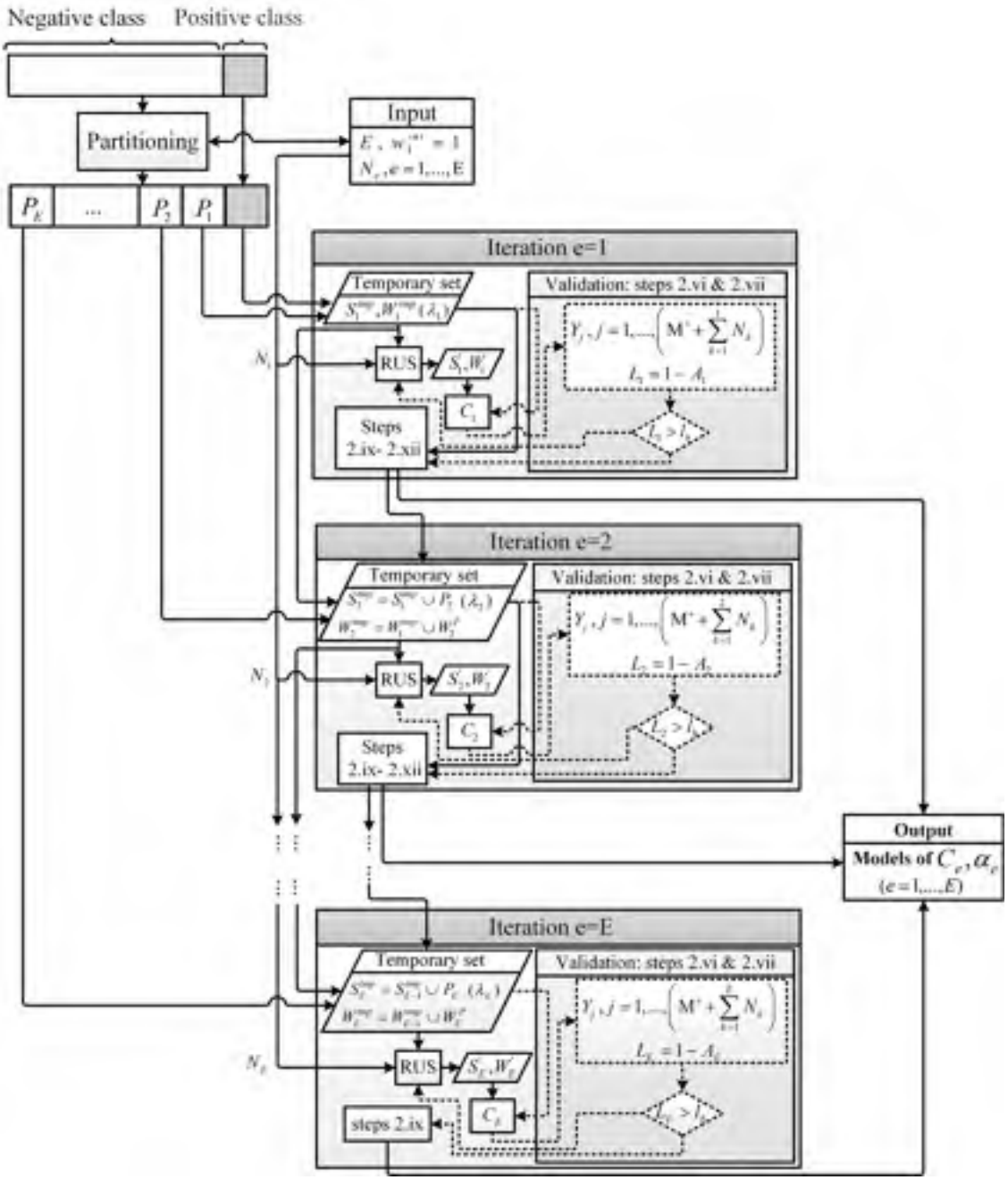


Figure 4.1 Block diagram representation of PBoost learning method.

Algorithm 7: Progressive Boosting ensemble learning method.

Input: Training set: $\mathbf{S} = \{(\mathbf{x}_i, y_i); i = 1, ..., M\}, y_i \in \{-1, 1\}, M = M^- + M^+$

Output: Predicted score or label: $H(\cdot)$

Partition non-target samples from $\mathbf{S}$ into E clusters $\{\mathbf{P}_e; e = 1, ..., E\}$.

Create a temporary training set and weight vector: $\mathbf{S}_1^{\text{tmp}} \leftarrow \{(\mathbf{x}_i, y_i) \in \mathbf{S} | y_i = 1\}$ and $\mathbf{W}_1^{\text{tmp}}(k) = 1, k = 1, ..., M^+$.

Initialize $w_1^{\text{ini}} = 1$.

Set $l_b = \frac{M^-}{(1+\beta^2)M^+ + M^-}$

for $e = 1, .., E$ **do**

  Initialize weight distribution of $\mathbf{P}_e$ as $\mathbf{W}_e^p(k) = w_e^{\text{ini}}, k = 1, ..., N_e^P$.

  $\mathbf{S}_e^{\text{tmp}} \leftarrow \mathbf{S}_e^{\text{tmp}} \bigcup \mathbf{P}_e$, $\mathbf{W}_e^{\text{tmp}} \leftarrow \mathbf{W}_e^{\text{tmp}} \bigcup \mathbf{W}_e^p$

  Normalize $\mathbf{W}_e^{\text{tmp}}$ such that: $\sum \mathbf{W}_e^{\text{tmp}} = 1$.

  Randomly select N_e^P non-target samples from $\mathbf{S}_e^{\text{tmp}}$ based on $\mathbf{W}_e^{\text{tmp}}$, to create a training subset $\mathbf{S}_e^{'}$ with $\mathbf{W}_e^{'}$.

  Train C_e on $\mathbf{S}_e^{'}$ with $\mathbf{W}_e^{'}$.

  Test C_e on $\mathbf{S}_e^{\text{tmp}}$ and get back labels $Y_j, j = 1, ..., (M^+ + \sum_{f=1}^{e} N_f)$.

  Calculate the pseudo-loss for $\mathbf{S}_e^{\text{tmp}}$ from $\mathbf{W}_e^{\text{tmp}}$ (using Equations 4.1 to 4.6):

  $\mathbf{W}_e^{\text{tmp},+} = \{\mathbf{W}_e^{\text{tmp}}(j), j = 1, ..., (M^+ + \sum_{f=1}^{e} N_f) | y_j = 1\}$,

  $\mathbf{W}_e^{\text{tmp},-} = \{\mathbf{W}_e^{\text{tmp}}(j), j = 1, ..., (M^+ + \sum_{f=1}^{e} N_f) | y_j = -1\}$,

  $\text{TP}_e = \sum_{(k,Y_k):Y_k=1} \mathbf{W}_e^{\text{tmp},+}(k), k = 1, ..., M^+$,

  $\text{FP}_e = \sum_{(k,Y_k):Y_k=1} \mathbf{W}_e^{\text{tmp},-}(k), k = 1, ..., \sum_{f=1}^{e} N_f$,

  $\text{TN}_e = \sum_{(k,Y_k):Y_k=-1} \mathbf{W}_e^{\text{tmp},-}(k), k = 1, ..., \sum_{f=1}^{e} N_f$,

  $\text{FN}_e = \sum_{(k,Y_k):Y_k=-1} \mathbf{W}_e^{\text{tmp},+}(k), k = 1, ..., M^+$,

  $L_e = 1 - A_F = \frac{\text{FP}_e + \beta^2 \text{FN}_e}{(1+\beta^2)\text{TP}_e + \text{FP}_e + \beta^2 \text{FN}_e}$.

  If $L_e > l_b$ **go to** step 9

  Calculate the weight update parameter:

  $\alpha_e = \frac{L_e}{1 - L_e}$

  Update $\mathbf{W}_{\text{e+1}}^{\text{tmp}}(j) = \mathbf{W}_{\text{e}}^{\text{tmp}}(j) \alpha_e^{|\text{y}_\text{j} - \text{Y}_\text{j}|/2}$

  Normalize $\mathbf{W}_{\text{e+1}}^{\text{tmp}}$ such that: $\sum \mathbf{W}_{\text{e+1}}^{\text{tmp}} = 1$.

  Set $w_{\text{e+1}}^{\text{ini}} = \max(\mathbf{W}_{\text{e}}^{\text{tmp}}), y_j = -1$

for $e = 1, .., E$ **do**

  l Test C_e classifier on $\mathbf{x}$ and get back $h_e(\mathbf{x})$.

Output the final hypothesis: $H(\cdot) = \sum_{e=1}^{E} h_e(\cdot) \log \frac{1}{\alpha_e}$

Even though it is desirable to limit the loss of information during under-sampling of data, some samples (like borderline samples) are of more interest than others for training classifiers

in the ensemble. In Boosting ensembles, these samples are often detected as misclassified samples because borderline samples play more important role in defining the decision boundary and they are more likely to be misclassified. More importance is given to these samples by assigning higher weights to them, so that they have a higher chance to be included in training subset(s). In the proposed PBoost ensemble, after normalization of $\mathbf{W}_e^{\text{tmp}}$, its maximum value among negative samples is selected as the initial weight for the next iteration (line 25):

$$w_{\text{e}+1}^{\text{ini}} = \max_{y_j=-1} \{\mathbf{W}_{\text{e}}^{\text{tmp}}(j)\},\ j = 1, ..., M^+ + \sum_{f=1}^{e} N_f. \tag{4.10}$$

This value corresponds to the weight of more important misclassified negative samples. Therefore, in each iteration, new samples and misclassified samples from previous iterations have more chance to be included in the training subset. Finally, α_e is used to obtain the final class prediction of the ensemble from in line 28.

PBoost is somewhat inspired from RUSBoost, but differs in three main respects. First, during each iteration, instead of random under-sampling with replacement, most of training negative samples are selected from disjoint partitions. Consequently, repeatedly selection of the same samples over all iterations and information loss is avoided while the diversity increases. Second, instead of validating the classifiers on all samples, the classifiers are validated only on a subset of training set that grows in size and imbalance over iterations. Therefore, robustness to different levels of data imbalance and complexity increases, and the computations complexity of validation step decreases significantly. Third, instead of weighted accuracy, F-measure, an imbalance-compatible performance metric, avoids biasing performance towards negative class.

4.2 Experimental Methodology

In the experiments of this chapter, the proposed PBoost ensemble learning method is compared to AdaBoost.M1 (Freund *et al.*, 1996), and one method from each family of the data-level Boosting approaches reviewed in Section 1.2.1.1 including SMOTEBoost (Chawla *et al.*, 2003), RUSBoost (Seiffert *et al.*, 2010), and RB-Boost (Díez-Pastor *et al.*, 2015). The datasets

that are used for the experiments include: (1) A set of synthetic 2D data sets in which the level of skew and overlap between classes are controllable, (2) the Face In Action (FIA) video database (Goh *et al.*, 2005) that emulates a passport checking scenario in face re-identification application, and (3) COX Face dataset the face recognition in video surveillance applications (Huang *et al.*, 2015)(see Section 2.1.2)

A brief description of the implemented ensembles, their variants and the abbreviations used for them are shown in Table 4.1. The abbreviations assigned to these ensembles are selected based on their sampling techniques and loss factor as described in the following.

The baseline sampling techniques include Ada (resampling in AdaBoost), SMT (SMOTE in SMOTEBoost), RUS (random under-sampling in RUSBoost), RB (random balance in RB-Boost). For PBoost three partitioning techniques are used for under-sampling the negative class to evaluate the effect of the partitioning technique on the performance of PBoost ensemble: random under sampling without replacement (PRUS) and cluster under-sampling (PCUS) are used as general partitioning techniques for PBoost disregarding the data structure, whether or not the negative class is partitioned a priori. For PCUS, kernel k-means is used for clustering negative samples. To select k, it is varied over a range of possible values and the value of Dunn index (Dunn, 1973) is calculated for each case using a validation set. Finally, the optimal k, is selected when Dunn index takes its maximum value. The third partitioning technique is progressive trajectory under-sampling (PTUS).

The loss factor is calculated in two ways based on: the traditional technique i.e. weighted accuracy, and the F-measure. To indicate the use of F-measure in the Boosting ensembles in Table 4.1, the abbreviation is followed by -F. For the use of proposed loss factor calculation with the F-measure, β is set as 2 in all experiments because $\beta \geq 1$ is more suitable for imbalanced data classification when the positive class is the minority class. An experiment is done to evaluate the performance of Boosting ensembles with different values of β.

The performance of classifiers is measured in terms of AUPR (as a global performance metric), the F_2-measure (a metric of performance that takes the imbalance level of data into account and

Table 4.1 Ensembles and their variants.

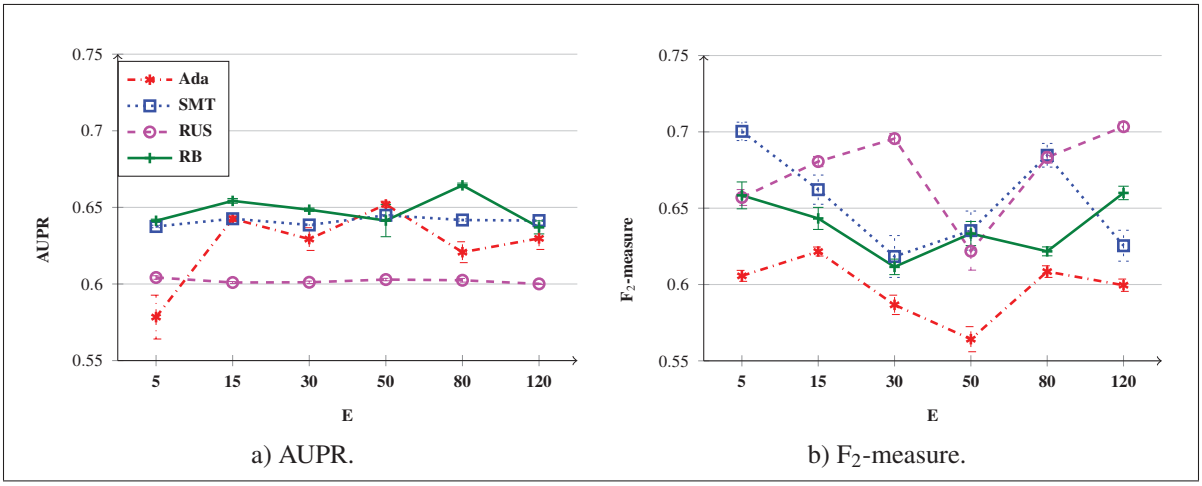
Abbreviation	Sampling method	Boosting Ensemble	Loss factor
Ada:	Resampling with replacement	AdaBoost (Freund *et al.*, 1996)	Weighted accuracy
Ada-F:	Resampling with replacement	Modified AdaBoost (Freund *et al.*, 1996)	Proposed F-measure
SMT:	Synthetic minority over-sampling technique (SMOTE)	SMOTEBoost (Chawla *et al.*, 2003)	Weighted accuracy
SMT-F	Synthetic minority over-sampling technique(SMOTE)	Modified SMOTEBoost (Chawla *et al.*, 2003)	Proposed F-measure
RUS:	Random under-sampling with replacement (RUS)	RUSBoost (Seiffert *et al.*, 2010)	Weighted accuracy
RUS-F:	Random under-sampling with replacement (RUS)	Modified RUSBoost (Seiffert *et al.*, 2010)	Proposed F-measure
TUS:	Selecting trajectories in video dataset or ideal clusters in synthetic dataset	Growing imbalance	-
RB:	Combination of up-sampling (SMOTE) and under-sampling (RUS)	RB-Boost (Díez-Pastor *et al.*, 2015)	Weighted accuracy
RB-F:	Combination of up-sampling (SMOTE) and under-sampling (RUS)	Modified RB-Boost (Díez-Pastor *et al.*, 2015)	Proposed F-measure
PRUS:	Random under-sampling without replacement (RUSwR)	Progressive Boosting	Weighted accuracy
PRUS-F:	Random under-sampling without replacement (RUSwR)	Progressive Boosting	Proposed F-measure
PCUS:	Selecting clusters found by k-means	Progressive Boosting	Weighted accuracy
PCUS-F:	Selecting clusters found by k-means	Progressive Boosting	Proposed F-measure
PTUS:	Selecting trajectories or ideal clusters	Progressive Boosting	Weighted accuracy
PTUS-F:	Selecting trajectories or ideal clusters	Progressive Boosting	Proposed F-measure

gives a little more importance to the target class by setting $\beta = 2$), and the G-mean (to measure the performance for both classes at the same time disregarding the imbalance).

In the experiments with synthetic and video data sets, two different imbalance levels are used for training and four different imbalance levels are used for testing. This is to evaluate the sensitivity of classification systems to the level of imbalance during training and their robustness to possible variations in skew level during operations. In experiments with synthetic data, the overlap level between positive and negative classes are also varied because the issue of imbalance is related to the level of overlap between classes (López *et al.*, 2013).

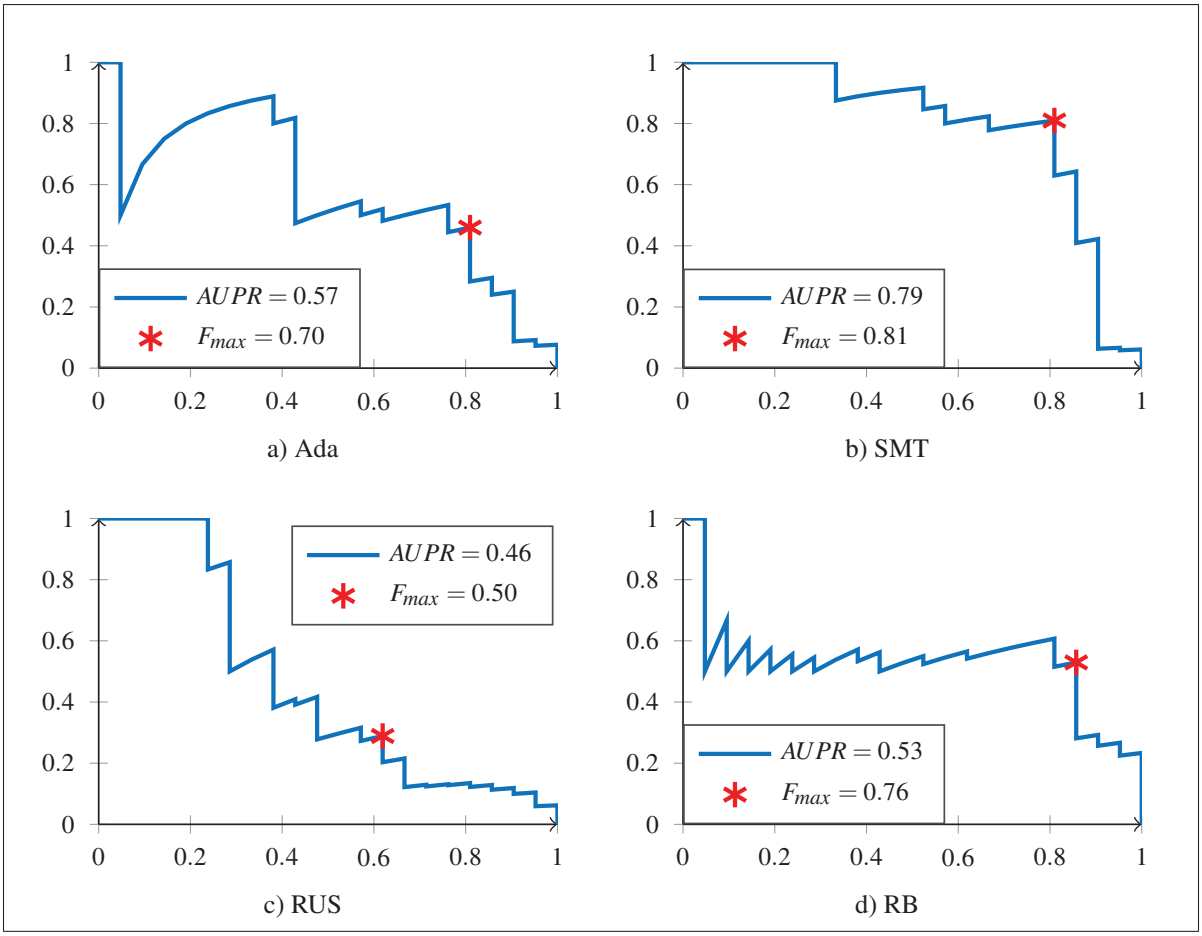
In experiments with synthetic and video datasets, the size of all Boosting ensembles is set equal to the maximum imbalance level of the data, except from PCUS. The reason for this setting is that the number of ideal clusters and the number of trajectories are both known and equal to the level of skew. In addition, based on a preliminary experiment under setting D_2 (see Table 2.1) on baseline ensembles in Figure 4.2, it is observed that the size of these ensembles does not have a significant impact on their performance. The performance of these ensembles vary in terms of F_2-measure as the ensemble size grows. However, their global performance in terms of AUPR do not change significantly. For PCUS, the size of ensemble is selected equal to the optimal k obtained using Dunn index.

The values of performance metrics are averaged over all replications. In the experiments of this chapter, the decision threshold to obtain the F_2-measure is set to the value that maximizes



a) AUPR. b) F_2-measure.

Figure 4.2 Performance of baseline Boosting ensembles for different values of E on D_2 with $\Lambda_{\text{test}} = 1:100$.



a) Ada b) SMT

c) RUS d) RB

Figure 4.3 PR curve of baseline Boosting ensembles on validation data and finding the optimal threshold

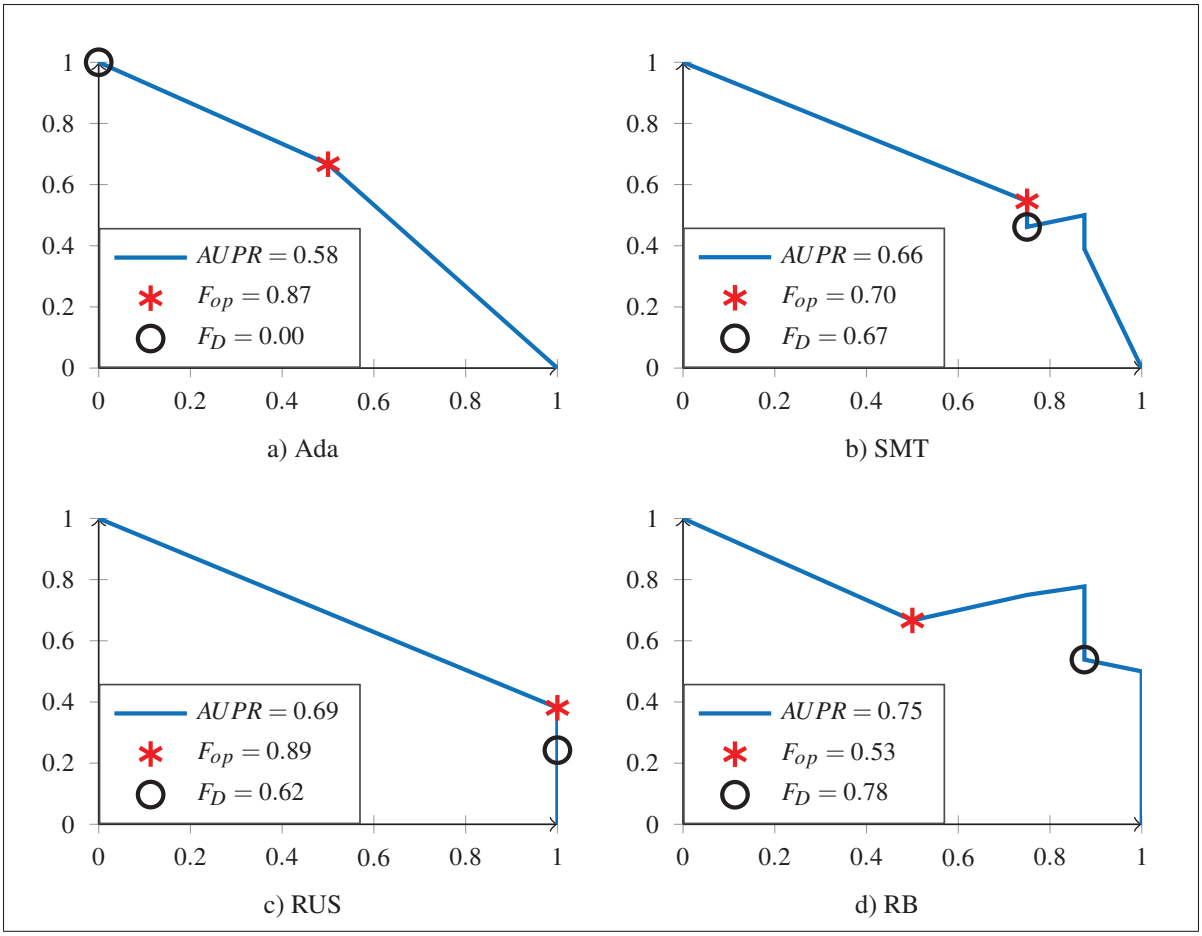


Figure 4.4 PR curve of baseline Boosting ensembles on test data using the optimal threshold obtained from validation step.

the value of F_2-measure on the validation data for comparing the performance of different classification algorithms.

An example is shown in Figure 4.3, the PR curve of an experiment under setting D_2 on the validation data with skew level of 1:50. In Figure 4.4, the ensembles are tested on a different test set and F_{op} shows the value of F-measure when the optimal threshold is selected using the validation step described. F_{D} is the value of F-measure when the combination function in Boosting ensembles is majority voting and the decisions of base classifiers are combined. It is observed that F_{op} and F_{D} may differ significantly and in most cases $F_{\text{op}} > F_{\text{D}}$.

In our experiments, the performance of the proposed PBoost ensemble is also compared to Boosting ensembles from literature in terms of computational complexity. Time complexity for SVM training depends on several factors including the number of training samples, the learning (optimization) algorithm and the number of features. The computational complexity of SVM implemented in LibSVM is evaluated in (Chang & Lin, 2011), as $O(n_{\text{tr}}d)$ per iteration I, where n_{tr} is the training set size, and d is the number of features. The authors state that "the number of iterations p may be higher than linear to the number of training data". Therefore, the complexity is $O(n_{\text{tr}}^{p} \cdot d)$ for some $p > 2$. This means that, time complexity for SVM training is not proportional to, but increases more than linearly with respect to the training set size.

In the proposed PRUS and baseline Boosting ensembles, p is unknown and d is identical in all algorithms. Each iteration of Boosting ensembles includes a validation step that should be added to training complexity to obtain the overall time complexity of learning process. Time complexity of the validation step $O(n_{\text{SV}} \cdot n_{\text{val}})$, depends on the number of validation samples n_{val} and the number of support vectors n_{SV} obtained from training each SVM. The reason is that, when an RBF SVM with n_{SV} support vectors is tested on a probe sample $\mathbf{x}$, the value of $K(\mathbf{x}, \text{SV}_\text{j}) = \exp\{{}^{-\|\mathbf{x}-\text{SV}_\text{j}\|^2}/_{2\kappa^2}\}$ is accumulated for all support vectors ($j = 1, \ldots, n_{\text{SV}}$) and the sign of the resulting quantity determines the decision.

Table 4.2 shows the number of samples to train and validate the ensembles of the size E. The number of validation samples in baseline Boosting ensembles is the same and equal to the overall number of training samples. However, the overall number of samples used for validation in PTUS is calculated as:

$$\sum_{e=1}^{E}(M^{+} + \sum_{f=1}^{e} N_f) = EM^{+} + \sum_{e=1}^{E}(E-(e-1))N_e^{P}, \tag{4.11}$$

$$= EM^{+} + E^2 - \sum_{e=1}^{E} eN_e^{P} + M^{-}, \tag{4.12}$$

$$= EM^{+} + M^{-} + E^2 - \sum_{e=1}^{E} eN_e^{P}. \tag{4.13}$$

Table 4.2 Number of training and validation samples.

Ensemble	n_{tr} in iteration e	Total n_{tr}	n_{val} in iteration e	Total n_{val}
Ada	$M^+ + M^-$	$E(M^+ + M^-)$	$M^+ + M^-$	$E(M^+ + M^-)$
SMT	$2M^-$	$2EM^-$	$M^+ + M^-$	$E(M^+ + M^-)$
RUS	$2M^+$	$2EM^+$	$M^+ + M^-$	$E(M^+ + M^-)$
TUS	$M^+ + \sum_{f=1}^{e} N_f$	$EM^+ + M^- + E^2 - \sum_{e=1}^{E} eN_e^P$	$M^+ + \sum_{f=1}^{e} N_f$	$EM^+ + M^- + E^2 - \sum_{e=1}^{E} eN_e^P$
RB	$M^+ + M^-$	$E(M^+ + M^-)$	$M^+ + M^-$	$E(M^+ + M^-)$
PTUS	$M^+ + N_e^P$	$EM^+ + M^-$	$M^+ + \sum_{f=1}^{e} N_f$	$EM^+ + M^- + E^2 - \sum_{e=1}^{E} eN_e^P$

This value is less than $E(M^+ + M^-)$ that is the total number of validation samples in the sate of the art Boosting ensembles. Table 4.2 shows that the total number of training and validation samples in PTUS ensemble is the smallest one. The number of training and validation samples in TUS is also found using Eq. 4.13.

4.3 Results and Discussion

The performance of the proposed and ensemble learning methods from the literature are analyzed for synthetic and video data in 4 parts: (1) accuracy and robustness over different levels of overlap and imbalance between design and test data and of using the proposed loss factor; (2) the performance of RUSBoost with and without progressive partitioning; (3) the combined impact of progressive partitioning and proposed loss factor; (4) the computation complexity during design and testing.

4.3.1 Results of Experiments with Synthetic Data

4.3.1.1 Impact of Loss Factor Based on the F-measure

The performance of the baseline Boosting ensembles: AdaBoost , SMOTEBoost, RUSBoost, and RB-Boost are compared in Tables 4.3, 4.4 and 4.5 for different settings. In addition, F_β is used to optimize loss factor calculation in these ensembles.

Given a fixed skew level of test data, the performance of all Boosting ensembles declines in terms of F-measure and AUPR as the overlap between positive and negative classes grows.

This decline of performance is more significant when test data is imbalanced compared to the case where test data is balanced. In our experiments, changes in skew level of test data result in different number of misclassified negative samples and no change in the number of correctly classified positive samples. Therefore, even for the same level of overlap, the performance of all ensembles degrades, in terms of F-measure and AUPR when testing on a more imbalanced data. However, the value of G-mean is relatively high and does not change significantly with variations in imbalance.

For the same level of overlap and different imbalance of training data (settings D_1 and D_3) the performance of all Boosting ensembles is lower when imbalance of training data is lower. The reason is that less information is provided for training and also the skew level of training and test data has a greater difference. Overall, Tables 4.3, 4.4 and 4.5 show that SMT and RB are the most robust to changes in overlap and imbalance than Ada and RUS. In addition, using the F-measure loss factor improves the performance of RUS significantly with D_1, D_2 and D_3 for all Λ_{test} in terms of all three metrics. Performance of SMT and RB improves only in terms of AUPR and with D_1 and D_2. Performance of Ada improves when $\Lambda_{\text{test}} = 1:100$ in terms of G-mean and AUPR.

Using the F-measure loss factor may improve the performance of the Boosting ensembles that rely on under-sampling of data in terms of F-measure, especially for more difficult problems with overlapping data. The performance of Boosting ensembles that involve up-sampling of positive samples does not improve significantly in terms of F-measure. However, the global performance of these Boosting ensembles in terms of AUPR does improve after using the F-measure loss factor.

In Table 4.6, the performance of baseline ensembles and their variants for different values of β is compared for $D2$. The goal is to evaluate the effect of the value of β on improving the performance when the proposed loss factor is used. This Table shows the performance only when $\Lambda_{\text{test}} = 1:100$ because the performance of baseline systems usually decline for higher skew levels of test data.

Table 4.3 Average of F_2-measure performance of baseline techniques with and without F-measure loss factor on synthetic data over different levels of skew and overlap in test data.

Ensembles	Train Data	D_1 (Λ_{train} =1:50, $\delta = 0.2$)				D_2 (Λ_{train} =1:50, $\delta = 0.1$)				D_3 (Λ_{train} =1:20, $\delta = 0.2$)			
	Λ_{test}	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**
Ada		0.97 ± 0.02	0.97 ± 0.03	0.92 ± 0.02	0.78 ± 0.04	0.85 ± 0.05	0.75 ± 0.07	0.62 ± 0.05	0.42 ± 0.05	0.98 ± 0.02	0.93 ± 0.03	0.85 ± 0.02	0.57 ± 0.04
Ada-F		0.96 ± 0.03	0.96 ± 0.03	0.92 ± 0.03	**0.79** ± 0.04	**0.87** ± 0.09	0.74 ± 0.08	0.58 ± 0.05	0.42 ± 0.08	0.98 ± 0.02	0.92 ± 0.03	0.85 ± 0.02	**0.58** ± 0.05
RUS		0.89 ± 0.11	0.85 ± 0.21	0.55 ± 0.19	0.34 ± 0.15	0.82 ± 0.09	0.62 ± 0.10	0.47 ± 0.08	0.27 ± 0.08	0.56 ± 0.20	0.35 ± 0.24	0.21 ± 0.18	0.13 ± 0.11
RUS-F		**0.93** ± 0.06	**0.93** ± 0.06	**0.81** ± 0.11	**0.63** ± 0.17	**0.93** ± 0.04	**0.71** ± 0.06	**0.51** ± 0.03	**0.36** ± 0.07	**0.85** ± 0.15	**0.77** ± 0.20	**0.67** ± 0.21	**0.47** ± 0.18
SMT		0.96 ± 0.04	0.96 ± 0.04	0.94 ± 0.04	0.90 ± 0.03	0.94 ± 0.04	0.85 ± 0.02	0.65 ± 0.02	0.61 ± 0.02	0.96 ± 0.04	0.91 ± 0.01	0.90 ± 0.02	0.74 ± 0.03
SMT-F		0.95 ± 0.04	0.95 ± 0.04	0.94 ± 0.04	0.90 ± 0.03	0.94 ± 0.04	0.85 ± 0.02	0.65 ± 0.02	0.61 ± 0.02	0.96 ± 0.04	0.91 ± 0.01	0.90 ± 0.02	0.74 ± 0.03
RB		0.96 ± 0.02	0.96 ± 0.02	0.94 ± 0.02	0.90 ± 0.01	0.91 ± 0.05	0.85 ± 0.01	0.63 ± 0.01	0.60 ± 0.01	0.96 ± 0.03	0.92 ± 0.02	0.90 ± 0.02	0.73 ± 0.05
RB-F		0.96 ± 0.03	0.96 ± 0.03	0.94 ± 0.03	0.90 ± 0.02	0.91 ± 0.03	0.85 ± 0.02	0.63 ± 0.02	0.59 ± 0.03	0.97 ± 0.03	0.91 ± 0.01	0.89 ± 0.03	0.72 ± 0.05

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

Evaluation is done in terms of the same F_β-measure that is used in loss factor calculation. The results are shown in terms of both F_D and F_{op}. F_D is the value of F-measure when the decisions of base classifiers are combined in Boosting ensembles and F_{op} is the value of F-measure when the scores of base classifiers are combined in Boosting ensembles and the optimal decision threshold of each ensemble is set to the point that maximizes F-measure when that ensemble is validated on an independent set of data (see section 4.2.1.).

The performance of Ada improves for all values of β in terms of both F_D and F_{op}. Some improvements are seen for SMT and RB, but F_D and F_{op} tend to stay the same in most cases and decrease in a few cases. The performance of RUS improves for $\beta = 1$ and 2 in terms of both F_D and F_{op}, and the improvement tends to decrease for higher β values. This was expected, since using higher values of β to calculate F-measure means giving more importance to recall than precision. Therefore, the impact of imbalance is masked when higher values of β is used and the performance may not change when the loss factor is calculated based on F-measure. For each of the classification systems, the same reason result in higher values of F_D and F_{op}

Table 4.4 Average of G-mean performance of baseline techniques with and without F-measure loss factor on synthetic data over different levels of skew and overlap in test data.

Ensembles	Train Data	D_1 (Λ_{train} =1:50, $\delta = 0.2$)				D_2 (Λ_{train} =1:50, $\delta = 0.1$)				D_3 (Λ_{train} =1:20, $\delta = 0.2$)			
	Λ_{test}	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**
Ada		0.97 ± 0.02	0.97 ± 0.02	0.97 ± 0.02	0.97 ± 0.02	0.86 ± 0.05	0.86 ± 0.05	0.86 ± 0.05	0.85 ± 0.05	0.98 ± 0.02	0.98 ± 0.02	0.98 ± 0.02	0.97 ± 0.02
Ada-F		0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	**0.88** ± 0.08	**0.87** ± 0.07	**0.87** ± 0.07	**0.87** ± 0.07	0.98 ± 0.02	0.98 ± 0.02	0.98 ± 0.02	0.97 ± 0.02
RUS		0.93 ± 0.05	0.93 ± 0.05	0.92 ± 0.05	0.91 ± 0.06	0.84 ± 0.08	0.83 ± 0.08	0.83 ± 0.08	0.82 ± 0.08	0.50 ± 0.20	0.68 ± 0.23	0.68 ± 0.23	0.68 ± 0.23
RUS-F		**0.94** ± 0.06	**0.94** ± 0.06	**0.93** ± 0.06	**0.93** ± 0.05	**0.93** ± 0.04	**0.92** ± 0.04	**0.92** ± 0.04	**0.92** ± 0.04	**0.80** ± 0.25	**0.91** ± 0.08	**0.91** ± 0.08	**0.91** ± 0.08
SMT		0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.94 ± 0.04	0.94 ± 0.04	0.94 ± 0.04	0.94 ± 0.04	0.96 ± 0.04	0.96 ± 0.03	0.96 ± 0.04	0.96 ± 0.04
SMT-F		0.96 ± 0.04	0.96 ± 0.04	0.96 ± 0.04	0.95 ± 0.04	0.94 ± 0.03	0.94 ± 0.03	0.93 ± 0.03	0.94 ± 0.03	0.96 ± 0.04	0.96 ± 0.03	0.96 ± 0.04	0.96 ± 0.04
RB		0.96 ± 0.02	0.96 ± 0.02	0.96 ± 0.02	0.96 ± 0.02	0.91 ± 0.05	0.91 ± 0.04	0.90 ± 0.04	0.91 ± 0.05	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03
RB-F		0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.91 ± 0.03	0.91 ± 0.03	0.90 ± 0.03	0.91 ± 0.03	0.97 ± 0.03	0.96 ± 0.03	0.97 ± 0.03	0.96 ± 0.03

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

with higher values of β. Comparing F_D and F_{op} of each ensemble for each value of β shows that selecting the proper decision threshold can improve the performance in terms of accuracy and robustness, especially for lower values of β.

The results are not shown in terms of AUPR because AUPR does not change with variations in the value of β, since the importance of recall and precision stays the same and equal in obtaining AUPR.

4.3.1.2 Impact of progressive partitioning in RUSBoost

In this section, progressive partitioning is integrated into RUS without the use of F-measure in loss factor calculation. It is observed in Table 4.8 that robustness of RUS improves significantly after using this method of sampling. Indeed, using all samples for training through partitioning avoids loss of information and may improve the classification accuracy. In addition, validating on different imbalance levels of data increases the robustness to variations in the imbalance

Table 4.5 Average of AUPR performance of baseline techniques with and without F-measure loss factor on synthetic data over different levels of skew and overlap in test data.

Ensembles	Train Data	D_1 (Λ_{train} =1:50, $\delta = 0.2$)				D_2 (Λ_{train} =1:50, $\delta = 0.1$)				D_3 (Λ_{train} =1:20, $\delta = 0.2$)			
	Λ_{test}	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**
Ada		1.00 ± 0.00	1.00 ± 0.00	0.98 ± 0.01	0.89 ± 0.04	1.00 ± 0.00	0.82 ± 0.07	0.66 ± 0.05	0.34 ± 0.07	1.00 ± 0.00	0.99 ± 0.01	0.96 ± 0.01	0.70 ± 0.04
Ada-F		1.00 ± 0.00	1.00 ± 0.00	0.98 ± 0.01	**0.90** ± 0.03	1.00 ± 0.00	0.81 ± 0.08	0.55 ± 0.13	**0.36** ± 0.14	1.00 ± 0.00	0.99 ± 0.01	**0.95** ± 0.02	**0.71** ± 0.05
RUS		0.93 ± 0.21	0.90 ± 0.28	0.46 ± 0.25	0.29 ± 0.23	1.00 ± 0.00	0.70 ± 0.13	0.40 ± 0.08	0.20 ± 0.08	0.64 ± 0.20	0.34 ± 0.33	0.20 ± 0.23	0.10 ± 0.11
RUS-F		**1.00** ± 0.00	**1.00** ± 0.00	**0.86** ± 0.19	**0.71** ± 0.23	**1.00** ± 0.00	**0.75** ± 0.11	**0.37** ± 0.07	**0.26** ± 0.09	**0.89** ± 0.22	**0.81** ± 0.27	**0.73** ± 0.30	**0.51** ± 0.27
SMT		0.90 ± 0.32	0.90 ± 0.32	0.89 ± 0.31	0.88 ± 0.31	0.90 ± 0.32	0.84 ± 0.30	0.52 ± 0.19	0.51 ± 0.19	1.00 ± 0.00	0.99 ± 0.01	0.98 ± 0.01	0.88 ± 0.01
SMT-F		**1.00** ± 0.00	**1.00** ± 0.00	**0.99** ± 0.01	**0.98** ± 0.01	**1.00** ± 0.00	**0.93** ± 0.03	**0.58** ± 0.04	**0.57** ± 0.05	1.00 ± 0.00	0.99 ± 0.01	0.98 ± 0.01	0.88 ± 0.01
RB		0.80 ± 0.42	0.80 ± 0.42	0.79 ± 0.42	0.78 ± 0.41	0.60 ± 0.52	0.57 ± 0.49	0.32 ± 0.28	0.31 ± 0.27	1.00 ± 0.00	0.99 ± 0.01	0.98 ± 0.01	0.84 ± 0.03
RB-F		**1.00** ± 0.00	**1.00** ± 0.00	**0.99** ± 0.00	**0.98** ± 0.01	**1.00** ± 0.00	**0.94** ± 0.02	**0.53** ± 0.03	**0.51** ± 0.03	1.00 ± 0.00	0.98 ± 0.01	0.98 ± 0.01	**0.85** ± 0.02

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

Table 4.6 Average of F_2-measure performance of baseline techniques with and without F-measure loss factor for different values of β on D_2, $\Lambda_{test} = 1:100$.

Ensembles	F_D					F_{op}				
	$\beta = 1$	$\beta = 2$	$\beta = 4$	$\beta = 7$	$\beta = 10$	$\beta = 1$	$\beta = 2$	$\beta = 4$	$\beta = 7$	$\beta = 10$
Ada	0.24 ± 0.14	0.26 ± 0.15	0.19 ± 0.17	0.20 ± 0.15	0.24 ± 0.16	0.38 ± 0.10	0.25 ± 0.18	0.27 ± 0.16	0.34 ± 0.06	0.32 ± 0.13
Ada-F	**0.30** ± 0.07	**0.31** ± 0.07	**0.33** ± 0.05	**0.34** ± 0.03	**0.33** ± 0.04	**0.41** ± 0.09	**0.39** ± 0.11	**0.36** ± 0.05	**0.36** ± 0.08	**0.38** ± 0.07
RUS	0.09 ± 0.01	0.09 ± 0.01	0.09 ± 0.01	0.09 ± 0.01	0.09 ± 0.01	0.51 ± 0.03	0.46 ± 0.03	0.48 ± 0.04	0.43 ± 0.13	0.48 ± 0.03
RUS-F	**0.19** ± 0.04	**0.10** ± 0.01	0.09 ± 0.01	0.09 ± 0.01	0.08 ± 0.02	**0.52** ± 0.08	**0.48** ± 0.03	0.43 ± 0.10	**0.44** ± 0.12	0.46 ± 0.07
SMT	0.36 ± 0.02	0.36 ± 0.02	0.36 ± 0.02	0.33 ± 0.12	0.33 ± 0.12	0.49 ± 0.05	0.47 ± 0.04	0.41 ± 0.15	0.41 ± 0.15	0.45 ± 0.03
SMT-F	0.36 ± 0.02	0.36 ± 0.02	0.36 ± 0.02	**0.37** ± 0.02	**0.36** ± 0.02	0.49 ± 0.05	0.47 ± 0.04	**0.45** ± 0.03	**0.45** ± 0.03	0.45 ± 0.03
RB	0.33 ± 0.01	0.32 ± 0.02	0.29 ± 0.10	0.26 ± 0.14	0.23 ± 0.16	0.47 ± 0.03	0.41 ± 0.15	0.34 ± 0.18	0.38 ± 0.14	0.29 ± 0.20
RB-F	0.30 ± 0.02	0.30 ± 0.02	**0.30** ± 0.02	**0.30** ± 0.02	**0.31** ± 0.02	0.47 ± 0.04	**0.46** ± 0.05	**0.41** ± 0.02	**0.42** ± 0.02	**0.41** ± 0.02

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

level of test data. The performance of PTUS is significantly better than RUS in terms of both

F-measure and G-mean with D_1, D_2 and D_3 and all Λ_{test}. In terms of AUPR, integrating PTUS improves the performance of RUS with D_2, and higher skew levels of test data with D_1 and D_2. Integrating PRUS and PCUS also improves the performance of RUS specially in terms of F-measure and AUPR and with D_1 and D_3.

Table 4.7 Average of F_2-measure, G-mean and AUPR performance of RUSBoost with and without integrating progressive Boosting on synthetic data over different levels of skew and overlap of test data.

Ensembles	Train Data	D_1 (Λ_{train} =1:50, $\delta = 0.2$)				D_2 (Λ_{train} =1:50, $\delta = 0.1$)				D_3 (Λ_{train} =1:20, $\delta = 0.2$)			
	Λ_{test}	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**
F_2-measure													
RUS		0.89 ± 0.11	0.85 ± 0.21	0.55 ± 0.19	0.34 ± 0.15	0.82 ± 0.09	0.62 ± 0.10	0.47 ± 0.08	0.27 ± 0.08	0.56 ± 0.20	0.35 ± 0.24	0.21 ± 0.18	0.13 ± 0.11
PRUS		0.82 ± 0.09	0.64 ± 0.11	**0.61** ± 0.12	**0.55** ± 0.15	**0.90** ± 0.09	0.43 ± 0.18	0.35 ± 0.14	**0.32** ± 0.11	**0.70** ± 0.06	**0.58** ± 0.18	**0.56** ± 0.20	**0.48** ± 0.19
PCUS		0.83 ± 0.12	0.61 ± 0.17	**0.58** ± 0.16	**0.54** ± 0.14	**0.92** ± 0.13	0.60 ± 0.15	**0.48** ± 0.09	**0.43** ± 0.09	**0.88** ± 0.09	**0.84** ± 0.09	**0.82** ± 0.08	**0.71** ± 0.06
PTUS		**0.97** ± 0.03	**0.92** ± 0.03	**0.92** ± 0.03	**0.90** ± 0.03	**0.99** ± 0.01	**0.79** ± 0.09	**0.63** ± 0.09	**0.58** ± 0.12	**0.94** ± 0.05	**0.92** ± 0.05	**0.91** ± 0.04	**0.78** ± 0.04
G-mean													
RUS		0.93 ± 0.05	0.93 ± 0.05	0.92 ± 0.05	0.91 ± 0.06	0.84 ± 0.08	0.83 ± 0.08	0.83 ± 0.08	0.82 ± 0.08	0.50 ± 0.20	0.68 ± 0.23	0.68 ± 0.23	0.68 ± 0.23
PRUS		0.85 ± 0.06	0.70 ± 0.09	0.69 ± 0.09	0.69 ± 0.09	**0.91** ± 0.08	0.57 ± 0.15	0.53 ± 0.17	0.53 ± 0.17	**0.78** ± 0.05	**0.76** ± 0.06	**0.76** ± 0.06	**0.76** ± 0.06
PCUS		0.86 ± 0.11	0.69 ± 0.14	0.69 ± 0.14	0.67 ± 0.15	**0.95** ± 0.08	0.71 ± 0.15	0.70 ± 0.15	0.69 ± 0.16	**0.89** ± 0.08	**0.87** ± 0.07	**0.87** ± 0.08	**0.85** ± 0.07
PTUS		**0.97** ± 0.03	**0.93** ± 0.03	**0.93** ± 0.03	**0.93** ± 0.03	**0.99** ± 0.01	**0.94** ± 0.02	**0.93** ± 0.02	**0.93** ± 0.06	**0.94** ± 0.05	**0.94** ± 0.05	**0.93** ± 0.04	**0.92** ± 0.05
AUPR													
RUS		0.93 ± 0.21	0.90 ± 0.28	0.46 ± 0.25	0.29 ± 0.23	1.00 ± 0.00	0.70 ± 0.13	0.40 ± 0.08	0.20 ± 0.08	0.64 ± 0.20	0.34 ± 0.33	0.20 ± 0.23	0.10 ± 0.11
PRUS		**1.00** ± 0.00	**0.95** ± 0.14	**0.64** ± 0.30	**0.49** ± 0.35	1.00 ± 0.00	**0.77** ± 0.08	**0.43** ± 0.04	**0.26** ± 0.08	**0.88** ± 0.21	**0.81** ± 0.26	**0.62** ± 0.28	**0.37** ± 0.20
PCUS		**1.00** ± 0.00	**0.96** ± 0.05	**0.69** ± 0.24	**0.45** ± 0.29	1.00 ± 0.00	0.57 ± 0.19	0.38 ± 0.15	**0.27** ± 0.15	**0.99** ± 0.02	**0.86** ± 0.20	**0.76** ± 0.17	**0.53** ± 0.16
PTUS		0.90 ± 0.32	0.90 ± 0.32	**0.89** ± 0.31	**0.86** ± 0.31	0.90 ± 0.32	0.44 ± 0.24	0.35 ± 0.22	**0.29** ± 0.21	**1.00** ± 0.00	**0.97** ± 0.02	**0.96** ± 0.03	**0.80** ± 0.07

The boldface entries correspond to improvement in the performance of RUS after integrating progressive Boosting.

4.3.1.3 Impact of progressive partitioning and loss factor combined

In this section progressive partitioning and the proposed loss factor are integrated into RUS algorithm, resulting in PRUS-F, PCUS-F, and PTUS-F. In terms of F-measure, PTUS-F out-

performs other classification systems for higher skew levels of test data. PTUS-F outperforms others in terms of G-mean with D_2. Comparing Tables 4.3 and 4.5 with 4.7 and 4.8 shows that combining the use of F-measure and progressive partitioning is more effective in increasing performance and robustness compared to using each of them independently because during learning process, accuracy and robustness to imbalance improve at the same time, not separately. If the negative class is partitioned a priori (TUS), PBoost performs significantly better than the case when general partitioning techniques (RUS and CUS) are used.

Table 4.8 Average of F_2-measure performance of proposed and baseline techniques on synthetic data over different levels of skew and overlap of test data.

Ensembles	**Train Data**	D_1 (Λ_{train} =1:50, $\delta = 0.2$)				D_2 (Λ_{train} =1:50, $\delta = 0.1$)				D_3 (Λ_{train} =1:20, $\delta = 0.2$)			
	Λ_{test}	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**
Ada		0.97 ± 0.02	**0.97** ± 0.03	0.92 ± 0.02	0.78 ± 0.04	0.85 ± 0.05	0.75 ± 0.07	0.62 ± 0.05	0.42 ± 0.05	0.98 ± 0.02	0.93 ± 0.03	0.85 ± 0.02	0.57 ± 0.04
RUS		0.89 ± 0.11	0.85 ± 0.21	0.55 ± 0.19	0.34 ± 0.15	0.82 ± 0.09	0.62 ± 0.10	0.47 ± 0.08	0.27 ± 0.08	0.56 ± 0.20	0.35 ± 0.24	0.21 ± 0.18	0.13 ± 0.11
SMT		0.96 ± 0.04	0.96 ± 0.04	**0.94** ± 0.04	0.90 ± 0.03	0.94 ± 0.04	**0.85** ± 0.02	0.65 ± 0.02	0.61 ± 0.02	0.96 ± 0.04	0.91 ± 0.01	0.90 ± 0.02	0.74 ± 0.03
RB		0.96 ± 0.02	0.96 ± 0.02	**0.94** ± 0.02	0.90 ± 0.01	0.91 ± 0.05	**0.85** ± 0.01	0.63 ± 0.01	0.60 ± 0.01	0.96 ± 0.03	0.92 ± 0.02	0.90 ± 0.02	0.73 ± 0.05
TUS		0.95 ± 0.05	0.95 ± 0.05	**0.94** ± 0.04	0.89 ± 0.04	0.83 ± 0.02	0.71 ± 0.02	**0.67** ± 0.02	0.62 ± 0.02	**0.99** ± 0.02	**0.96** ± 0.02	0.87 ± 0.02	0.77 ± 0.01
PRUS-F		0.90 ± 0.07	0.76 ± 0.10	0.75 ± 0.09	0.74 ± 0.09	0.99 ± 0.02	0.54 ± 0.12	0.45 ± 0.08	0.44 ± 0.08	0.72 ± 0.06	0.63 ± 0.16	0.62 ± 0.16	0.57 ± 0.13
PCUS-F		0.80 ± 0.10	0.57 ± 0.22	0.54 ± 0.21	0.53 ± 0.21	0.91 ± 0.11	0.55 ± 0.19	0.46 ± 0.13	0.42 ± 0.12	0.87 ± 0.07	0.83 ± 0.08	0.78 ± 0.09	0.65 ± 0.13
PTUS-F		**0.98** ± 0.01	0.93 ± 0.03	0.92 ± 0.03	**0.91** ± 0.03	**1.00** ± 0.00	0.83 ± 0.03	**0.67** ± 0.03	**0.63** ± 0.04	0.93 ± 0.05	0.92 ± 0.05	**0.92** ± 0.05	**0.81** ± 0.05

The boldface entries correspond to the best values of performance for each skew level of testing.

4.3.2 Results of Experiments with Video Data

Similarly to the synthetic data sets, the results of experiments on video datasets are shown in three parts, assessing the impact of: (1) using the F-measure loss factor on the performance of baseline Boosting ensembles, (2) integrating progressive partitioning into RUS, and (3) using the F-measure loss factor and progressive partitioning compared with the baseline and ensemble methods from the literature.

Table 4.9 Average of G-mean performance of proposed and baseline techniques on synthetic data over different levels of skew and overlap of test data.

Ensembles	Train Data	D_1 (Λ_{train} =1:50, $\delta = 0.2$)				D_2 (Λ_{train} =1:50, $\delta = 0.1$)				D_3 (Λ_{train} =1:20, $\delta = 0.2$)			
	Λ_{test}	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**
Ada		0.97 ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02	0.86 ± 0.05	0.86 ± 0.05	0.86 ± 0.05	0.85 ± 0.05	0.98 ± 0.02	**0.98** ± 0.02	**0.98** ± 0.02	0.97 ± 0.02
RUS		0.93 ± 0.05	0.93 ± 0.05	0.92 ± 0.05	0.91 ± 0.06	0.84 ± 0.08	0.83 ± 0.08	0.83 ± 0.08	0.82 ± 0.08	0.50 ± 0.20	0.68 ± 0.23	0.68 ± 0.23	0.68 ± 0.23
SMT		0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.94 ± 0.04	**0.94** ± 0.04	**0.94** ± 0.04	**0.94** ± 0.04	0.96 ± 0.04	0.96 ± 0.03	0.96 ± 0.04	0.96 ± 0.04
RB		0.96 ± 0.02	0.96 ± 0.02	0.96 ± 0.02	0.96 ± 0.02	0.91 ± 0.05	0.91 ± 0.04	0.90 ± 0.04	0.91 ± 0.05	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03	0.96 ± 0.03
TUS		0.95 ± 0.04	0.95 ± 0.04	0.95 ± 0.04	0.95 ± 0.04	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	0.84 ± 0.01	**0.99** ± 0.01	**0.98** ± 0.01	**0.98** ± 0.01	**0.98** ± 0.01
PRUS-F		0.90 ± 0.06	0.78 ± 0.08	0.78 ± 0.08	0.78 ± 0.08	0.99 ± 0.02	0.62 ± 0.09	0.62 ± 0.09	0.61 ± 0.09	0.78 ± 0.05	0.77 ± 0.05	0.76 ± 0.05	0.76 ± 0.06
PCUS-F		0.78 ± 0.20	0.65 ± 0.18	0.65 ± 0.18	0.65 ± 0.18	0.94 ± 0.07	0.68 ± 0.17	0.67 ± 0.17	0.64 ± 0.19	0.88 ± 0.06	0.86 ± 0.06	0.86 ± 0.06	0.86 ± 0.06
PTUS-F		**0.98** ± 0.01	0.93 ± 0.03	0.93 ± 0.03	0.93 ± 0.03	**1.00** ± 0.00	**0.94** ± 0.02	**0.94** ± 0.02	**0.94** ± 0.02	**0.94** ± 0.04	0.94 ± 0.04	0.94 ± 0.04	0.93 ± 0.05

The boldface entries correspond to the best values of performance for each skew level of testing.

Table 4.10 Average of AUPR performance of proposed and baseline techniques on synthetic data over different levels of skew and overlap of test data.

Ensembles	Train Data	D_1 (Λ_{train} =1:50, $\delta = 0.2$)				D_2 (Λ_{train} =1:50, $\delta = 0.1$)				D_3 (Λ_{train} =1:20, $\delta = 0.2$)			
	Λ_{test}	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**
Ada		**1.00** ± 0.00	**1.00** ± 0.00	0.98 ± 0.01	0.89 ± 0.04	**1.00** ± 0.00	0.82 ± 0.07	0.66 ± 0.05	0.34 ± 0.07	**1.00** ± 0.00	**0.99** ± 0.01	0.96 ± 0.01	0.70 ± 0.04
RUS		0.93 ± 0.21	0.90 ± 0.28	0.46 ± 0.25	0.29 ± 0.23	**1.00** ± 0.00	0.70 ± 0.13	0.40 ± 0.08	0.20 ± 0.08	0.64 ± 0.20	0.34 ± 0.33	0.20 ± 0.23	0.10 ± 0.11
SMT		0.90 ± 0.32	0.90 ± 0.32	0.89 ± 0.31	0.88 ± 0.31	0.90 ± 0.32	0.84 ± 0.30	0.52 ± 0.19	0.51 ± 0.19	**1.00** ± 0.00	**0.99** ± 0.01	**0.98** ± 0.01	0.88 ± 0.01
RB		0.80 ± 0.42	0.80 ± 0.42	0.79 ± 0.42	0.78 ± 0.41	0.60 ± 0.52	0.57 ± 0.49	0.32 ± 0.28	0.31 ± 0.27	**1.00** ± 0.00	**0.99** ± 0.01	**0.98** ± 0.01	0.84 ± 0.03
TUS		**1.00** ± 0.00	**1.00** ± 0.00	**1.00** ± 0.00	0.96 ± 0.01	0.99 ± 0.00	0.75 ± 0.04	**0.68** ± 0.01	**0.62** ± 0.02	**1.00** ± 0.00	**0.99** ± 0.01	0.96 ± 0.01	**0.92** ± 0.01
PRUS-F		**1.00** ± 0.00	**1.00** ± 0.00	0.99 ± 0.01	**0.97** ± 0.02	**1.00** ± 0.00	**0.86** ± 0.05	0.44 ± 0.04	0.35 ± 0.07	0.99 ± 0.01	**0.99** ± 0.01	0.90 ± 0.09	0.64 ± 0.15
PCUS-F		0.99 ± 0.04	0.92 ± 0.10	0.66 ± 0.27	0.47 ± 0.29	1.00 ± 0.00	0.57 ± 0.19	0.39 ± 0.15	0.27 ± 0.17	0.96 ± 0.07	0.87 ± 0.14	0.74 ± 0.17	0.53 ± 0.22
PTUS-F		**1.00** ± 0.00	**1.00** ± 0.00	0.98 ± 0.01	**0.97** ± 0.02	**1.00** ± 0.00	0.80 ± 0.09	0.55 ± 0.07	0.50 ± 0.09	**1.00** ± 0.00	0.98 ± 0.01	0.97 ± 0.01	0.85 ± 0.04

The boldface entries correspond to the best values of performance for each skew level of testing.

From Tables 4.11 and 4.14, the performance level of all ensembles is lower when the skew level of training data is higher. This is despite the fact that when the imbalance of training data is lower, the data that is used to test classifiers contain samples from some individuals that are

not in the training data. Using the F-measure loss factor improves the performance of Ada and RUS in terms of F-measure, and has less impact on the performance of RB and SMT in most cases of skew between classes in training and testing data. In terms of G-mean (Tables 4.12 and 4.15), the performance improves for all ensembles with both datasets, except for RUS with FIA. The performance of these ensembles after using the F-measure loss factor does not change significantly in terms of AUPR (Tables 4.13 and 4.16). In fact, the use of F-measure loss factor performs similarly to adjusting the decision threshold of the Boosting algorithms to better account for imbalance and therefore may improve the performance only in terms of local performance metrics like F-measure.

Table 4.11 Average of F_2-measure performance of baseline techniques before and after using the F-measure loss factor on FIA data sets over different levels of skew in training and test data.

Ensembles	**Train Data**	Λ_{train} =1:50				Λ_{train} =1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
Ada		0.80 ± 0.18	0.80 ± 0.18	0.80 ± 0.18	0.79 ± 0.18	0.62 ± 0.29	0.62 ± 0.29	0.62 ± 0.29	0.62 ± 0.29
Ada-F		**0.99** ± 0.01	**0.99** ± 0.01	0.75 ± 0.03	0.75 ± 0.02	**0.87** ± 0.09	**0.74** ± 0.08	0.60 ± 0.05	0.62 ± 0.08
RUS		0.98 ± 0.02	0.86 ± 0.10	0.72 ± 0.15	0.57 ± 0.16	0.98 ± 0.02	0.85 ± 0.10	0.70 ± 0.16	0.59 ± 0.17
RUS-F		0.96 ± 0.04	**0.89** ± 0.03	0.72 ± 0.03	**0.58** ± 0.04	0.93 ± 0.04	0.85 ± 0.06	**0.71** ± 0.03	**0.60** ± 0.07
SMT		0.93 ± 0.04	0.93 ± 0.04	0.93 ± 0.04	0.92 ± 0.05	0.92 ± 0.05	0.92 ± 0.05	0.92 ± 0.05	0.92 ± 0.05
SMT-F		**0.98** ± 0.02	**0.98** ± 0.02	0.92 ± 0.01	0.90 ± 0.02	**0.94** ± 0.04	0.90 ± 0.02	0.90 ± 0.02	0.90 ± 0.02
RB		0.89 ± 0.24	0.89 ± 0.24	0.89 ± 0.24	0.87 ± 0.24	0.77 ± 0.37	0.77 ± 0.37	0.76 ± 0.37	0.76 ± 0.37
RB-F		**0.99** ± 0.01	**0.99** ± 0.01	0.88 ± 0.01	0.85 ± 0.03	**0.91** ± 0.03	**0.85** ± 0.02	0.75 ± 0.02	0.74 ± 0.03

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

After integrating the progressive partitioning in RUS using PRUS and PTUS, the performance of RUS improves and becomes more robust in terms of both F-measure and AUPR (see Tables 4.17 and 4.18), especially when TUS is used for partitioning. Validating base classifiers on different imbalance levels of imbalance result in more robust classification systems and using

Table 4.12 Average of G-mean performance of baseline techniques before and after using the F-measure loss factor on FIA data sets over different levels of skew in training and test data.

Ensembles	Train Data	Λ_{train} =1:50				Λ_{train} =1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
Ada		0.82 ± 0.16	0.82 ± 0.16	0.82 ± 0.16	0.82 ± 0.16	0.67 ± 0.25	0.67 ± 0.25	0.67 ± 0.25	0.67 ± 0.25
Ada-F		**0.99** ± 0.01	**0.99** ± 0.01	**0.97** ± 0.01	**0.97** ± 0.01	**0.88** ± 0.08	**0.87** ± 0.07	**0.87** ± 0.07	**0.87** ± 0.07
RUS		0.97 ± 0.05	0.97 ± 0.02	0.97 ± 0.02	0.97 ± 0.02	0.95 ± 0.06	0.97 ± 0.02	0.97 ± 0.02	0.97 ± 0.02
RUS-F		0.96 ± 0.04	0.96 ± 0.04	0.95 ± 0.03	0.95 ± 0.03	0.93 ± 0.04	0.92 ± 0.04	0.92 ± 0.04	0.92 ± 0.04
SMT		0.94 ± 0.04	0.94 ± 0.04	0.94 ± 0.04	0.94 ± 0.04	0.92 ± 0.04	0.92 ± 0.04	0.92 ± 0.04	0.92 ± 0.04
SMT-F		**0.98** ± 0.02	**0.98** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02	**0.94** ± 0.03	**0.94** ± 0.03	**0.93** ± 0.03	**0.94** ± 0.03
RB		0.89 ± 0.24	0.89 ± 0.24	0.89 ± 0.24	0.89 ± 0.24	0.77 ± 0.37	0.77 ± 0.37	0.77 ± 0.37	0.77 ± 0.37
RB-F		**0.99** ± 0.01	**0.99** ± 0.01	**0.98** ± 0.01	**0.98** ± 0.01	**0.91** ± 0.03	**0.91** ± 0.03	**0.90** ± 0.03	**0.91** ± 0.03

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

all samples for training through partitioning avoids loss of information and may improve the classification accuracy.

Comparing the performance of final PBoost variants with baseline ensembles in Tables 4.19, 4.20, 4.21 and 4.22, 4.23, 4.24, PTUS-F outperforms all other approaches in terms of F-measure. In terms of G-mean RUS performs the best and in terms of AUPR, SMT has the highest mean value. From these results, it is observed that combining the use of F-measure and integration of progressive partitioning, is more effective in increasing performance and robustness compared to using each of them independently. In the experiments on the video data, trajectory under-sampling is more effective when used in PBoost compared to random under-sampling without replacement and cluster under-sampling. This is the case when partitions of negative class are known a priori.

Table 4.13 Average of AUPR performance of baseline techniques before and after using the F-measure loss factor on FIA data sets over different levels of skew in training and test data.

Ensembles	Train Data	Λ_{train} =1:50				Λ_{train} =1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
Ada		0.85 ± 0.35	0.82 ± 0.34	0.80 ± 0.33	0.77 ± 0.33	0.83 ± 0.36	0.79 ± 0.35	0.76 ± 0.35	0.74 ± 0.34
Ada-F		**1.00** ± 0.00	**1.00** ± 0.00	0.80 ± 0.06	0.77 ± 0.05	**1.00** ± 0.00	**0.81** ± 0.08	0.75 ± 0.13	0.72 ± 0.14
RUS		0.91 ± 0.29	0.88 ± 0.29	0.86 ± 0.28	0.85 ± 0.28	0.90 ± 0.30	0.88 ± 0.30	0.86 ± 0.29	0.85 ± 0.29
RUS-F		**1.00** ± 0.00	**0.96** ± 0.02	0.85 ± 0.04	0.84 ± 0.04	**1.00** ± 0.00	0.88 ± 0.11	0.86 ± 0.07	0.84 ± 0.09
SMT		1.00 ± 0.00	0.99 ± 0.01	0.99 ± 0.01	0.98 ± 0.02	1.00 ± 0.00	0.99 ± 0.01	0.99 ± 0.01	0.98 ± 0.02
SMT-F		**1.00** ± 0.00	**1.00** ± 0.00	0.96 ± 0.02	0.99 ± 0.01	**1.00** ± 0.00	0.93 ± 0.03	0.98 ± 0.04	0.97 ± 0.05
RB		0.93 ± 0.26	0.92 ± 0.25	0.92 ± 0.25	0.91 ± 0.25	0.81 ± 0.39	0.80 ± 0.39	0.80 ± 0.39	0.79 ± 0.39
RB-F		**1.00** ± 0.00	**1.00** ± 0.00	0.91 ± 0.03	0.90 ± 0.02	**1.00** ± 0.00	**0.94** ± 0.02	0.79 ± 0.03	0.79 ± 0.03

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

4.3.3 Statistical Comparison of the Classification Systems

The statistical comparison in this Chapter is carried out once for each skew level of test data. $K = 8$ is the number of classification systems in Tables 4.8, 4.17, 4.22, 4.23, and 4.24. $N = 8$ is the number of independent datasets used for statistical comparison that includes six synthetic datasets and two video datasets. The results of experiments with three of the synthetic datasets D_1, D_2 and D_3 have been presented in section 4.3.1. Remaining three datasets used for statistical comparison are $D_4(\Lambda_{\text{train}} = 1:20, \delta = 0.1)$, $D_5(\Lambda_{\text{train}} = 1:100, \delta = 0.2)$, $D_6(\Lambda_{\text{train}} = 1:100, \delta = 0.1)$. The data used for two settings of experiments on video data are dependent because all the samples that are used when $\Lambda_{\text{train}} = 1:50$ are used again when $\Lambda_{\text{train}} = 1:100$. Therefore, only the results with $\Lambda_{\text{train}} = 1:100$ are used for statistical comparison of the classification systems with FIA and COX datasets.

For each skew level of test data ($s = 1,...,4$), Table 4.25 shows the mean rank of each classification system ($R_k^s = (1/N)\sum_n r_{\text{n,s}}^k$ where $k = 1,...,K$ and $n = 1,...,N$). $r_{\text{n,s}}^k$ is the rank of k^{th}

Table 4.14 Average of F_2-measure performance of baseline techniques before and after using the F-measure loss factor on COX data sets over different levels of skew in training and test data.

Ensembles	Train Data	Λ_{train} =1:50				Λ_{train} =1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
Ada		0.48 ± 0.30	0.48 ± 0.30	0.48 ± 0.30	0.48 ± 0.30	0.43 ± 0.25	0.43 ± 0.25	0.43 ± 0.25	0.43 ± 0.25
Ada-F		**0.99** ± 0.01	**0.99** ± 0.01	**0.55** ± 0.03	0.47 ± 0.02	**0.87** ± 0.09	**0.74** ± 0.08	**0.58** ± 0.05	0.42 ± 0.08
RUS		0.89 ± 0.11	0.50 ± 0.22	0.43 ± 0.22	0.37 ± 0.24	0.87 ± 0.12	0.44 ± 0.19	0.40 ± 0.19	0.36 ± 0.20
RUS-F		**0.96** ± 0.04	**0.89** ± 0.03	**0.55** ± 0.03	**0.42** ± 0.04	**0.93** ± 0.04	**0.71** ± 0.06	**0.51** ± 0.03	0.36 ± 0.07
SMT		0.81 ± 0.15	0.80 ± 0.15	0.80 ± 0.15	0.80 ± 0.15	0.82 ± 0.12	0.82 ± 0.12	0.82 ± 0.12	0.82 ± 0.12
SMT-F		**0.98** ± 0.02	**0.98** ± 0.02	0.80 ± 0.01	0.80 ± 0.02	**0.94** ± 0.04	**0.85** ± 0.02	0.80 ± 0.02	0.80 ± 0.02
RB		0.83 ± 0.13	0.83 ± 0.13	0.83 ± 0.13	0.82 ± 0.13	0.82 ± 0.17	0.82 ± 0.17	0.82 ± 0.17	0.82 ± 0.17
RB-F		**0.99** ± 0.01	**0.99** ± 0.01	0.82 ± 0.01	0.81 ± 0.03	**0.91** ± 0.03	**0.85** ± 0.02	0.80 ± 0.02	0.79 ± 0.03

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

classification system for s^{th} skew level with n^{th} dataset such that best performing algorithm gets rank of 1 and the worst one gets rank of 8. In case of ties average ranks are assigned. From Table 4.25, it is observed that PTUS-F takes the highest rank in terms of F-measure and G-mean. To determine if there are any significant differences between the ranks, we use Friedman test and the subsequent version of Iman and Davenport (Iman & Davenport, 1980) which rejects the null hypothesis that all classification systems perform the same. Then we use Hochberg test (Hochberg, 1988) to compare the Boosting ensemble methods from the literature (Ada, RUS, SMT, RB and TUS) to PTUS-F. Table 4.26 shows the p-values of these comparisons. With α of 0.05, the boldface entries of the table indicate the rejection of null hypothesis that PTUS-F does not outperform these ensembles. It is observed that the null hypothesis is rejected in 14 out of 20 comparisons in terms of F-measure, and 6 out of 20 comparisons in terms of G-mean and AUPR.

Table 4.15 Average of G-mean performance of baseline techniques before and after using the F-measure loss factor on COX data sets over different levels of skew in training and test data.

Ensembles	Train Data	Λ_{train} =1:50				Λ_{train} =1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
Ada		0.56 ± 0.25	0.56 ± 0.25	0.56 ± 0.25	0.56 ± 0.25	0.52 ± 0.20	0.52 ± 0.20	0.52 ± 0.20	0.52 ± 0.20
Ada-F		**0.99** ± 0.01	**0.99** ± 0.01	**0.97** ± 0.01	**0.97** ± 0.01	**0.88** ± 0.08	**0.87** ± 0.07	**0.87** ± 0.07	**0.87** ± 0.07
RUS		0.85 ± 0.18	0.90 ± 0.07	0.94 ± 0.05	0.94 ± 0.05	0.83 ± 0.19	0.89 ± 0.08	0.93 ± 0.07	0.94 ± 0.07
RUS-F		**0.96** ± 0.04	**0.96** ± 0.04	**0.95** ± 0.03	**0.95** ± 0.03	**0.93** ± 0.04	**0.92** ± 0.04	0.92 ± 0.04	0.92 ± 0.04
SMT		0.83 ± 0.12	0.83 ± 0.12	0.83 ± 0.12	0.83 ± 0.12	0.84 ± 0.10	0.84 ± 0.10	0.84 ± 0.10	0.84 ± 0.10
SMT-F		**0.98** ± 0.02	**0.98** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02	**0.94** ± 0.03	**0.94** ± 0.03	**0.93** ± 0.03	**0.94** ± 0.03
RB		0.85 ± 0.11	0.85 ± 0.11	0.85 ± 0.11	0.85 ± 0.11	0.84 ± 0.16	0.84 ± 0.16	0.84 ± 0.16	0.84 ± 0.16
RB-F		**0.99** ± 0.01	**0.99** ± 0.01	**0.98** ± 0.01	**0.98** ± 0.01	**0.91** ± 0.03	**0.91** ± 0.03	**0.90** ± 0.03	**0.91** ± 0.03

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

4.3.4 Computational Complexity

In this section the time complexity needed to design and test the proposed and baseline Boosting ensembles are compared. To compare the training time and memory cost of these ensembles, the number of training samples is counted and to compare their validation time and memory cost the number of validation samples and the number of support vectors of base classifiers are considered.

Figure 4.5 show the results obtained with setting D_2 in our experiments. The number of training and validation samples, the average number of support vectors, and overall number of evaluations of the kernel function ($n_{\mathrm{SV}} \cdot n_{\mathrm{val}}$) is presented in Figure 4.5(a)-(d) to estimate and compare design time of the proposed and baseline Boosting ensembles. To compare the complexity of these classification systems during testing $O(n_{\mathrm{SV}})$ with a probe sample $\mathbf{x}$, we compared the overall number of support vectors in these ensembles in Figure 4.5(e) because computing each SVM output requires n_{SV} evaluations of the kernel function.

Table 4.16 Average of AUPR performance of baseline techniques before and after using the F-measure loss factor on COX data sets over different levels of skew in training and test data.

Ensembles	Train Data	Λ_{train} =1:50				Λ_{train} =1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
Ada		0.80 ± 0.36	0.66 ± 0.34	0.64 ± 0.34	0.60 ± 0.35	0.83 ± 0.32	0.65 ± 0.32	0.63 ± 0.33	0.61 ± 0.33
Ada-F		**1.00** ± 0.00	**1.00** ± 0.00	0.62 ± 0.06	0.60 ± 0.05	**1.00** ± 0.00	**0.81** ± 0.08	0.62 ± 0.13	0.60 ± 0.14
RUS		0.89 ± 0.28	0.81 ± 0.27	0.80 ± 0.27	0.79 ± 0.27	0.66 ± 0.46	0.57 ± 0.41	0.56 ± 0.41	0.56 ± 0.41
RUS-F		**1.00** ± 0.00	**0.96** ± 0.02	0.79 ± 0.04	0.77 ± 0.04	**1.00** ± 0.00	**0.75** ± 0.11	0.57 ± 0.07	0.56 ± 0.09
SMT		0.99 ± 0.02	0.95 ± 0.05	0.95 ± 0.05	0.91 ± 0.07	0.99 ± 0.02	0.97 ± 0.05	0.96 ± 0.05	0.96 ± 0.05
SMT-F		**1.00** ± 0.00	**1.00** ± 0.00	0.94 ± 0.02	0.90 ± 0.01	**1.00** ± 0.00	0.96 ± 0.03	0.95 ± 0.04	0.95 ± 0.05
RB		0.99 ± 0.03	0.94 ± 0.06	0.94 ± 0.06	0.91 ± 0.07	0.96 ± 0.17	0.93 ± 0.17	0.92 ± 0.17	0.92 ± 0.17
RB-F		**1.00** ± 0.00	**1.00** ± 0.00	0.93 ± 0.03	0.90 ± 0.02	**1.00** ± 0.00	**0.94** ± 0.02	0.91 ± 0.03	0.90 ± 0.03

The boldface entries correspond to improvement in the performance of the Boosting ensemble after modifying loss factor using F-measure.

Given n^e_{tr} as the number of samples to train the e^{th} classifier in the ensemble, Figure 4.5(a) shows $\sum_{e=1}^{E} n^e_{\text{tr}}$. In Figure 4.5(b), $\sum_{e=1}^{E} n^e_{\text{val}}$ is presented, where n^e_{val} is the number of samples that the e^{th} classifier in the ensemble is validated with. Average number of support vectors in E classifiers of the ensembles are shown in Figure 4.5(c). Given n^e_{SV} as the number of support vectors obtained after training the e^{th} classifier in the ensemble, Figure 4.5(d) shows $\sum_{e=1}^{E} n^e_{\text{val}} \cdot n^e_{\text{SV}}$ for each ensemble.

In terms of training (see Figure 4.5(a)), PTUS and RUS are under-sampling ensembles and have the lowest computational cost, while SMT and RB-Boost include up-sampling and are significantly more costly. Total number of validation samples is equal for Ada, SMT, RUS and RB, and total number of validation samples is less with PTUS (see Figure 4.5(b)). The average number of support vectors is higher for SMT (see Figure 4.5(c)) because the base classifiers in this ensemble are trained on higher number of samples. Therefore, SMT is the most costly method, in terms of validation (see Figure 4.5(d)). Note that time and memory required for partitioning in PTUS, and generating synthetic samples in SMT and RB-Boost

Table 4.17 Average of F_2-measure, G-mean and AUPR performance of RUSBoost with and without integrating progressive Boosting FIA data sets over different levels of skew in training and test data.

Ensembles	Train Data	Λ_{train} =1:50				Λ_{train} =1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
F_2-measure									
RUS		0.98 ± 0.02	0.86 ± 0.10	0.72 ± 0.15	0.57 ± 0.16	0.98 ± 0.02	0.85 ± 0.10	0.70 ± 0.16	0.59 ± 0.17
PRUS		**0.99** ± 0.01	**0.95** ± 0.04	**0.91** ± 0.06	**0.89** ± 0.06	**0.99** ± 0.01	**0.94** ± 0.04	**0.90** ± 0.06	**0.89** ± 0.07
PCUS		**0.99** ± 0.01	**0.95** ± 0.04	**0.91** ± 0.06	**0.88** ± 0.06	**0.99** ± 0.01	**0.95** ± 0.04	**0.91** ± 0.06	**0.89** ± 0.06
PTUS		**0.99** ± 0.01	**0.95** ± 0.04	**0.92** ± 0.05	**0.90** ± 0.06	**0.99** ± 0.01	**0.95** ± 0.04	**0.92** ± 0.06	**0.90** ± 0.06
G-mean									
RUS		0.97 ± 0.05	0.97 ± 0.02	0.97 ± 0.02	0.97 ± 0.02	0.95 ± 0.06	0.97 ± 0.02	0.97 ± 0.02	0.97 ± 0.02
PRUS		**0.98** ± 0.03	**0.97** ± 0.03	**0.95** ± 0.04	0.94 ± 0.04	**0.97** ± 0.04	**0.97** ± 0.03	0.95 ± 0.03	0.94 ± 0.04
PCUS		**0.98** ± 0.03	**0.97** ± 0.02	**0.95** ± 0.03	0.94 ± 0.04	**0.97** ± 0.04	**0.97** ± 0.02	0.95 ± 0.03	0.94 ± 0.04
PTUS		**0.98** ± 0.03	**0.97** ± 0.03	**0.95** ± 0.04	0.94 ± 0.04	**0.98** ± 0.03	**0.97** ± 0.03	0.95 ± 0.04	0.94 ± 0.04
AUPR									
RUS		0.91 ± 0.29	0.88 ± 0.29	0.86 ± 0.28	0.85 ± 0.28	0.90 ± 0.30	0.88 ± 0.30	0.86 ± 0.29	0.85 ± 0.29
PRUS		**1.00** ± 0.00	**0.95** ± 0.03	**0.93** ± 0.05	**0.91** ± 0.05	**1.00** ± 0.00	**0.95** ± 0.03	**0.93** ± 0.05	**0.91** ± 0.05
PCUS		**1.00** ± 0.00	**0.97** ± 0.03	**0.95** ± 0.04	**0.93** ± 0.05	**0.98** ± 0.14	**0.95** ± 0.14	**0.93** ± 0.14	**0.91** ± 0.14
PTUS		**0.98** ± 0.14	**0.96** ± 0.14	**0.94** ± 0.14	**0.92** ± 0.14	**0.97** ± 0.16	**0.95** ± 0.16	**0.93** ± 0.16	**0.91** ± 0.16

The boldface entries correspond to improvement in the performance of RUS after integrating progressive Boosting.

is neglected here. Nevertheless, PTUS is the most efficient ensemble technique in terms of designing memory and time complexity.

The number of training and validation samples as well as the average number of support vectors is smaller with PTUS and therefore, PTUS is less costly in terms of design time and memory complexity.

In terms of testing time complexity (see Figure 4.5(e)) PTUS and RUS have the lowest number of evaluations of the kernel function per probe sample. RB-Boost and SMT have the highest number of evaluations of the kernel function per probe sample.

Table 4.18 Average of F_2-measure, G-mean and AUPR performance of RUSBoost with and without integrating progressive Boosting COX data sets over different levels of skew in training and test data.

Ensembles	Train Data	Λ_{train} =1:50				Λ_{train} =1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
F_2-measure									
RUS		0.89 ± 0.11	0.50 ± 0.22	0.43 ± 0.22	0.37 ± 0.24	0.87 ± 0.12	0.44 ± 0.19	0.40 ± 0.19	0.36 ± 0.20
PRUS		**0.95** ± 0.06	**0.88** ± 0.10	**0.86** ± 0.09	**0.85** ± 0.10	**0.95** ± 0.06	**0.86** ± 0.13	**0.85** ± 0.13	**0.84** ± 0.13
PCUS		**0.95** ± 0.07	**0.88** ± 0.11	**0.85** ± 0.11	**0.84** ± 0.11	**0.95** ± 0.07	**0.85** ± 0.13	**0.84** ± 0.14	**0.84** ± 0.14
PTUS		**0.95** ± 0.06	**0.88** ± 0.10	**0.86** ± 0.10	**0.85** ± 0.11	**0.94** ± 0.07	**0.87** ± 0.12	**0.86** ± 0.12	**0.85** ± 0.13
G-mean									
RUS		0.85 ± 0.18	0.90 ± 0.07	0.94 ± 0.05	0.94 ± 0.05	0.83 ± 0.19	0.89 ± 0.08	0.93 ± 0.07	0.94 ± 0.07
PRUS		**0.95** ± 0.06	**0.92** ± 0.07	0.92 ± 0.07	0.91 ± 0.07	**0.95** ± 0.06	**0.91** ± 0.08	0.90 ± 0.09	0.90 ± 0.09
PCUS		**0.95** ± 0.07	**0.91** ± 0.08	0.90 ± 0.08	0.90 ± 0.08	**0.95** ± 0.08	**0.91** ± 0.09	0.90 ± 0.09	0.89 ± 0.10
PTUS		**0.95** ± 0.06	**0.91** ± 0.09	0.90 ± 0.09	0.88 ± 0.09	**0.94** ± 0.06	**0.91** ± 0.08	0.90 ± 0.09	0.89 ± 0.10
AUPR									
RUS		0.89 ± 0.28	0.81 ± 0.27	0.80 ± 0.27	0.79 ± 0.27	0.66 ± 0.46	0.57 ± 0.41	0.56 ± 0.41	0.56 ± 0.41
PRUS		**0.98** ± 0.04	**0.88** ± 0.11	**0.85** ± 0.11	**0.83** ± 0.11	**0.83** ± 0.21	**0.72** ± 0.23	**0.71** ± 0.23	**0.71** ± 0.23
PCUS		**0.97** ± 0.05	**0.88** ± 0.12	**0.87** ± 0.12	**0.85** ± 0.13	**0.83** ± 0.21	**0.72** ± 0.23	**0.70** ± 0.23	**0.70** ± 0.24
PTUS		**0.97** ± 0.16	**0.88** ± 0.17	**0.86** ± 0.18	**0.84** ± 0.18	**0.84** ± 0.35	**0.76** ± 0.33	**0.75** ± 0.33	**0.73** ± 0.33

The boldface entries correspond to improvement in the performance of RUS after integrating progressive Boosting.

Although AdaBoost is given the same ensemble size, it fails to generate enough classifiers and consequently result in smaller number of support vectors. Therefore, the total number of validation and testing processes of Ada is lower than expected.

4.3.5 Summary of Results

As a summary of results on synthetic and video datasets, we observed that: Using the proposed loss factor calculation may reduce the bias of performance in Boosting ensembles and increase the accuracy. Partitioning improves the performance of RUS in all cases in terms of both accuracy and robustness to imbalance. Integrating both partitioning and the proposed loss factor

Table 4.19 Average of F_2-measure performance of proposed and baseline techniques FIA data set over different levels of skew in training and test data.

Ensembles	Train Data	Λ_{train} =1:50				Λ_{train} =1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
Ada		0.80 ± 0.18	0.80 ± 0.18	0.80 ± 0.18	0.79 ± 0.18	0.62 ± 0.29	0.62 ± 0.29	0.62 ± 0.29	0.62 ± 0.29
RUS		0.98 ± 0.02	0.86 ± 0.10	0.72 ± 0.15	0.57 ± 0.16	0.98 ± 0.02	0.85 ± 0.10	0.70 ± 0.16	0.59 ± 0.17
SMT		0.93 ± 0.04	**0.93** ± 0.04	**0.93** ± 0.04	**0.92** ± 0.05	0.92 ± 0.05	0.92 ± 0.05	**0.92** ± 0.05	**0.92** ± 0.05
RB		0.89 ± 0.24	0.89 ± 0.24	0.89 ± 0.24	0.87 ± 0.24	0.77 ± 0.37	0.77 ± 0.37	0.76 ± 0.37	0.76 ± 0.37
TUS		0.63 ± 0.18	0.63 ± 0.17	0.62 ± 0.17	0.62 ± 0.17	0.87 ± 0.08	0.87 ± 0.08	0.87 ± 0.08	0.86 ± 0.08
PRUS-F		**0.99** ± 0.01	0.95 ± 0.05	0.92 ± 0.06	0.90 ± 0.06	**0.99** ± 0.01	**0.95** ± 0.04	0.91 ± 0.06	0.90 ± 0.06
PCUS-F		**0.99** ± 0.01	0.95 ± 0.04	0.92 ± 0.06	0.90 ± 0.06	**0.99** ± 0.02	**0.95** ± 0.04	0.91 ± 0.06	0.90 ± 0.07
PTUS-F		**0.99** ± 0.01	0.95 ± 0.04	0.92 ± 0.05	0.90 ± 0.06	**0.99** ± 0.01	**0.95** ± 0.04	**0.92** ± 0.06	0.90 ± 0.06

The boldface entries correspond to the best values of performance for each skew level of testing.

Table 4.20 Average of G-mean performance of proposed and baseline techniques FIA data set over different levels of skew in training and test data.

Ensembles	Train Data	Λ_{train} =1:50				Λ_{train} =1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
Ada		0.82 ± 0.16	0.82 ± 0.16	0.82 ± 0.16	0.82 ± 0.16	0.67 ± 0.25	0.67 ± 0.25	0.67 ± 0.25	0.67 ± 0.25
RUS		0.97 ± 0.05	**0.97** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02	0.95 ± 0.06	**0.97** ± 0.02	**0.97** ± 0.02	**0.97** ± 0.02
SMT		0.94 ± 0.04	0.94 ± 0.04	0.94 ± 0.04	0.94 ± 0.04	0.92 ± 0.04	0.92 ± 0.04	0.92 ± 0.04	0.92 ± 0.04
RB		0.89 ± 0.24	0.89 ± 0.24	0.89 ± 0.24	0.89 ± 0.24	0.77 ± 0.37	0.77 ± 0.37	0.77 ± 0.37	0.77 ± 0.37
TUS		0.49 ± 0.15	0.75 ± 0.13	0.76 ± 0.14	0.76 ± 0.14	0.88 ± 0.07	0.88 ± 0.07	0.88 ± 0.07	0.88 ± 0.07
PRUS-F		**0.98** ± 0.03	**0.97** ± 0.03	0.95 ± 0.04	0.94 ± 0.04	**0.98** ± 0.04	**0.97** ± 0.03	0.95 ± 0.03	0.94 ± 0.04
PCUS-F		**0.98** ± 0.03	**0.97** ± 0.02	0.95 ± 0.04	0.94 ± 0.04	0.97 ± 0.04	**0.97** ± 0.02	0.95 ± 0.03	0.94 ± 0.04
PTUS-F		**0.98** ± 0.02	**0.97** ± 0.03	0.95 ± 0.04	0.94 ± 0.04	**0.98** ± 0.03	**0.97** ± 0.02	0.95 ± 0.04	0.94 ± 0.04

The boldface entries correspond to the best values of performance for each skew level of testing.

outperforms the Boosting ensembles from the literature, relying on the choice of partitioning technique for each dataset such that: With synthetic data, PTUS-F outperforms all systems in terms of both F-measure and AUPR, while PRUS and PCUS outperform RUS in most cases

Table 4.21 Average of AUPR performance of proposed and baseline techniques FIA data set over different levels of skew in training and test data.

Ensembles	Train Data	Λ_{train} =1:50				Λ_{train} =1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
Ada		0.85 ± 0.35	0.82 ± 0.34	0.80 ± 0.33	0.77 ± 0.33	0.83 ± 0.36	0.79 ± 0.35	0.76 ± 0.35	0.74 ± 0.34
RUS		0.91 ± 0.29	0.88 ± 0.29	0.86 ± 0.28	0.85 ± 0.28	0.90 ± 0.30	0.88 ± 0.30	0.86 ± 0.29	0.85 ± 0.29
SMT		**1.00** ± 0.00	**0.98** ± 0.01	**0.96** ± 0.01	**0.94** ± 0.02	**1.00** ± 0.00	**0.99** ± 0.01	**0.97** ± 0.01	0.94 ± 0.02
RB		0.93 ± 0.26	0.92 ± 0.25	0.92 ± 0.25	0.91 ± 0.25	0.81 ± 0.39	0.80 ± 0.39	0.80 ± 0.39	0.79 ± 0.39
TUS		0.62 ± 0.15	0.62 ± 0.15	0.61 ± 0.15	0.60 ± 0.15	**1.00** ± 0.00	**0.99** ± 0.02	0.97 ± 0.03	**0.96** ± 0.03
PRUS-F		**1.00** ± 0.00	0.97 ± 0.04	0.95 ± 0.05	0.94 ± 0.06	**1.00** ± 0.00	0.98 ± 0.03	0.95 ± 0.04	0.94 ± 0.05
PCUS-F		**1.00** ± 0.00	0.98 ± 0.03	0.95 ± 0.05	0.93 ± 0.05	**1.00** ± 0.00	0.97 ± 0.03	0.95 ± 0.05	0.93 ± 0.06
PTUS-F		**1.00** ± 0.00	0.98 ± 0.03	0.96 ± 0.04	0.94 ± 0.05	**1.00** ± 0.00	0.98 ± 0.03	0.95 ± 0.04	0.94 ± 0.05

The boldface entries correspond to the best values of performance for each skew level of testing.

Table 4.22 Average of F_2-measure performance of proposed and baseline techniques COX data set over different levels of skew in training and test data.

Ensembles	Train Data	Λ_{train} =1:50				Λ_{train} =1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
Ada		0.48 ± 0.30	0.48 ± 0.30	0.48 ± 0.30	0.48 ± 0.30	0.43 ± 0.25	0.43 ± 0.25	0.43 ± 0.25	0.43 ± 0.25
RUS		0.89 ± 0.11	0.50 ± 0.22	0.43 ± 0.22	0.37 ± 0.24	0.87 ± 0.12	0.44 ± 0.19	0.40 ± 0.19	0.36 ± 0.20
SMT		0.81 ± 0.15	0.80 ± 0.15	0.80 ± 0.15	0.80 ± 0.15	0.82 ± 0.12	0.82 ± 0.12	0.82 ± 0.12	0.82 ± 0.12
RB		0.83 ± 0.13	0.83 ± 0.13	0.83 ± 0.13	0.82 ± 0.13	0.82 ± 0.17	0.82 ± 0.17	0.82 ± 0.17	0.82 ± 0.17
TUS		0.58 ± 0.27	0.49 ± 0.23	0.49 ± 0.23	0.48 ± 0.23	0.59 ± 0.27	0.59 ± 0.27	0.59 ± 0.27	0.59 ± 0.27
PRUS-F		0.94 ± 0.07	0.87 ± 0.13	0.86 ± 0.14	0.85 ± 0.14	**0.95** ± 0.06	0.87 ± 0.12	0.86 ± 0.13	0.85 ± 0.14
PCUS-F		**0.95** ± 0.07	0.85 ± 0.16	0.84 ± 0.17	0.82 ± 0.18	**0.95** ± 0.07	0.85 ± 0.16	0.84 ± 0.17	0.83 ± 0.17
PTUS-F		**0.95** ± 0.06	**0.89** ± 0.11	**0.87** ± 0.11	**0.86** ± 0.11	0.94 ± 0.06	**0.88** ± 0.11	**0.87** ± 0.11	**0.86** ± 0.11

The boldface entries correspond to the best values of performance for each skew level of testing.

of skew and overlap between classes. With the video data, PTUS is more accurate than the Boosting ensembles from the literature, PRUS as well as PCUS. PBoost is computationally less costly than the Boosting ensembles from the literature in terms of computational complex-

Table 4.23 Average of G-mean performance of proposed and baseline techniques COX data set over different levels of skew in training and test data.

Ensembles	Train Data	Λ_{train} =1:50				Λ_{train} =1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
G-mean									
Ada		0.56 ± 0.25	0.56 ± 0.25	0.56 ± 0.25	0.56 ± 0.25	0.52 ± 0.20	0.52 ± 0.20	0.52 ± 0.20	0.52 ± 0.20
RUS		0.85 ± 0.18	0.90 ± 0.07	**0.94** ± 0.05	**0.94** ± 0.05	0.83 ± 0.19	0.89 ± 0.08	**0.93** ± 0.07	**0.94** ± 0.07
SMT		0.83 ± 0.12	0.83 ± 0.12	0.83 ± 0.12	0.83 ± 0.12	0.84 ± 0.10	0.84 ± 0.10	0.84 ± 0.10	0.84 ± 0.10
RB		0.85 ± 0.11	0.85 ± 0.11	0.85 ± 0.11	0.85 ± 0.11	0.84 ± 0.16	0.84 ± 0.16	0.84 ± 0.16	0.84 ± 0.16
TUS		0.70 ± 0.21	0.68 ± 0.20	0.69 ± 0.21	0.69 ± 0.21	0.71 ± 0.21	0.71 ± 0.21	0.71 ± 0.21	0.71 ± 0.21
PRUS-F		0.94 ± 0.08	**0.91** ± 0.08	0.91 ± 0.08	0.90 ± 0.09	**0.95** ± 0.07	**0.92** ± 0.07	0.91 ± 0.08	0.90 ± 0.09
PCUS-F		**0.95** ± 0.07	**0.91** ± 0.08	0.90 ± 0.08	0.89 ± 0.08	**0.95** ± 0.08	0.91 ± 0.09	0.91 ± 0.09	0.90 ± 0.09
PTUS-F		**0.95** ± 0.08	**0.91** ± 0.09	0.90 ± 0.09	0.89 ± 0.09	0.94 ± 0.07	0.91 ± 0.08	0.90 ± 0.08	0.90 ± 0.09

The boldface entries correspond to the best values of performance for each skew level of testing.

Table 4.24 Average of AUPR performance of proposed and baseline techniques COX data set over different levels of skew in training and test data.

Ensembles	Train Data	Λ_{train} =1:50				Λ_{train} =1:100			
	Λ_{test}	1:1	1:20	1:50	1:100	1:1	1:20	1:50	1:100
Ada		0.80 ± 0.36	0.66 ± 0.34	0.64 ± 0.34	0.60 ± 0.35	0.83 ± 0.32	0.65 ± 0.32	0.63 ± 0.33	0.61 ± 0.33
RUS		0.89 ± 0.28	0.81 ± 0.27	0.80 ± 0.27	0.79 ± 0.27	0.66 ± 0.46	0.57 ± 0.41	0.56 ± 0.41	0.56 ± 0.41
SMT		**0.99** ± 0.02	**0.95** ± 0.05	**0.95** ± 0.05	**0.91** ± 0.07	**0.99** ± 0.02	**0.97** ± 0.05	**0.96** ± 0.05	**0.96** ± 0.05
RB		**0.99** ± 0.03	0.94 ± 0.06	0.94 ± 0.06	**0.91** ± 0.07	0.96 ± 0.17	0.93 ± 0.17	0.92 ± 0.17	0.92 ± 0.17
TUS		**0.99** ± 0.02	0.40 ± 0.16	0.38 ± 0.17	0.37 ± 0.17	**0.99** ± 0.01	0.95 ± 0.05	0.93 ± 0.06	0.91 ± 0.08
PRUS-F		0.97 ± 0.06	0.88 ± 0.14	0.87 ± 0.14	0.86 ± 0.14	0.98 ± 0.04	0.88 ± 0.12	0.87 ± 0.14	0.86 ± 0.14
PCUS-F		0.97 ± 0.06	0.85 ± 0.18	0.83 ± 0.19	0.82 ± 0.20	0.97 ± 0.06	0.85 ± 0.18	0.83 ± 0.19	0.83 ± 0.20
PTUS-F		**0.99** ± 0.02	0.91 ± 0.09	0.89 ± 0.10	0.87 ± 0.11	**0.99** ± 0.02	0.91 ± 0.09	0.89 ± 0.11	0.88 ± 0.12

The boldface entries correspond to the best values of performance for each skew level of testing.

ity. Therefore, PBoost is an effective approach in correct classification of data when data is imbalanced in comparison to the Boosting ensembles from the literature especially for face re-

Table 4.25 Average ranking of the performance of proposed and baseline techniques.

Ensembles	Train Data	F-measure				G-mean				AUPR			
	Λ_{test}	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**
Ada		4.50	3.25	5.38	5.75	4.50	3.38	3.38	3.38	2.88	3.62	4.25	4.50
RUS		6.62	6.12	6.88	7.75	6.62	5.50	5.12	5.12	3.50	5.81	6.75	7.50
SMT		4.25	3.12	2.50	2.62	4.25	3.25	3.25	**3.12**	2.12	2.38	**2.12**	**2.00**
RB		4.38	3.38	3.88	4.12	4.50	3.38	3.38	3.50	5.56	6.31	5.62	5.00
TUS		5.12	4.38	4.00	3.88	5.12	4.25	4.25	4.12	2.31	4.06	3.38	3.00
PRUS-F		3.50	6.38	5.62	4.88	3.62	6.38	6.50	6.75	1.62	**2.00**	3.75	3.25
PCUS-F		5.88	6.50	6.12	5.38	5.62	6.88	7.00	6.62	3.00	5.94	6.88	6.88
PTUS-F		**1.75**	**2.88**	**1.62**	**1.62**	**1.75**	**3.00**	**3.12**	3.38	**1.50**	3.38	3.25	3.88

Table 4.26 P-values of statistical comparison between PTUS-F and baseline techniques.

Ensembles	Metric	F-measure				G-mean				AUPR			
	Λ_{test}	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**	**1:1**	**1:20**	**1:50**	**1:100**
Ada		**0.0125**	0.3821	**0.0011**	**0.0004**	0.1250	0.3821	0.4207	0.5000	0.1314	0.4207	0.2090	0.3050
RUS		**0.0010**	**0.0040**	**0.0000**	**0.0000**	**0.0000**	**0.0207**	0.0516	0.0778	0.0516	**0.0233**	**0.0022**	**0.0016**
SMT		**0.0207**	0.4207	0.2389	**0.0000**	**0.0207**	0.4207	0.4602	0.5000	0.3050	0.5000	0.5000	0.5000
RB		**0.0162**	0.3446	**0.0336**	0.2090	**0.0125**	0.3821	0.4207	**0.0398**	**0.0005**	**0.0084**	**0.0268**	0.1814
TUS		**0.0030**	0.3888	**0.0268**	**0.0336**	**0.0030**	0.1539	0.1814	0.2709	0.2546	0.2877	0.4602	0.5000

identification. This method relies on the choice of partitioning technique for each dataset and performs significantly better when a more suitable partitioning technique is used. In problems that the natural clusters are known, as with trajectory under-sampling in face re-identification application, the performance is better than using the general partitioning methods such as random under-sampling without replacement or k-means clustering. Therefore, PBoost can be more efficient than baseline Boosting ensembles for face re-identification under imbalance considering both accuracy and complexity factors. Note that the proposed PBoost algorithm can be implemented with any discriminative classifier.

4.4 Conclusion

In this Chapter, a new Boosting ensemble algorithm named as PBoost is proposed to address imbalance based on the idea of modifying RUSBoost by (1) under-sampling the majority class using partitional techniques and in particular trajectory-based partitioning, inspired by face re-identification applications, (2) validating classifiers on a growing validation subset, and (3)

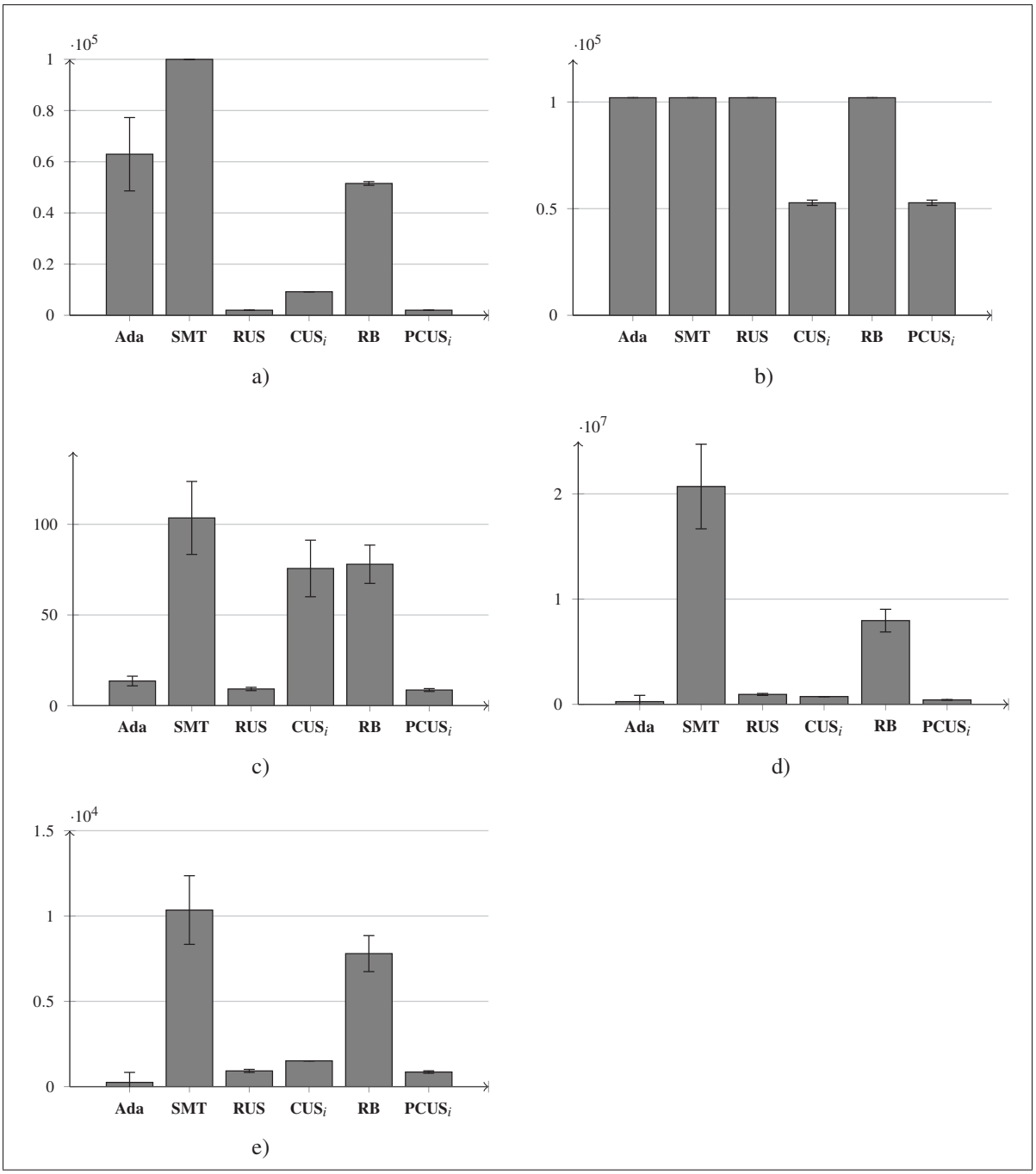


Figure 4.5 Complexity related to the design and testing process of ensembles: (a) total number of training samples, (b) total number of validation samples, (c) Number of $n_{\mathrm{SV}} \cdot n_{\mathrm{val}}$ during validation, (d) total number of $n_{\mathrm{SV}} \cdot n_{\mathrm{val}}$ during validation, (e) total number of evaluations of the kernel function per probe sample during testing.

using a more suitable loss factor calculation. The partitions enter the Boosting process progressively for designing classifiers over iterations to avoid information loss and to maintain diversity among them. Validating base classifiers on a growing number of negative samples makes the PBoost ensembles more robust to possible skew levels of data during operations in addition to lowering the computational complexity. The loss factor based on F-measure handles bias of performance towards negative class, and guides the Boosting process in a more effective direction with the purpose of correctly classifying both classes. Experiments show that PBoost may perform differently with different techniques of partitioning for each dataset such that more suitable partitioning result in better performance. For face re-identification application, the PBoost ensemble using trajectory under-sampling outperforms the Boosting ensembles from the literature in terms of accuracy. In addition, PBoost has significantly lower computational complexity in both designing and testing stages compared to the Boosting ensembles from the literature. The applicability of PBoost to multi-class classification problems and other real-world applications can be further investigated in a future work.

CHAPTER 5

F-MEASURE CURVES: A TOOL TO VISUALIZE CLASSIFIER PERFORMANCE UNDER IMBALANCE

Evaluating classification performance is an important step, for both guiding the learning process and comparing systems to each other. Classification systems are usually trained over a number of iterations, and the direction of the design process in each iteration depends on the outcome of the classifier(s) in the previous iteration(s). As an example, in Boosting ensembles, the error of the classifier in each iteration affects the sample selection in the next iteration as well as the final prediction function. What's more, after designing any classification system, its performance should be compared to alternative systems for the problem in hand.

This aspect becomes more critical in the case of class imbalance, since in this case the most widely used performance metric, classification accuracy, tends to favour the correct classification of the most populated class (or classes). This is an issue in many machine learning applications, where the number of available samples from the class of interest ("positive", or "target" class) is heavily outnumbered by other classes especially in two-class classification problems. Since the objective functions of many standard learning algorithms (e.g., support vector machines) seek to maximize unsuitable performance metrics for imbalance, the trained classifiers become biased towards correctly recognizing the majority ("negative" or "non-target" class) at the expense of high misclassification rates for the positive class. On the other hand, the widely used ROC curve, area under ROC curve and G-mean favour the correct classification of positive samples, in expense of excessive misclassification of negative samples. When data is highly imbalanced, a change in the number of correctly classified positive samples (TP) and the number of misclassified negative samples (FP), reflect in a more significant change in the true positive rate (TPR) compared to the change in false positive rate (FPR).

To address this issue, other performance metrics like the expected cost (EC) and the F-measure are being used in imbalanced data classification. Such metrics follow different objectives in terms of favouring the correct classification of positive samples, and of avoiding the opposite

drawback of allowing excessive misclassification of negative samples. The choice between them is therefore application-dependent. When the performance measure is EC, two graphical techniques have been proposed to easily visualize and compare classifier performance under all possible operational points, i.e. class prior probabilities and misclassification costs (or relative preference of classes): Cost curves (CC) (Drummond & Holte, 2006) and Brier curves (BC) (Ferri *et al.*, 2011). These plots exhibit several advantages over the traditional, well-known Receiver Operating Characteristic (ROC) plot since ROC curves are independent of class imbalance (Davis & Goadrich, 2006). Precision-Recall (PR) space is in turn more suitable than ROC space for imbalance problem and plot the performance of the classifier in terms of precision vs. recall (or TPR) (Landgrebe *et al.*, 2006; Davis & Goadrich, 2006). The reason is that, precision is preferred to FPR when classifying imbalanced data because precision measures the proportion of TPR to the FPR multiplied by the skew level. However, PR curves are difficult to analyze when comparing classifiers under different skew levels of data because each skew level of data may result in a different curve and the curves in this space correspond to equal preference of classes.

The other widely used metric in information retrieval and class imbalance problems is F-measure that have been analyzed by many researchers (Pillai *et al.*, 2017; Dembczynski *et al.*, 2011; Lipton *et al.*, 2014; Parambath *et al.*, 2014). The main benefit of F-measure is that it compares the performance of the classifier in terms of recall (or TPR) to precision using a factor that controls their relative importance. However, no analogous to cost space performance visualization tool exists for the F-measure. We point out that in the Precision-Recall (PR) space, the F-measure is presented as hyperbolic isometrics (Hanczar & Nadif, 2013; Flach & Kull, 2015), and does not allow to easily visualize the F-measure of a given classifier under different operational conditions (different class priors and different preference of precision and recall).

In summary the main contribution of this chapter is proposing a new global performance evaluation space that allows evaluating performance directly, in terms of the scalar F-measure metric. To our knowledge, no performance visualization tool analogous to CC and BC exist for the F-measure. The proposed F-measure space has the following properties.

- Possibility of visualizing the performance of any classifier (soft or crisp) under different imbalance levels of deployment data.

- Possibility of selecting the best threshold of a classifier under the given imbalance level and preference between precision and recall.

- Possibility of comparing more than two classifiers over different decision thresholds and under different imbalance levels of test data with the ability of selecting a preference level between classes.

- Possibility of selecting the best combination of a set of classifiers based on their performance in the F-measure space. As the second contribution, the proposed F-measure space is used to modify the Iterative Boolean Combination (IBC) method to adapt the selection and combination of classifiers in the ensemble for an optimal performance under different operating conditions (imbalance levels).

- The F-measure space is preferred to the ROC and Precision-Recall spaces to compare classifiers under different imbalance levels and preference between classes.

- The F-measure space can be preferable to cost space in some applications when precision-recall is preferred to the misclassification cost like in information retrieval. In addition, the F-measure space is more sensitive to class imbalance and tuning the preference between classes results in a visible difference in performance in the F-measure space compared to the cost space.

5.1 The F-Measure Space

In this section, an alternative visualization tool is proposed that is analogous to cost curves for evaluating the F-measure of one classifier. This tool is used to compare classifiers under different operating conditions, that correspond to class priors and to the α parameter in the case of the F-measure. To this aim, we start by rewriting the F-measure from Eq. (1.14) as follows,

to make its dependence on $P(+)$ and on α explicit:

$$F_\alpha = \frac{\text{TPR}}{\alpha(\text{TPR} + \lambda \cdot \text{FPR}) + (1-\alpha)} \tag{5.1}$$

$$= \frac{{}^1\!/\!_\alpha \text{TPR}}{{}^1\!/\!_\alpha + {}^1\!/\!_{P(+)}\text{FPR} + \text{TPR} - \text{FPR} - 1} \tag{5.2}$$

As a reminder from section 1.2.4.3, $P(+)$ is the $P_{deploy(+)}$ in this context, with respect to a fixed (pre-trained) classifier. Contrary to the EC of Eq. (1.12), Eq. (5.2) shows that the F-measure cannot be written as a function of a single parameter ($PC(+)$ in case of EC) that takes into account both $P(+)$ and α. This means that the F_α values of a classifier should in principle be plotted as a 3D surface as a function of two distinct variables, $P(+)$ and α. However, this would not allow an easy visualization. Therefore, since the main focus of this Chapter is class imbalance, in the following we will consider a simpler plot of the F-measure as function of $P(+)$ only, for a fixed value of α.

5.1.1 F-measure curve of a classifier

Let us first consider the behavior of F_α for a given crisp classifier (i.e., for given TPR and FPR values), as a function of $P(+)$. From Eq. (5.2) one obtains that, when $P(+) = 0$, $F_\alpha = 0$, whereas when $P(+) = 1$, $F_\alpha = \text{TPR}/(\alpha(\text{TPR}-1)+1)$. It is then easy to see that the first derivative of F_α with respect to $P(+)$ is strictly positive; the second derivative is strictly negative when TPR $>$ FPR, which is always the case for a non-trivial classifier. Accordingly, the F-measure curve that corresponds to a given classifier is an increasing and concave function of $P(+)$.

For different values of α we get a family of curves. For $\alpha = 0$ we have $F_\alpha = \text{TPR}$ for any value of $P(+)$ and for $\alpha = 1$ we have $F_\alpha = \text{Pr}$. Therefore, for $0 < \alpha < 1$, each curve starts at $F_\alpha = 0$ for $P(+) = 0$ and ends in $F_\alpha = \text{Pr}$ for $P(+) = 1$.

By computing the first derivative of F_α (Eq. 5.2) with respect to α, for any fixed $P(+)$, one gets that its value is zero for $P(+) = FPR/(FPR - TPR + 1)$, it is negative for smaller $P(+)$

values, and it is positive for higher $P(+)$ values. This means that all curves (including the one for $\alpha = 0$) cross when $P(+) = FPR/(FPR - TPR + 1)$.

An example is shown in Fig. 5.1, for a classifier with TPR $= 0.8$ and FPR $= 0.15$, and for five values of α.

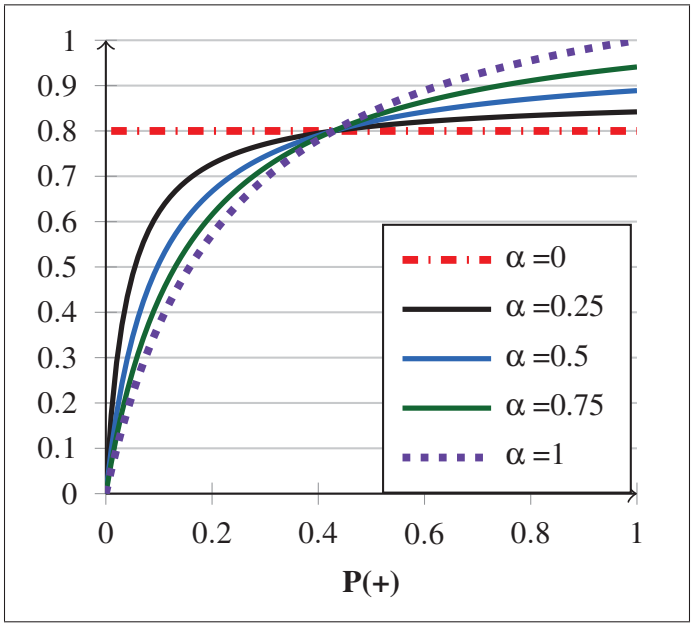


Figure 5.1 The F_α curves for a given classifier with TPR=0.8 and FPR=0.15, for different values of α. Note that for all values of $P(+)$ and $\alpha = 0$, $F_\alpha = TPR$.

Consider now the behavior of the F-measure curve for a given soft classifier and a given α value, when the decision threshold changes. Let us first recall that, as mentioned in Sect. 1.2.3.3, a point in the ROC space corresponds to a line in the cost space. Similarly, it corresponds to a (non-linear) curve in the F-measure space. As the decision threshold of a classifier changes, one obtains a curve in ROC space and a family of lines in cost space. Similarly, one also obtains a family of (non-linear) curves in F-measure space. More precisely, as the decision threshold increases from its maximum to its minimum (assuming that higher classifier scores correspond to a higher probability that the input sample is positive), TPR and FPR start at $TPR = 0$ and $FPR = 0$, and increase towards $TPR = 1$ and $FPR = 1$. For a given value of α, the corresponding curves in F-measure space move away from the Y axis and get closer

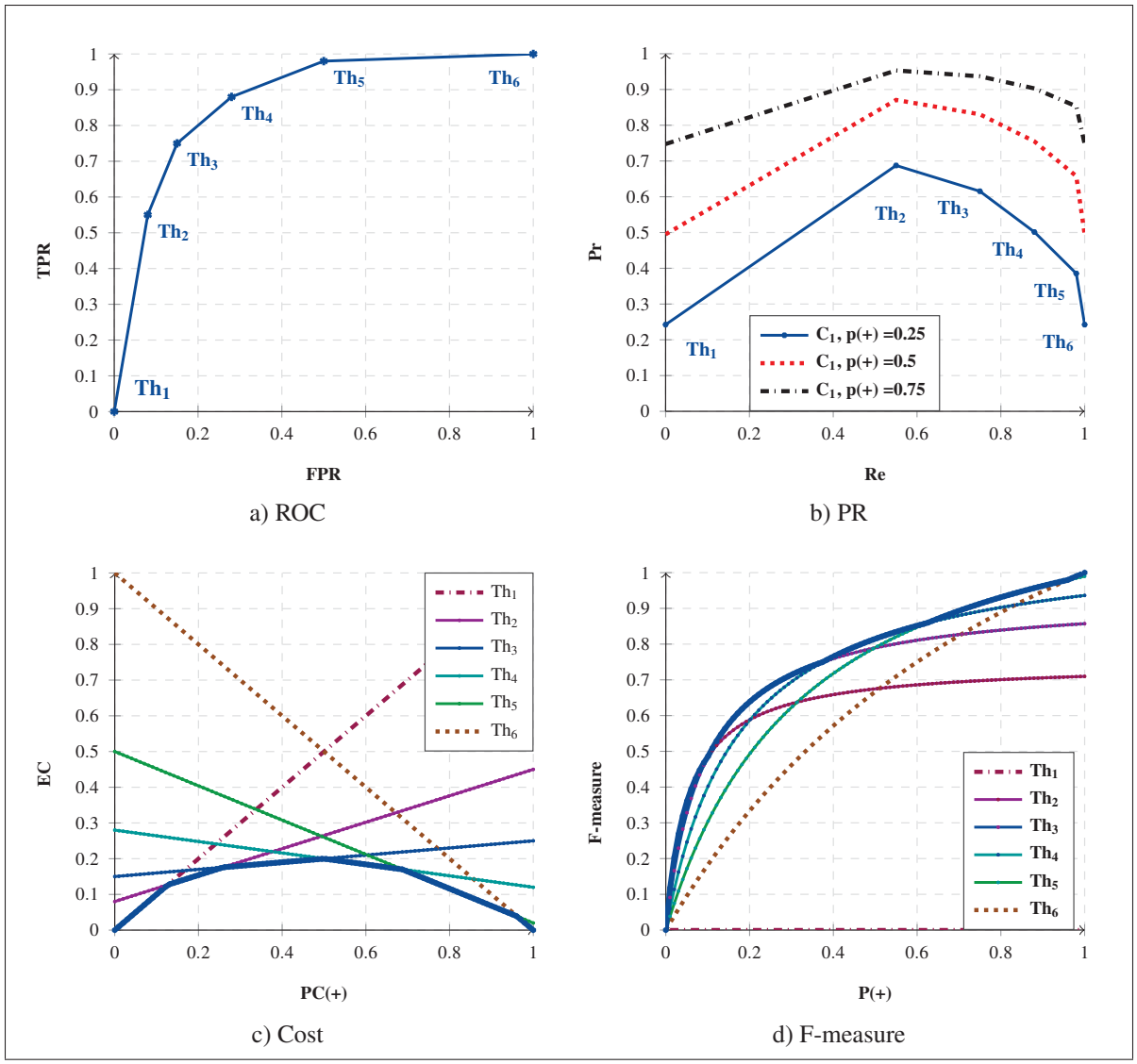
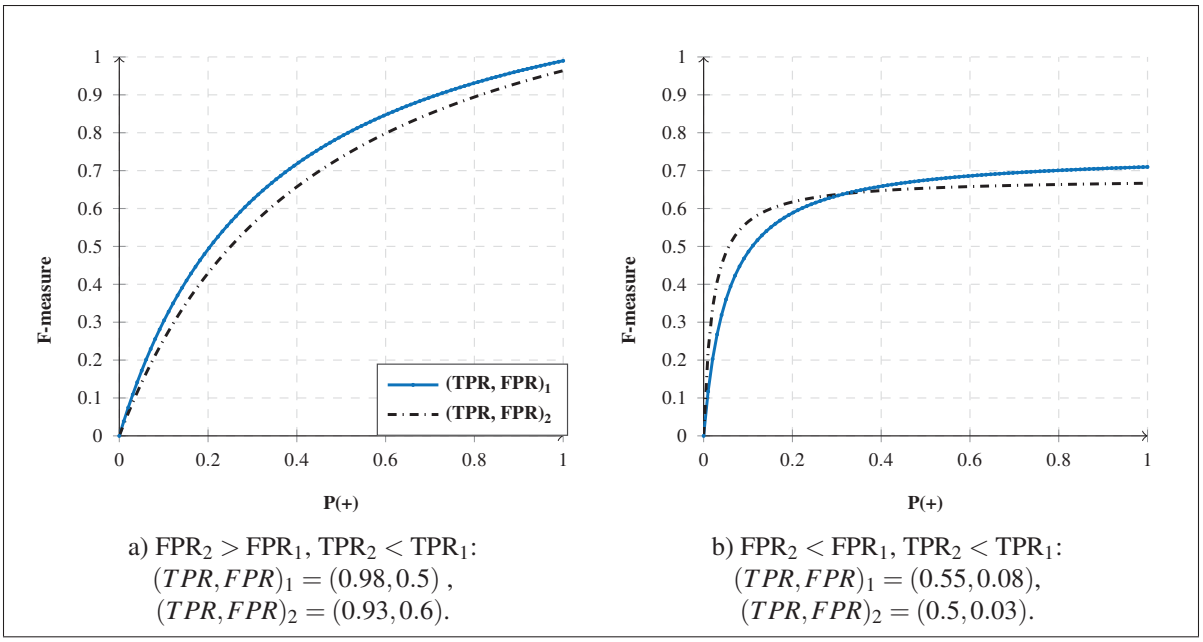


Figure 5.2 Performance of one soft classifier in ROC, inverted PR, Cost and F-measure spaces with $m = 0.5$ and $\alpha = 0.5$.

to the diagonal line connecting the lower-left point $P(+) = 0, F_\alpha = 0$ to the upper-right point $P(+) = 1, F_\alpha = 1$. This behavior is intuitive: as we move from (0,0) on the ROC curve towards the (1,1), both FPR and TPR are increasing. However, for a non-trivial classifier, the increase in TPR becomes less and less greater for a small increase of FPR. In other words, the slope of the tangent line to the ROC curve becomes steeper. The steeper the tangent line to the curve is, the better performance is achieved for the correct classification of positive samples when the class skew is high (smaller P(+)).



a) $\text{FPR}_2 > \text{FPR}_1$, $\text{TPR}_2 < \text{TPR}_1$:
$(TPR, FPR)_1 = (0.98, 0.5)$,
$(TPR, FPR)_2 = (0.93, 0.6)$.

b) $\text{FPR}_2 < \text{FPR}_1$, $\text{TPR}_2 < \text{TPR}_1$:
$(TPR, FPR)_1 = (0.55, 0.08)$,
$(TPR, FPR)_2 = (0.5, 0.03)$.

Figure 5.3 F-measure curves of two classifiers ($\alpha = 0.5$).

In the example shown in Fig. 5.2, we assume a classifier with six different threshold values ($\text{Th}_1 > \text{Th}_2 > \ldots > \text{Th}_6$), which correspond to $\text{TPR}_1 = [10^{-6}, 0.55, 0.75, 0.88, 0.98, 1]$, and $\text{FPR}_1 = [10^{-6}, 0.08, 0.15, 0.28, 0.5, 1]$. From top-left to bottom-right, Fig. 5.2 shows the convex hull of the corresponding ROC curve, the corresponding precision-recall curves for three values of $P(+)$, cost lines, and the F-measure curves (one for each point in ROC space) with $\alpha = 0.5$. From Eq. 1.13, corresponding precision to the points in ROC space, could vary based on the skew level of data, as seen in Fig. 5.2.

For any given operating condition, i.e., for each value of $P(+)$, it is clear that only one of the decision thresholds provides the highest F_α. Accordingly, among the curves that correspond to all the available pairs of (TPR, FPR) values of a soft classifier, their upper envelope shows the best performance of the classifier with the most suitable tuning of decision threshold for each operating condition.

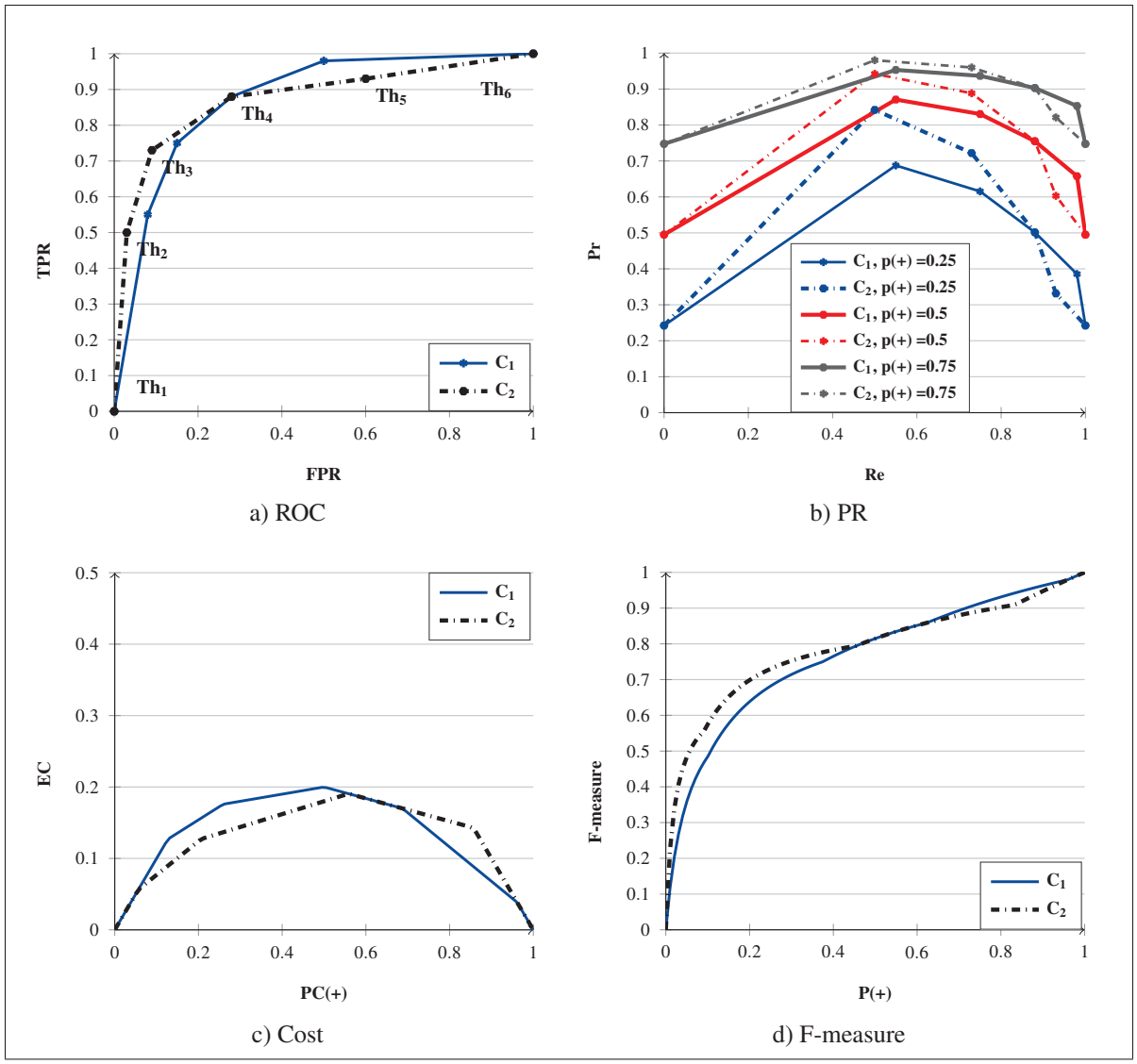


Figure 5.4 Performance of two soft classifiers in ROC, inverted PR, Cost and F-measure spaces with $m = 0.5$ and $\alpha = 0.5$.

5.1.2 Comparing classifiers in the F-measure space

We now discuss how two or more classifiers, characterized by given values of (TPR_i, FPR_i) and (TPR_j, FPR_j), etc. , can be compared in the F-measure space, for a fixed value of α. As explained above, the F-measure curve of any classifier starts at $F_\alpha = 0$ for $P(+) = 0$, whereas the one of a classifier characterized by (TPR_i, FPR_i) ends at $F_\alpha^i = TPR_i/[\alpha(TPR_i - 1) + 1]$ when $P(+) = 1$. Note that the latter value depends only on the TPR value, not on FPR.

It is easy to see that $F_\alpha^j > F_\alpha^i$ for all values of $P(+) > 0$, under three different conditions:

(i) when $\text{FPR}_i = \text{FPR}_j$ and $\text{TPR}_j > \text{TPR}_i$;

(ii) when $\text{TPR}_i = \text{TPR}_j$ and $\text{FPR}_j < \text{FPR}_i$;

(iii) when $\text{FPR}_j < \text{FPR}_i$ and $\text{TPR}_j > \text{TPR}_i$.

The analogous conditions under which $F_\alpha^2 < F_\alpha^1$ for all values of $P(+) > 0$ can be easily obtained. As an example, consider two classifiers with $\text{FPR}_1 < \text{FPR}_2$ and $\text{TPR}_1 > \text{TPR}_2$, where $(TPR, FPR)_1 = (0.98, 0.5)$ and $(TPR, FPR)_2 = (0.93, 0.6)$. The corresponding F-curves are shown in Fig. 5.3(a) ($\alpha = 0.5$). It can be seen that C_2 dominates C_1 for all values of $P(+)$.

Instead, if $\text{FPR}_j < \text{FPR}_i$ and $\text{TPR}_j < \text{TPR}_i$, or when $\text{FPR}_j > \text{FPR}_i$ and $\text{TPR}_j > \text{TPR}_i$, then the corresponding F-measure curves cross in a *single* point, with the following value of $P^*_{\text{i,j}}(+)$:

$$P^*_{\text{i,j}}(+) = \frac{\text{FPR}_i \cdot \text{TPR}_j - \text{FPR}_j \cdot \text{TPR}_i}{((\alpha-1)/\alpha)(\text{TPR}_j - \text{TPR}_i) + \text{FPR}_i \cdot \text{TPR}_j - \text{FPR}_j \cdot \text{TPR}_i} \,. \tag{5.3}$$

As an example, consider two classifiers with $\text{FPR}_1 > \text{FPR}_2$ and $\text{TPR}_1 > \text{TPR}_2$, where $(TPR, FPR)_1 = (0.55, 0.08)$ and $(TPR, FPR)_2 = (0.5, 0.03)$. The corresponding F-curves are shown in Fig. 5.3(b).

In particular, from Eq. (5.3), we can determine the exact range of $P(+)$ for the given α value when one classifier can outperform the other in terms of F-measure. We see that, if $\text{FPR}_i \cdot \text{TPR}_j = \text{FPR}_j \cdot \text{TPR}_i$, the curves cross only when $P(+) = 0$ ($P^*_{\text{i,j}}(+) = 0$), which means that the classifier with highest TPR and lowest FPR exhibits a higher value of F_α for all values of $P(+) > 0$. If $\text{FPR}_i \cdot \text{TPR}_j \neq \text{FPR}_j \cdot \text{TPR}_i$, the classifier with highest TPR and FPR exhibits a higher value of F_α for $P(+) > P^*_{\text{i,j}}(+)$ than Eq. (5.3), and the opposite happens for $P^*_{\text{i,j}}(+) < P(+)$.

Therefore, given a set of classifiers characterized by given values of (TPR_i, FPR_i), those that lie on the upper envelope of F-measure curves can be determined accurately because a classifier

$C_j(TPR_j, FPR_j)$ dominates $C_i(TPR_i, FPR_i)$ in contributing to the upper envelope F-curve for all $P(+) > P^*_{\mathrm{i,j}}(+)$ if and only if one of the following conditions hold:

(i) $\mathrm{FPR}_j < \mathrm{FPR}_i$ and $\mathrm{TPR}_j > \mathrm{TPR}_i$, $P^*_{\mathrm{i,j}}(+) = 0$;

(ii) $\mathrm{FPR}_j > \mathrm{FPR}_i$ and $\mathrm{TPR}_j > \mathrm{TPR}_i$, $P^*_{\mathrm{i,j}}(+) \neq 0$.

If these conditions hold, the classifiers lie on the ROC convex hull and the upper envelope of F-curves of the given classifiers is obtained from F-curves of all or a subset of classifiers that lie on the ROC convex hull. Note that two classifiers may both correspond to the upper envelope in F-measure space at $P^*_{\mathrm{i,j}}(+)$ if condition (ii) holds.

The overall performance of two or more soft classifiers can be easily compared by comparing the upper envelopes of their F-curves. An example of the comparison of two classifiers is shown in Fig. 5.4. Classifier C_1 is the same as in Fig. 5.2. Also for classifier C_2 we consider six different threshold values ($\mathrm{Th}_1 > \mathrm{Th}_2 > \ldots > \mathrm{Th}_6$), corresponding to $\mathrm{TPR}_2 = [10^{-6}, 0.5, 0.73, 0.88, 0.93, 1]$, and $\mathrm{FPR}_2 = [10^{-6}, 0.03, 0.09, 0.28, 0.6, 1]$. In Fig. 5.4(a), we show the convex hulls of the ROC curves of the two classifiers, which cross on a single point around $FPR = 0.3$. Fig. 5.4(b) shows the PR curve of these classifiers for three values of $P(+)$. It is difficult to make a statement to compare C_1 and C_2 about their performance for different skew levels of data from their PR curves.

In Fig. 5.4(c), the lower envelopes of the cost curves are compared. The cost curve of these classifiers cross when $PC(+)$ is close to 0.7, and thus C_1 and C_2 perform the same for approximately $0.6 < PC(+) < 0.7$, and C_1 outperforms C_2 for $PC(+) < 0.6$. From Eq. 1.17, the classifiers can be compared in this space for any given $P(+)$, C_{FN} and C_{FP}.

In Fig. 5.4(d), the upper envelopes of the F-measure curves of C_1 and C_2 are compared for $\alpha = 0.5$. From F-space, C_2 outperforms C_1 for $P(+) < 0.4$, whereas C_1 and C_2 perform the same for $0.4 < P(+) < 0.6$, and C_1 outperforms C_2 for $P(+) > 0.6$. These examples show that comparing the F-measure of two or more classifiers over all skew levels using the F-measure space is as easy as comparing their expected cost using the cost curves.

5.1.3 F-measure Space vs. Cost Space

As explained in section 1.2.3.3, cost space is usually used to compare classifiers' performance in terms of expected cost, given two cost factors C_{FP} and C_{FN}. In this chapter, a similar performance visualization space is proposed to compare classifiers' performance in terms of the F-measure. In this section, the similarities and dissimilarities of the proposed F-measure space and cost space are analysed. To investigate the analogy and the difference between the F-measure space and the cost space a factor m is defined for the expected cost calculation similar to α for the F-measure as follows:

$$m = \frac{C_{\mathrm{FP}}}{C_{\mathrm{FP}} + C_{\mathrm{FN}}}, \text{ where } 0 < m \leq 1 \tag{5.4}$$

m can take different values to weigh importance of positive and negative classes. For $m < 0.5$, $C_{\mathrm{FN}} > C_{\mathrm{FP}}$ (similar to $\alpha > 0.5$) and correct classification of positive class is considered more important. For $m = 0.5$, $C_{\mathrm{FP}} = C_{\mathrm{FN}}$ and correct classification of both classes becomes equally important. For $0.5 < m \leq 1$, $C_{\mathrm{FN}} < C_{\mathrm{FP}}$ (similar to $\alpha < 0.5$) and correct classification of positive class becomes less important. It should be noted that α and m are actually unrelated, so the pairs of values used in examples in the rest of the paper (where $m = \alpha$) have no particular meaning.

From 5.4 equation 1.17 is rewritten as:

$$PC(+) = \frac{(1/m - 1) \cdot P(+)}{(1/m - 2) \cdot P(+) + 1}. \tag{5.5}$$

For $m = 0$, $PC(+) = 1$ and $EC = 1 - TPR$, for $m = 0.5$, $PC(+) = P(+)$ and $EC = (1 - TPR - FPR)P(+) + FPR$, and for $m = 1$, $PC(+) = 0$ and $EC = FPR$. Figure 5.5 shows the expected cost of a single classifiers against $PC(+)$ and $P(+)$ for different values of m. Comparing Figure 5.5(a) with Figure 5.1 shows that the sensitivity of the F-measure to the correct classification of the positive and the negative classes can be adjusted by tuning α and therefore a classifier can result in different curves in the F-measure space depending on the

value of α. However, tuning m doesn't provide the same effect in the conventional cost space that depicts EC against $PC(+)$.

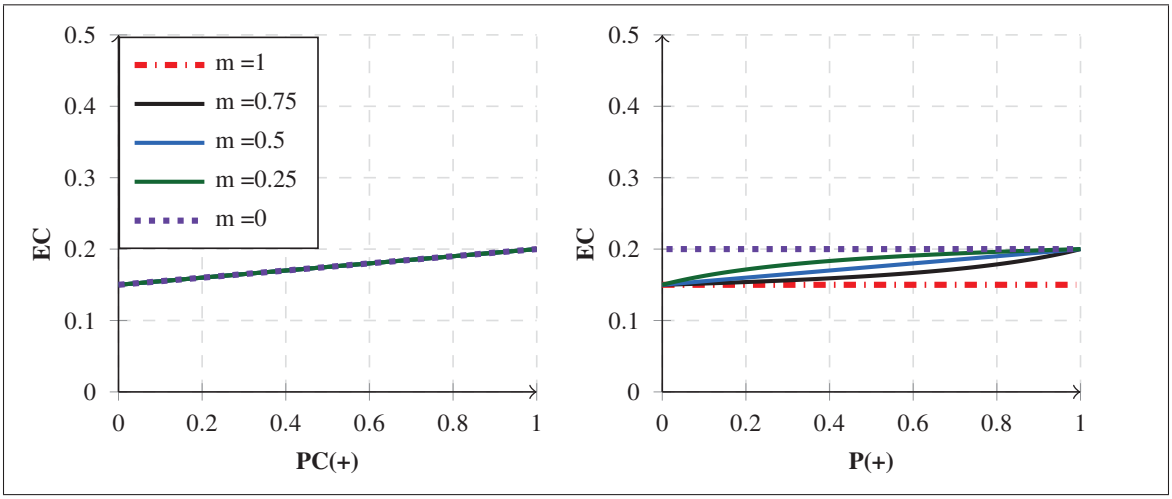


Figure 5.5 Global expected cost of a given classifier with TPR $= 0.8$ and FPR $= 0.15$, for different values of m against PC$(+)$ and $P(+)$. Note that for all values of $P(+)$: (1) for $m = 0$, $EC = 1 - TPR$, (2) for $m = 1$, $EC = FPR$.

The cost curves of two classifiers (C_i and C_j) may cross and one classifier may outperform the other for a certain range of operating points, i.e. either before or after the intersection point found as:

$$PC^{*}_{\mathrm{i,j}}(+) = \frac{\mathrm{FPR}_i - \mathrm{FPR}_j}{(\mathrm{TPR}_i - \mathrm{TPR}_j) + \mathrm{FPR}_i - \mathrm{FPR}_j}, \tag{5.6}$$

or,

$$P'_{\mathrm{i,j}}(+) = \frac{\mathrm{FPR}_i - \mathrm{FPR}_j}{(\mathrm{TPR}_i - \mathrm{TPR}_j) + (1/(m-1))(\mathrm{FPR}_i - \mathrm{FPR}_j).} \tag{5.7}$$

The relative performance of a pair of classifiers may differ in the F-measure and cost space and the best classifier for the same $P(+)$ can be different if one uses the EC or the F-measure because the crossing point of the curves in the F-measure and cost spaces can differ. Two examples are shown in Figure 5.6, where the F-measure and EC is plotted vs. $P(+)$ and

$PC(+)$ for (TPR, FPR) pairs to present cases when a crisp classifier dominates another one in CC space whereas their F-measure curves cross, or vice versa. In this example α and m are set to 0.5 (note that when $m = 0.5$, $PC(+) = P(+)$).

The difference between behaviors of the F-measure and cost curves is due to the difference between their sensitivity to the difference between FPR values of classifiers when data is imbalanced ($P(+) < 0.5$). The partial derivatives of EC (Eq. (1.18)) with respect to FPR is $1 - PC(+)$ for any values of TPR and FPR and therefore it only depends on $P(+)$ and m. However the partial derivative of the F-measure (Eq. (5.2)) with respect to FPR is $\frac{1-1/p}{TPR} F_\alpha^2$ and is a function of TPR, FPR, α and P(+). Similarly, the partial derivatives of EC with respect to TPR is $-PC(+)$ whereas the partial derivative of the F-measure with respect to TPR is $\frac{\alpha}{TPR} F_\alpha^2 + \frac{1}{TPR} F_\alpha$.

An example is shown in Figure 5.7 that demonstrate the behavior of curves in the F-measure and cost spaces when comparing different classifiers with different FPR values and the same TPR value. In Figure 5.7(a), the F-measure and 1-EC are plotted against $0 <$ FPR < 0.5 for a fixed TPR $= 0.75$, $P(+) = 0.25$, and $\alpha = m = 0.50$. It is observed that the F-measure exhibits a larger decrease than $1 - EC$ as FPR increases for the same value of TPR. Figure 5.7(b) shows the F-measure and cost curves of classifiers for a fixed TPR $= 0.75$, $0 <$ FPR < 0.5, $\alpha = m = 0.50$ and a range of $0.0001 < P(+) < 0.50$. Comparing the plots show that the difference between F-measure curves of classifiers with the same TPR and different FPR values is more significant than their cost curves, especially for higher imbalance level of data (smaller $P(+)$). For example, the cost curves of classifiers with $FPR = 0.01$ and 0.00 (shown as dashed and dotted curves) almost overlap whereas their F-measure curves exhibit visibly more different behavior as the $P(+)$ decreases. Very small changes in FPR when data is highly imbalanced means large changes in the number of false positives (or false alarms). Therefore, the F-measure space can be better than the cost space when the correct classification of positive class is attained at the expense of an excessive number of misclassified negative samples.

Note that the goal of CC (and of the F-measure space) is to evaluate classifier performance in terms of the skew level at *deployment* time (which is shown as P(+)). However, one can not access the exact value of P(+) and these curves are plotted as an estimate from Eq. 1.17 and eq:F-measure-alpha-new given a single TPR, FPR point. Therefore, the accuracy of these curves depend on the accuracy of the TPR, FPR. In practice the values of TPR and FPR are obtained by testing the classifier on test data and therefore with different test sets, different curves may be obtained for the same classifier.

5.2 Using F-measure Space to Tune Classification Systems

ROC curves can be used to set the parameters of the system such as the optimal decision threshold or to select the best classifier to gain the best performance for a particular operating condition. For that, ROCCH of the classifier(s) is found and the optimal classifier (or the threshold of the classifier) is selected by intersecting the iso-performance lines (extensively analysed by Drummond & Holte (2006)) corresponding to the given operating condition with the ROCCH at the most upper left side of the curve. If two classifiers (vertices) belong to the desired iso-performance line, there are two optimal classifiers (or threshold values). This process is easier in cost and F-measure spaces because the operating condition is shown on the x axis and one can easily find the optimal decision threshold of a soft classifier, or select the best classifier among a group or find the best combination of a set.

Two classifiers (vertices) in ROC space that belong to the desired iso-performance correspond to two lines in cost space that intersect in the given operating condition. Similarly, in F-space, two classifiers (vertices) in ROC space correspond to two curves. The cost and F-measure spaces are therefore useful to adapt the classification systems for the given conditions during operations. Three problems can be addressed using this space: (1) tuning the decision threshold of a single classifier; (2) choosing a given classifier between different available ones (each one with a predefined decision threshold); (3) choosing the best combination of a (subset of) available classifiers (each one with a predefined decision threshold).

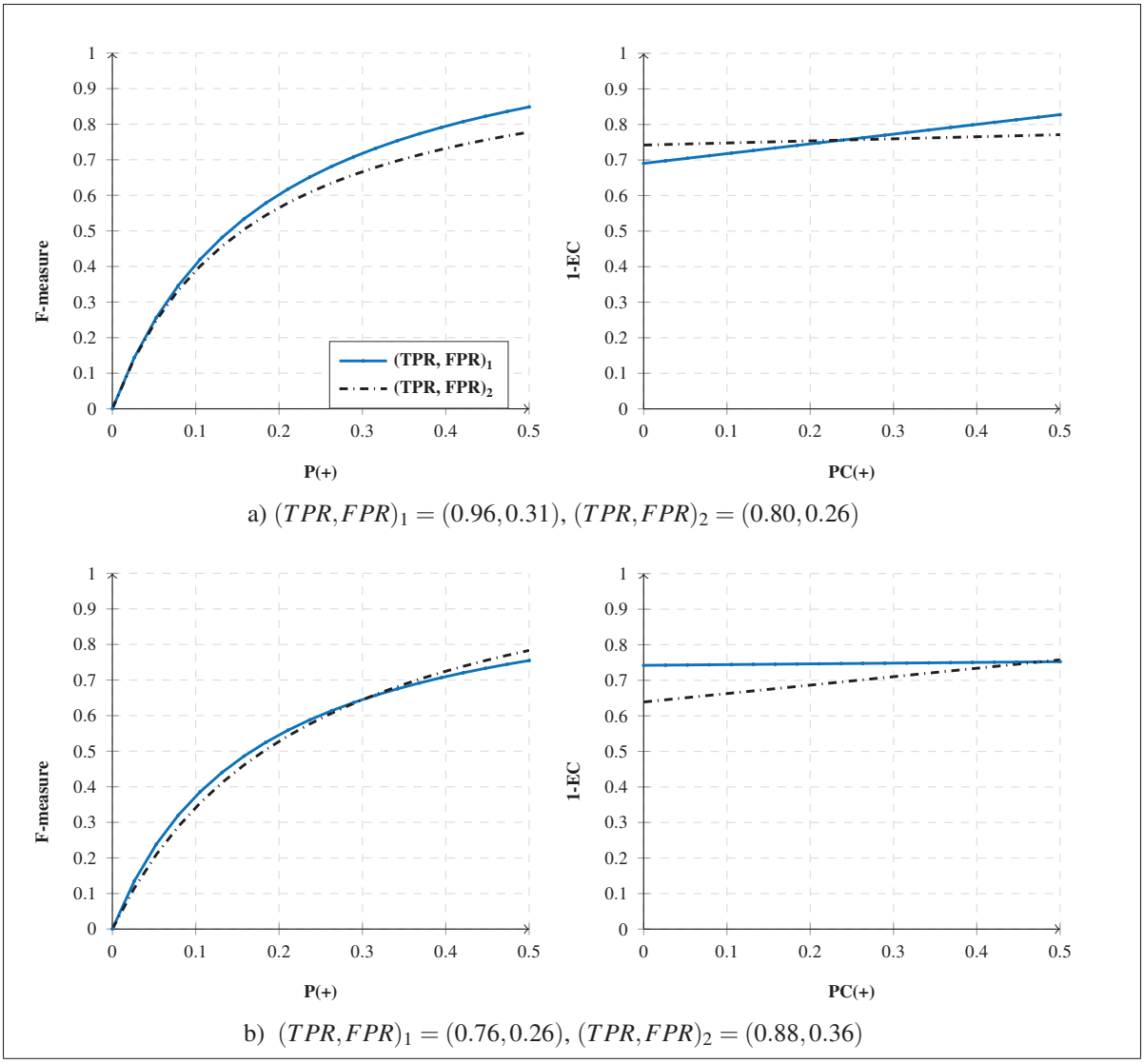
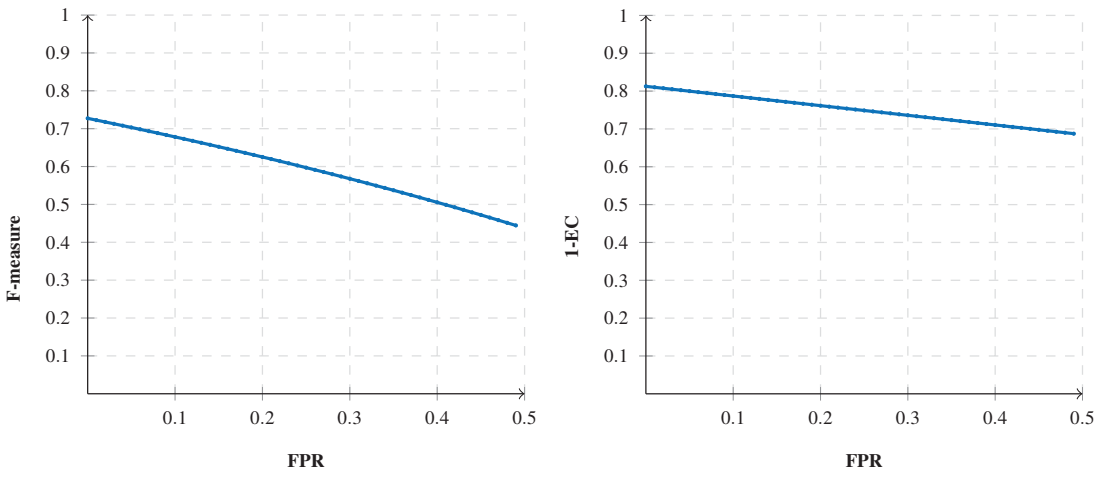


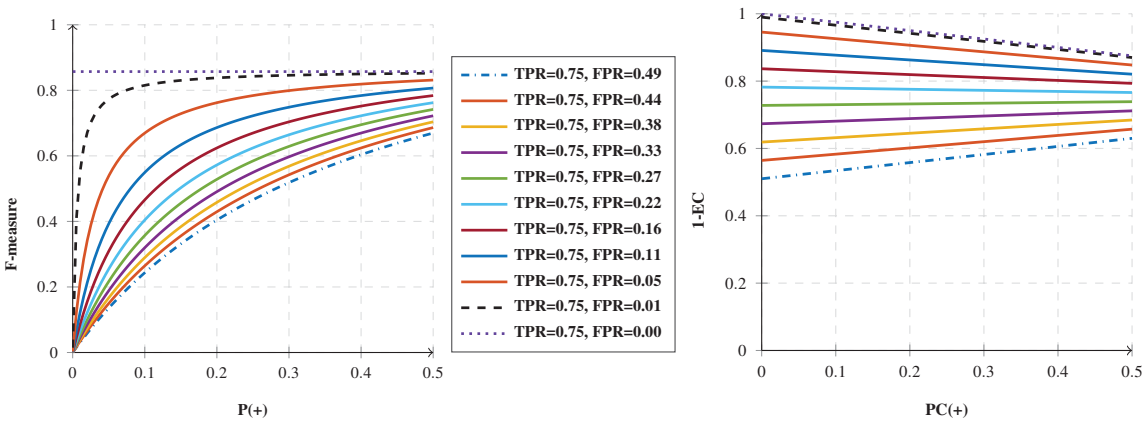
Figure 5.6 F-measure and cost curves of pairs of classifiers with $\alpha = m = 0.5$ (in this case $PC(+) = P(+)$).

5.2.1 Setting an Optimal Decision Threshold or Selecting the Best Classifier

Each decision threshold of a soft classifier corresponds to a crisp classifier and therefore setting an optimal decision threshold and selecting the best classifier among a group are carried out the same way. ROC curves are ideally suited for setting the optimal decision threshold of a classifier based on Neyman-Pearson criterion. In this case, a decision threshold is optimal if it corresponds to the maximum TPR for the given acceptable FPR. In cost space, a point can



a) F-measure and $1-EC$, $P(+)=0.25$



b) F-measure and $1-EC$ curves for all $P(+)$

Figure 5.7 TPR = 0.75 and different FPRs ($\alpha = m = 0.5$)

be placed on the y-axis representing the criterion (a vertical line crossing the acceptable FPR) and the classifier corresponding the cost line that intersects the y-axis under this point and also participates in forming the lower envelope is the best (Drummond & Holte, 2006). The lower bound of expected cost with FPR_{max} and TPR $= 1$ has $\text{EC}^{min} = (1-PC(+))\text{FPR}$. Analogously, in the proposed F-measure space the upper bound of F-measure value with FPR_{max} and TPR $=$ 1 has $F_\alpha^{max} = \frac{1}{1+\alpha(1/P(+)-1)\text{FPR}_{max}}$. Therefore, the best classifier is found as the one that has F_α closer to F_α^{max} and dominates the others because if $\text{FPR}_1 = \text{FPR}_2$ and $F_\alpha^1 < F_\alpha^2$, then for any α and $P(+)$, $\text{TPR}_1 < \text{TPR}_2$.

Another criterion to select the best classifier is the operating condition which is the skew level and the preference of classes. In ROC space the best classifier is found at the intersection of ROC convex hull with iso performance lines that correspond to the given operating condition in most up left part of ROC space. In both cost and F-measure spaces, the best classifier can be found more easily than ROC space because the skew level ($P(+)$) is shown on the x-axis. The preference between classes (α) is considered in plotting the curves corresponding to classifiers in F-measure space. In cost space the preference between classes (m) is considered along with $P(+)$ in $PC(+)$ on the x-axis.

5.2.2 Choosing the Best Combination of Available Classifiers Using Iterative Boolean Combination

For imbalanced data classification, the best single classifier or a combination of a subset of classifiers from a pool may be selected for each operating condition during deployment (Khreich *et al.*, 2010; Radtke *et al.*, 2014; De-la Torre *et al.*, 2015c). For this purpose, the classifiers in the pool are tested using validation sets with different imbalance levels after training to find the best (single or an ensemble of) classifier for the specific imbalance level of the validation data. During test, the imbalance level of the test data is estimated (using Hausdorff distance Edgar (2007)) and the corresponding best (single or an ensemble of) classifier is used for classification of test data. However, note that the goal of CC (and of the F-measure space) is to evaluate classifier performance in terms of the skew level at *deployment* time. So, using these tools the performance (at deployment time) for different imbalance levels can be estimated from the testing set performance estimated for a single imbalance level (that corresponds to a single TPR, FPR point) and from Eq. 1.17 and eq:F-measure-alpha-new), the same as in training data.

Several ways exist to combine classifiers in either score or decision levels including score averaging, majority voting, learning a meta-classifier on either decisions or score, etc. An interesting combination method is Iterative Boolean Combination (IBC) of classifiers Khreich *et al.* (2010), which involves selection and combination at the same time.

In this chapter, the IBC algorithm is optimized using the F-measure space to select the best combination of classifiers across a range of operating conditions (P(+)). Before describing the proposed approach in section 5.4 based on IBC, the Boolean combination of classifiers is described.

5.3 Boolean Combination of Classifiers

Diverse classifiers may be combined using the Boolean functions to achieve a more accurate and robust classification system. The Boolean combination of classifiers in ROC space have been investigated in literature for both crisp and soft classifiers (Black & Craig, 2002; Haker *et al.*, 2005; Fawcett, 2004; Barreno *et al.*, 2008; Tao & Veldhuis, 2009). The output of pairwise Boolean conjunction (AND) and disjunction (OR) of crisp classifiers may differ when they are conditionally independent or dependent. However, direct combination of responses from soft classifiers (probability estimates) considers the joint conditional probabilities of each classifier at each threshold. Therefore, no assumptions regarding the independence of the classifiers is required (Khreich *et al.*, 2010).

In this section the behaviour of curves of classifiers in ROC, EC, and F-measure spaces and their corresponding curves for pairwise combination of classifiers with Boolean functions are analysed.

5.3.1 Independent Classifiers

Given two conditionally independent classifiers C_i and C_j with $(\mathrm{TPR}_i, \mathrm{FPR}_i)$, and $(\mathrm{TPR}_j, \mathrm{FPR}_j)$, the TPR and FPR of $C_\wedge = C_i \wedge C_j$ (AND) and $C_\vee = C_i \vee C_j$ (OR) are obtained from:

$$\mathrm{TPR}_\wedge = \mathrm{TPR}_i \cdot \mathrm{TPR}_j, \tag{5.8}$$

$$\mathrm{FPR}_\wedge = \mathrm{FPR}_i \cdot \mathrm{FPR}_j, \tag{5.9}$$

$$\mathrm{TPR}_\vee = \mathrm{TPR}_i + \mathrm{TPR}_j - \mathrm{TPR}_i \cdot \mathrm{TPR}_j, \tag{5.10}$$

$$\mathrm{FPR}_\vee = \mathrm{FPR}_i + \mathrm{FPR}_j - \mathrm{FPR}_i \cdot \mathrm{FPR}_j. \tag{5.11}$$

From these equations, $\text{TPR}_\wedge < \text{TPR}_i$, $\text{TPR}_\wedge < \text{TPR}_j$, $\text{FPR}_\wedge < \text{FPR}_i$, and $\text{FPR}_\wedge < \text{FPR}_j$. Therefore, $(\text{TPR}_\wedge, \text{FPR}_\wedge)$ is located in lower-left side of both $(\text{TPR}_i, \text{FPR}_i)$ and $(\text{TPR}_j, \text{FPR}_j)$ in ROC space. For Boolean disjunction, $\text{TPR}_\vee > \text{TPR}_i$, $\text{TPR}_\vee > \text{TPR}_j$, and $\text{FPR}_\vee > \text{FPR}_i$, $\text{FPR}_\vee > \text{FPR}_j$. Therefore, $(\text{TPR}_\vee, \text{FPR}_\vee)$ is located in upper-right side of both $(\text{TPR}_i, \text{FPR}_i)$ and $(\text{TPR}_j, \text{FPR}_j)$ in ROC space.

In cost space, the expected cost of $C_\wedge$ and $C_\vee$ could be higher or lower than the expected cost of both C_i and C_j based on the value of $PC(+)$.

$$\begin{cases} EC_\wedge < EC_i & \text{if } PC(+) < \frac{\text{FPR}_i - \text{FPR}_\wedge}{(\frac{1-m}{m})(\text{TPR}_\wedge - \text{TPR}_i) + \text{FPR}_i - \text{FPR}_\wedge}. \\ EC_\wedge \geq EC_i & \text{otherwise.} \end{cases} \tag{5.12}$$

Similarly:

$$\begin{cases} EC_\vee < EC_i & \text{if } PC(+) > \frac{\text{FPR}_\vee - \text{FPR}_i}{(\frac{1-m}{m})(\text{TPR}_i - \text{TPR}_\vee) + \text{FPR}_\vee - \text{FPR}_i}. \\ EC_\vee \geq EC_\vee & \text{otherwise.} \end{cases} \tag{5.13}$$

In F-space, based on the conditions explained in section 5.1.2, $F_\alpha^\wedge > F_\alpha^i$, $F_\alpha^\wedge > F_\alpha^j$, $F_\alpha^\vee > F_\alpha^i$, and $F_\alpha^\vee > F_\alpha^j$ is not true for all values of $P(+) > 0$, and the corresponding F-measure curves cross in a single point. Therefore,

$$P^*_{\wedge,\text{i}}(+) = \frac{\text{FPR}_\wedge \cdot \text{TPR}_i - \text{FPR}_i \cdot \text{TPR}_\wedge}{(1 - {}^1\!/\!_\alpha)(\text{TPR}_i - \text{TPR}_\wedge) + \text{FPR}_\wedge \cdot \text{TPR}_i - \text{FPR}_i \cdot \text{TPR}_\wedge},$$

$$\begin{cases} F_\alpha^\wedge < F_\alpha^i & \text{if } P(+) < P^*_{\wedge,\text{i}}(+), \\ F_\alpha^\wedge \geq F_\alpha^i & \text{if } P(+) \geq P^*_{\wedge,\text{i}}(+), \end{cases} \tag{5.14}$$

$$P^*_{\vee,\mathrm{i}}(+) = \frac{\mathrm{FPR}_i \cdot \mathrm{TPR}_\vee - \mathrm{FPR}_\vee \cdot \mathrm{TPR}_i}{(1 - {}^1\!/_\alpha)(\mathrm{TPR}_\vee - \mathrm{TPR}_i) + \mathrm{FPR}_\vee \cdot \mathrm{TPR}_i - \mathrm{FPR}_i \cdot \mathrm{TPR}_\vee},$$

$$\begin{cases} F^\vee_\alpha < F^i_\alpha & \text{if } P(+) > P^*_{\vee,\mathrm{i}}(+), \\ F^\vee_\alpha \geq F^i_\alpha & \text{if } P(+) \leq P^*_{\vee,\mathrm{i}}(+). \end{cases} \tag{5.15}$$

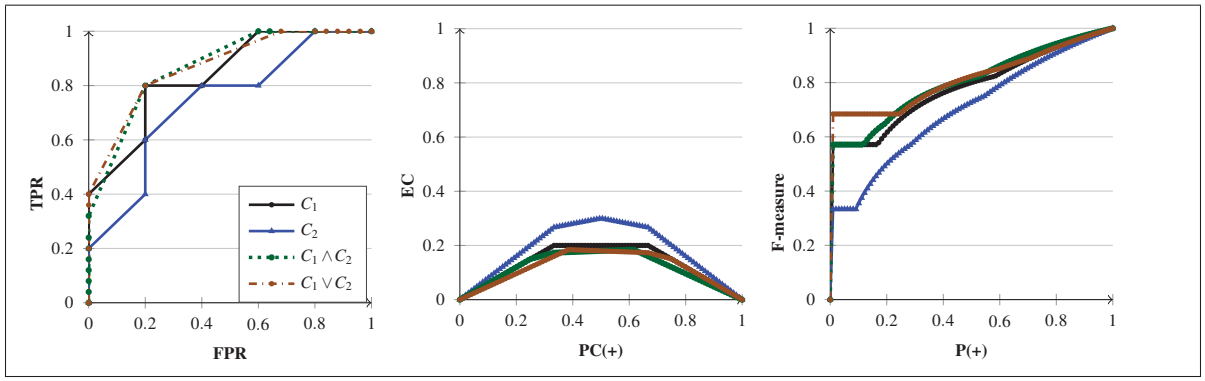


Figure 5.8 Indirect combination of independent (TPR, FPR) points.

5.3.2 Dependent Classifiers

In most real-world problems, the classifiers are dependent and the probability that both detectors classify positive samples correctly (positive correlation between classifiers) may take any value between $\mathrm{TPR}_i \cdot \mathrm{TPR}_j$, and $\min(\mathrm{TPR}_i, \mathrm{TPR}_j)$ (Black & Craig, 2002). Similarly the probability that both classifiers classify negative samples correctly (negative correlation between classifiers) takes a value between $(1 - \mathrm{FPR}_i)(1 - \mathrm{FPR}_j)$ and $\min((1 - \mathrm{FPR}_i), (1 - \mathrm{FPR}_j))$. Therefore,

$$\mathrm{TPR}_i \cdot \mathrm{TPR}_j \quad < \quad \mathrm{TPR}_\wedge < \min(\mathrm{TPR}_i, \mathrm{TPR}_j), \tag{5.16}$$

$$\mathrm{FPR}_i \cdot \mathrm{FPR}_j \;<\; \mathrm{FPR}_\wedge < \mathrm{FPR}_i + \mathrm{FPR}_j - 1 + \min(\mathrm{FPR}_i, \mathrm{FPR}_j),$$

$$\mathrm{TPR}_\vee \;>\; \mathrm{TPR}_i + \mathrm{TPR}_j - \min(\mathrm{TPR}_i, \mathrm{TPR}_j), \tag{5.17}$$

$$\mathrm{TPR}_\vee \;<\; \mathrm{TPR}_i + \mathrm{TPR}_j - \mathrm{TPR}_i \cdot \mathrm{TPR}_j, \tag{5.18}$$

$$\mathrm{FPR}_\vee \;>\; 1 - \min(1 - \mathrm{FPR}_i, 1 - \mathrm{FPR}_j), \tag{5.19}$$

$$\mathrm{FPR}_\vee \;<\; \mathrm{FPR}_i + \mathrm{FPR}_j - \mathrm{FPR}_i \cdot \mathrm{FPR}_j. \tag{5.20}$$

In this case $\mathrm{TPR}_\wedge < \mathrm{TPR}_i$ and $\mathrm{TPR}_\wedge < \mathrm{TPR}_j$. However, it is not easy to determine the relative position of resulting classifiers from AND and OR functions to the original classifiers in ROC, cost or F-measure spaces. Therefore, it is not possible to find an exact value neither for $(\mathrm{TPR}_\wedge, \mathrm{FPR}_\wedge)$ nor for $(\mathrm{TPR}_\vee, \mathrm{FPR}_\vee)$.

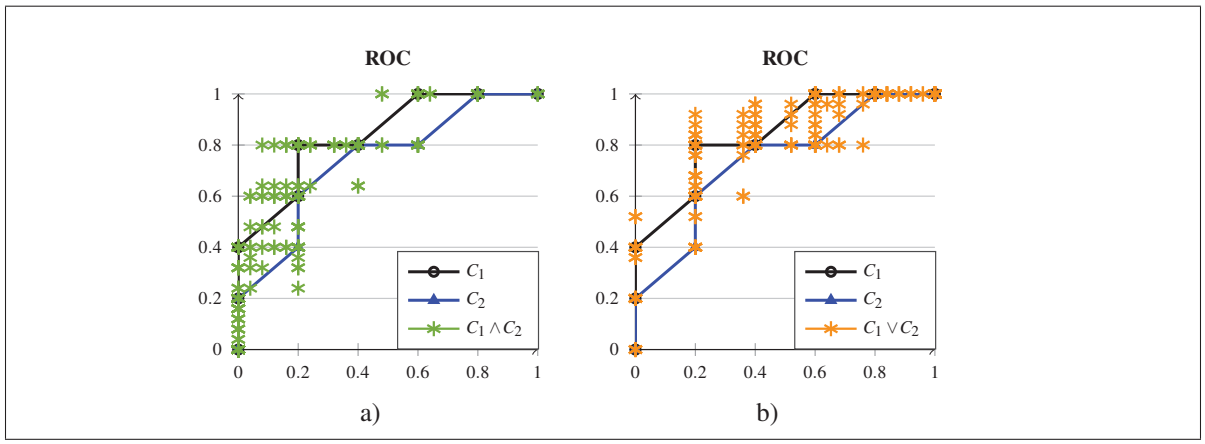


Figure 5.9 Resulting range of possible values from combination of two dependent classifiers using AND and OR functions in ROC space.

5.3.3 Direct Combination of Decisions

Exploiting direct combination of decisions from pairs of classifiers as done in Iterative Boolean Combination (IBC) (Khreich *et al.*, 2010) is more straightforward and implicitly accounts for dependence between classifiers. The direct combination of decisions from pairs of classifiers is carried out as follows.

The decision thresholds of each classifier are selected by sorting the scores of the classifier in ascending order. Let's consider Th_i^t $(i = 1, ..., N_i)$ as the thresholds of C_i and Th_j^t, $(j = 1, ..., N_j)$ as the thresholds of C_j. First two vectors d_i^k and d_j^k $(k = 1, ..., N_i, ..., N_i \times N_j)$ are found for each pair of (Th_i^k, Th_j^k) such that the elements of d_i^k and d_j^k are set to 1 when the decisions of C_i and C_j are correct, and to 0 when the decisions are incorrect. Then, d_i^k and d_j^k are directly combined using the Boolean functions to D_{ij}. Based on the true labels of samples, the confusion matrix is obtained from D_{ij} and (TPR_{ij}, FPR_{ij}) of the resulting classifier is calculated.

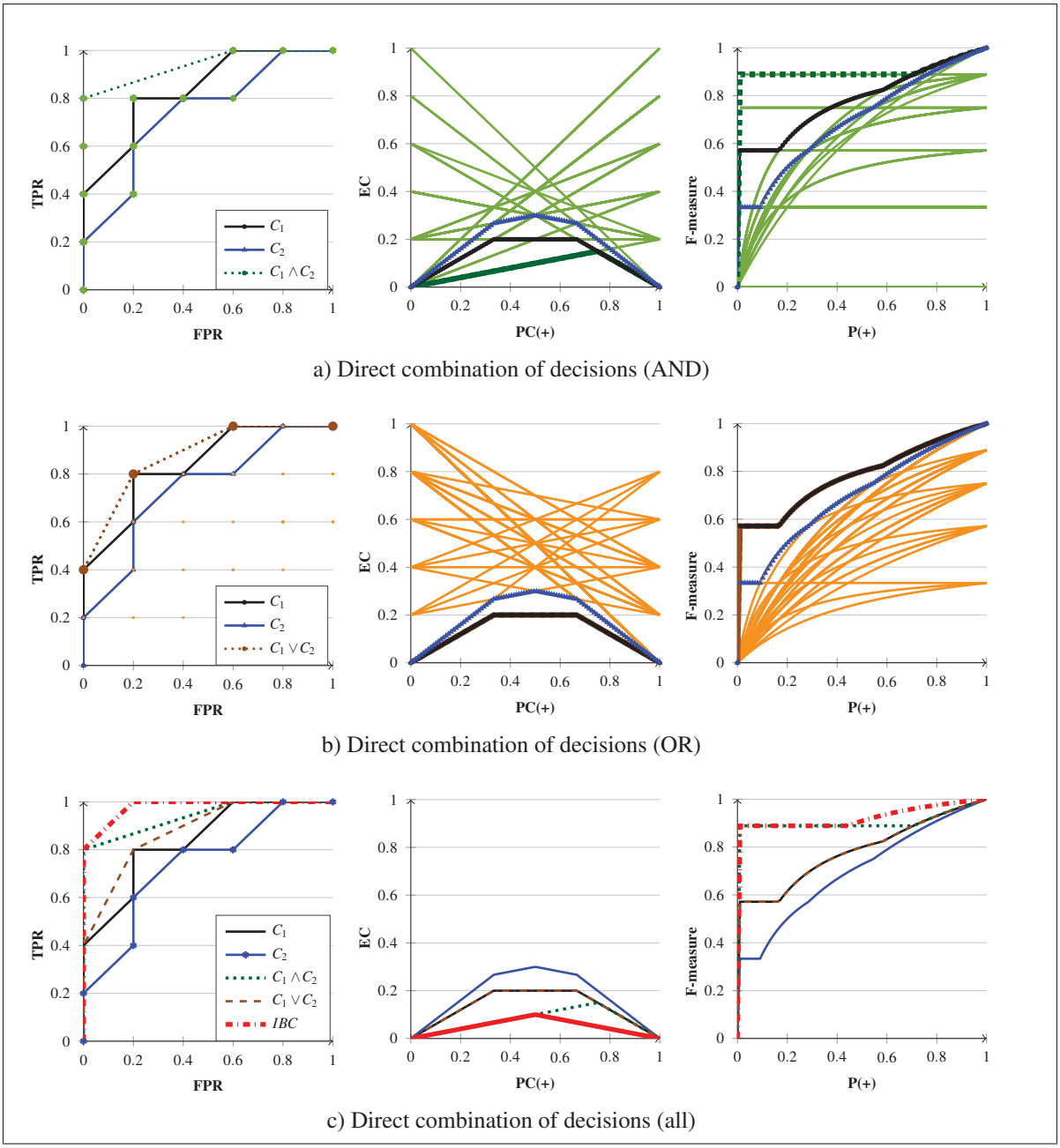
With the indirect combination method, first (TPR_i, FPR_i) and (TPR_j, FPR_j) of the classifiers are found from d_i^k and d_j^k. Then, the (TPR_{ij}, FPR_{ij}) of the combination is obtained from equations (5.8), (5.9), (5.10) or (5.11) based on the dependency or independency of the classifiers.

5.3.4 Comparing Combination Methods

Given two sets of scores, the Boolean combination of two soft classifiers in Figures 5.9 and A5.11 is found in three ways: (1) assuming dependent classifiers and using the values of TPR and FPR (Figure 5.9), (2) assuming independent classifiers and using the values of TPR and FPR (Figure 5.11), (3) direct combination of decisions given the scores (Figures 5.10).

It is observed that the three methods may have different combination results. With the first two methods, AND function improves the performance for lower values of P(+) better than OR function. However, with the third method we can only identify a range for the ROC curves of AND and OR combination results. Showing this range is more complicated in cost and F-measure spaces.

In Figure 5.10, "IBC' corresponds to combination of classifiers using 10 Boolean functions $(C_i \wedge C_j, \neg C_i \wedge C_j, C_i \wedge \neg C_j, \neg(C_i \wedge C_j), C_i \vee C_j, \neg C_i \vee C_j, C_i \vee \neg C_j, \neg(C_i \vee C_j), C_i \bigoplus C_j, C_i \equiv C_j)$ (Khreich *et al.*, 2010). Comparing results of AND, OR, and IBC in Figure 5.10 shows that using all Boolean functions to combine decisions of two classifiers directly results in a more accurate and robust classification system.



a) Direct combination of decisions (AND)

b) Direct combination of decisions (OR)

c) Direct combination of decisions (all)

Figure 5.10 Direct combination of decisions.

5.4 Iterative Boolean Combination of Classifiers in F-measure Space

Iterative Boolean Combination (IBC) of classifiers (Khreich *et al.*, 2010), which involves selection and combination at the same time, starts by combining two classifiers and keeps the combination if it improves ROCCH over the ROC curves of the combined classifiers. In the

next iteration, the resulting classifier from the previous iteration is combined with the third classifier and this process continues until all the classifiers are combined. The combination method used in this algorithm is the direct Boolean combination of decisions from pairs of classifiers.

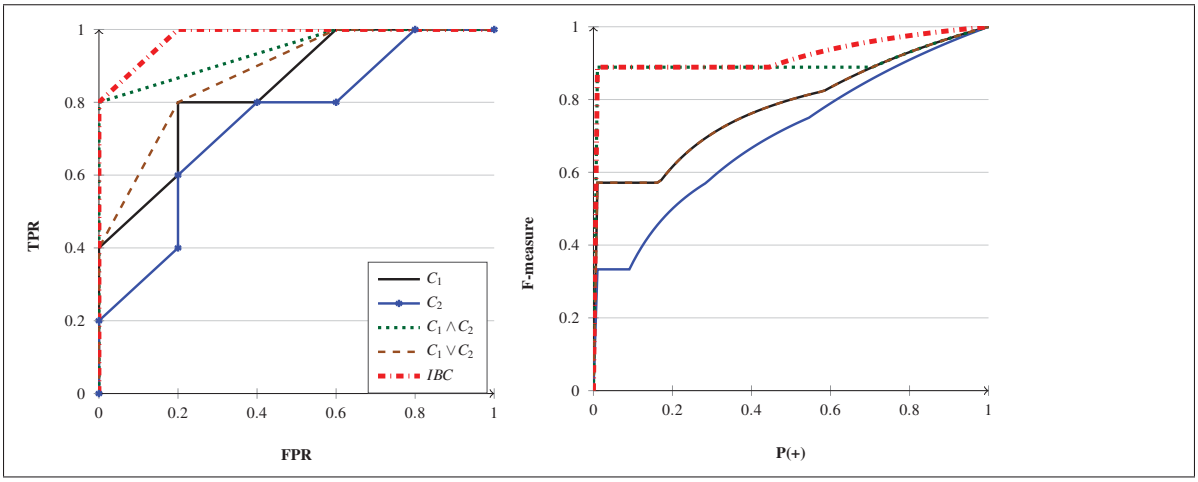


Figure 5.11 Boolean combination of decisions in ROC and F-measure spaces.

In this section the IBC algorithm is modified in the proposed F-measure space by replacing the AUC condition by the value of the F-measure (Alg. 8). A pool of classifiers are trained on a dataset with $P_{train}(+)$ and then tested on the validation set with $P_{validation}(+)$. Then all the decision vectors of each soft classifier are found by varying its decision threshold (lines 1-3 of Algo. 8). After that every single decision vector is combined with the other using each of the Boolean functions $B_1 : a \wedge b, B_2 : \neg a \wedge b, B_3 : a \wedge \neg b, B_4 : \neg(a \wedge b), B_5 : a \vee b, B_6 : \neg a \vee b, B_7 : a \vee \neg b, B_8 : \neg(a \vee b), B_9 : a \oplus b, B_{10} : a \equiv b$. This process results in $N_D = T_1 + T_2 ... + T_e + 10 \times E \times (E-1) \times T_1 \times T_2 ... \times T_e$ decision thresholds that also includes the original decision vectors of the classifiers in the pool.

In section 5.1.1, it was explained that a soft classifier in the F-measure space can be shown as the upper envelope of all the curves that correspond to all the available pairs of (TPR, FPR) values of that soft classifier, and the upper envelope shows the best performance of the classifier with the most suitable tuning of decision threshold for each operating condition. Therefore,

one can justify that finding the upper envelope of all the F-measure curves of the available pairs of (TPR, FPR) values obtained from a pool of classifiers and different combinations of them results in finding the best collection of classifiers across a range of $P(+)$ based on the F-measure. This idea is used in lines 13-16 of the proposed algorithm (Alg. 8) to collect the best classifiers or a combination of them for each operating condition.

The worst-case time complexity of combining a pair of classifiers with this algorithm is $\mathcal{O}(T_e T_k)$ for each of 10 Boolean operations, given classifiers c_e and c_k have respectively T_e and T_k thresholds Khreich *et al.* (2010). Therefore, for combining E classifiers in design stage of the proposed Boolean combination of classifiers in the F-measure space, the worst-case time complexity of Boolean combination algorithm is $\mathcal{O}(T_{\max}^2 E(E-1))$ Boolean operations. Where $T_{\max}$ corresponds to the maximum number of thresholds among E classifiers. During deployment, given a specific operating condition, the worst-case time complexity is $\mathcal{O}(T_D)$ (see Alg. 8).

An example is shown in Figure 5.11 for combining two soft classifiers in both ROC space and the proposed F-measure space. The AND function alone improves over the performance of C_1 and C_2 for higher imbalance levels (lower $P(+)$) and using Alg. 8 improves the performance over the whole range of operating conditions.

5.5 Experiments and Results

The experiments in this chapter consist of two parts carried out on the FIA and COX video datasets. In the first part (Section 5.2.1), three classifiers are trained and tested and their performance is visualized in ROC, PR, EC, and F-measure spaces. Then, given an operating condition (imbalance level and preference between classes), the F-measure and cost spaces are used for (1) selecting a single best classifier among them, (2) setting an optimal decision threshold of each classifier. In the second part (Section 5.2.2), the Bagging ensemble learning (Barandela *et al.*, 2003) method (with RBF-SVM base classifiers) for imbalance is adapted to given operating conditions using F-measure space by selecting and combining a subset of classifiers using a modified version of the IBC technique proposed in this chapter (see Algorithm 8).

Algorithm 8: Choosing the Best Combination of Classifiers in F-measure Space

Input: Soft classifiers: c_e , $e = 1,...,E$
Validation Set: $\mathbf{V} = \{(\mathbf{x}_i, y_i); i = 1,...,M\}$, $y_i \in \{0,1\}$
Preference between classes: α
Operating points: $P(+) = \{p_j, j = 1,...,N_{op}\}$
Boolean functions: $B_1 : a \wedge b, B_2 : \neg a \wedge b, B_3 : a \wedge \neg b, B_4 : \neg(a \wedge b), B_5 : a \vee b, B_6 : \neg a \vee b, B_7 : a \vee \neg b, B_8 : \neg(a \vee b), B_9 : a \oplus b, B_{10} : a \equiv b$

Output: Combined set of classifiers: BC

for $e = 1,..,E$ **do**
  Test c_e on $\mathbf{V}$ and get back scores set $\mathbf{S}_e$.
  Define T_e thresholds as the unique values in $\mathbf{S}_e$ and get back decisions $d_i^e (i = 1,...,T_e)$
Define D_{bc} and store all d_i^es.
for $l = 1,...,10$ **do**
  for $e = 1,...,E$ **do**
    for $k = 1,...,E (k \neq e)$ **do**
      for $t = 1,...,T_e$ **do**
        for $n = 1,...,T_k$ **do**
          Combine d_t^e and d_n^k using B_l and add the resulting decision vector to D_{bc}.
Find the F-measure curve of D_{bc} with the size $N_D = T_1 + T_2 ... + T_e + 10 \times E \times (E-1) \times T_1 \times T_2 ... \times T_e$ (as the upper envelope of F-measure curves corresponding to all decision vectors in D_{bc}) as:
for $i = 1,..,N_{op}$ **do**
  for $j = 1,..,N_D$ **do**
    Find TPR and FPR values from the j^{th} decision vector in D_{bc} and the true labels of the validation data using confusion matrix.
    Calculate $f_j = \frac{\alpha^{-1} TPR}{\alpha^{-1} + p_j^{-1} FPR + TPR - FPR - 1}$
  $F_i = \max_{j=1,...,N_D} f_j$
  Store the corresponding (l, e, k, t, n) for the i^{th} operating condition and call it BC_i to be used during testing and deployment.

5.5.1 Experiments for Comparing Classifiers in F-measure Space

In this experiment, C_1: Naive Bayes, C_2: MLP (one hidden layer, 8 hidden units) and C_3: RBF-SVM (LibSVM Chang & Lin (2011)) are designed and tested when $P_{\text{train}}(+) = P_{\text{test}}(+) = 0.01$. This experiment is an example of:

1. How the performance of different classifiers can be compared for each considered scalar performance measure, using the corresponding (different) global measures/visualization tools.

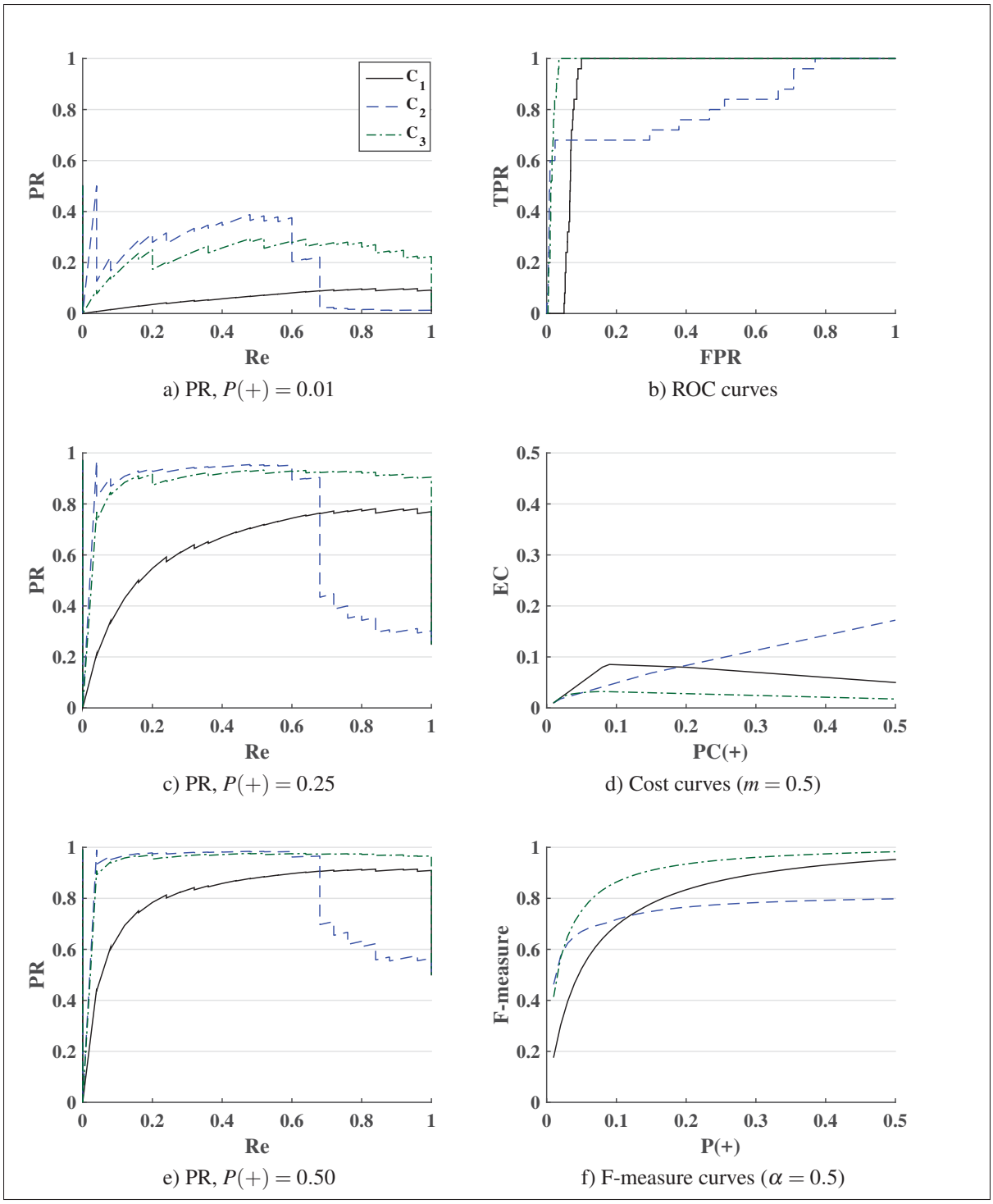
2. How the effect of class imbalance can be observed using these global measures/visualization tools.

For this experiment, one video is used for training and one video is used for testing. One individual is randomly selected as the target class and 99 individuals are selected randomly as the non-targets. The ROIs in the trajectory of the target individual are considered as the positive class and the ROIs in the trajectories of the non-target individuals are considered as the negative class. Therefore, $P_{\text{train}}(+) = P_{\text{test}}(+) = 0.01$. There are 25 samples in each trajectory.

Figures 5.12 and 5.13 shows the performance of the above mentioned classifiers respectively on COX and FIA datasets in Precision-Recall (when $P(+) = 0.01$, 0.25 and 0.50), ROC, EC, and F-measure spaces. In Cost and F-measure spaces both m and α are set to 0.5. Note that in this case $PC(+) = P(+)$. From ROC, Cost and F-measure spaces in Figure 5.12, it is observed that C_3 outperforms the others. In cost and F-measure spaces, it is easy to compare these classifiers for any given $P(+)$. In the cost space, C_2 outperforms C_1 when $PC(+) < 0.2$ whereas in the F-measure space C_2 outperforms C_1 when $P(+) < 0.11$. When $P(+) < 0.03$, the difference between the performance of C_2 and C_3 is not easy to detect in both the cost and the F-measure spaces. It is difficult to make a justification about the relative performance of these classifiers in the precision-recall space. However, in Figure 5.13, it is observed from all visualization spaces that C_3 dominates C_1 and C_2 over all operating conditions.

As a reminder from sections 1.2.3.3 and 5.1.1, note that different decision thresholds of a soft classifier correspond to different lines and curves in cost and F-measure spaces, respectively. The optimal threshold value for each operating condition corresponds to the lower and upper envelopes in the cost and F-measure spaces, respectively.

In Figures 5.14 and 5.15, both cost and F-measure spaces are used to select an optimal decision threshold for C_3 given different values of $P(+)$ on the COX and FIA datasets, respectively. Each row corresponds to one value of $P(+)$. In both Cost and F-measure spaces in Figures 5.14 and 5.15, the curves that correspond to each threshold (shown as Th) are plotted with grey color that appear as shaded areas in the figures of first and second columns. The final lower envelope



a) PR, $P(+) = 0.01$

b) ROC curves

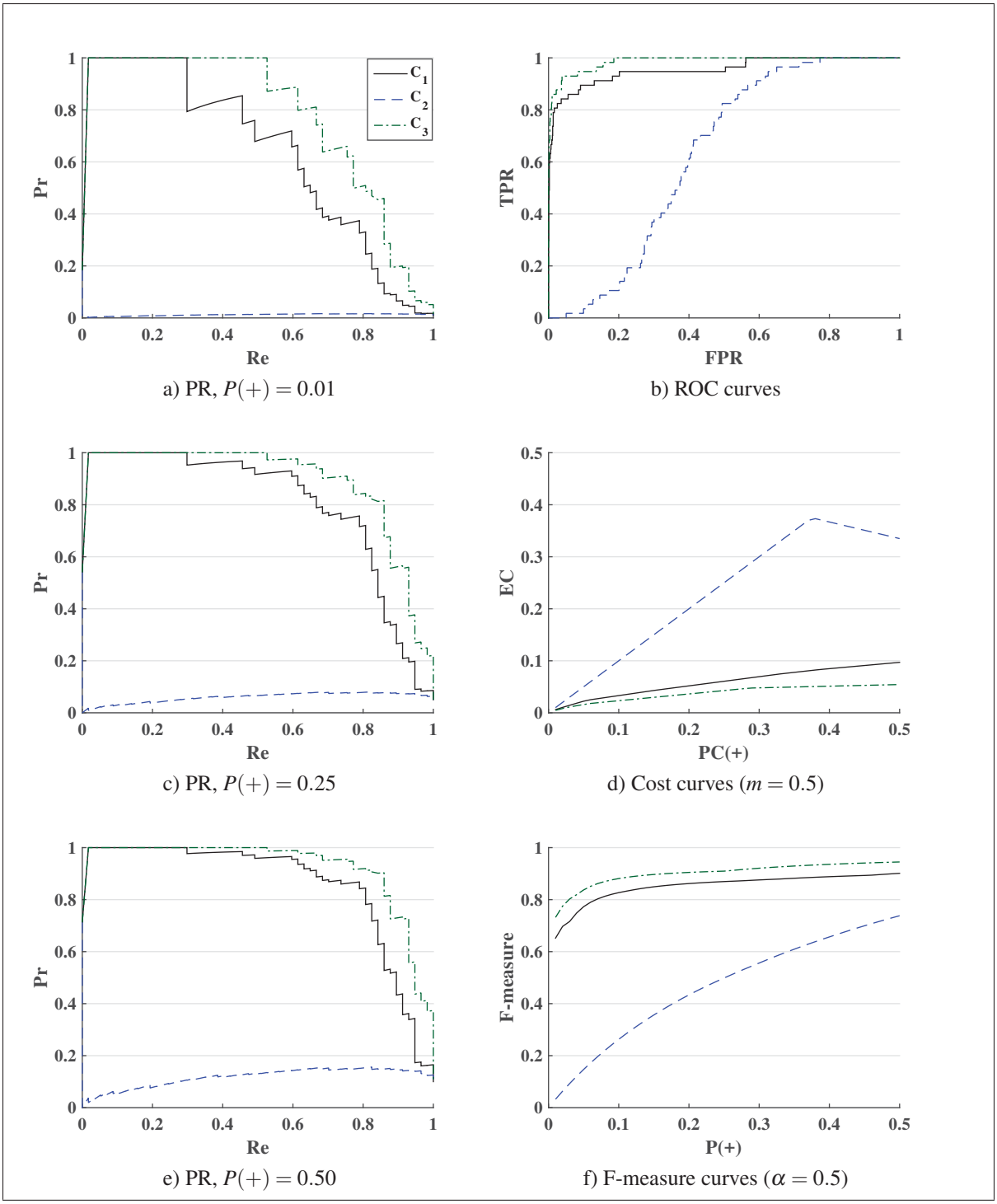
c) PR, $P(+) = 0.25$

d) Cost curves ($m = 0.5$)

e) PR, $P(+) = 0.50$

f) F-measure curves ($\alpha = 0.5$)

Figure 5.12 Comparing C_1, C_2 and C_3 for different values of $P(+)$ on COX dataset.



a) PR, $P(+) = 0.01$ b) ROC curves

c) PR, $P(+) = 0.25$ d) Cost curves ($m = 0.5$)

e) PR, $P(+) = 0.50$ f) F-measure curves ($\alpha = 0.5$)

Figure 5.13 Comparing C_1, C_2 and C_3 for different values of $P(+)$ on FIA dataset.

curve in the Cost space and the upper envelope curve in the F-measure space is also shown in these figures.

In this experiment, the unique values of the classification scores are used as the decision thresholds

If the given $P(+)$ is not considered during finding the lower/upper envelope of the cost and F-measure plots, no threshold value (or (TPR, FPR) pair) may be found that corresponds to the lower/upper envelope for that $P(+)$. In Figures 5.14 and 5.15, the cost and F-measure curves are plotted with $P(+) = 0.01, 0.02, 0.03, ..., 0.5$ and validated to find the optimal decision for $P(+) = 0.015, 0.035, 0.065, 0.285$ are not among those considered during plotting these curves.

When there is a threshold value (or (TPR, FPR) pair) that corresponds to the lower/upper envelope for a given $P(+)$ (Figure 5.14(a), (g), (d), (h), and Figure 5.15 (d) (e) (g) (h) (k)), that optimal point is returned and the corresponding cost line or F-measure curve is shown. However, in Figure 5.14(e), (b), (f), (c) and Figure 5.15(a), (b), (j) there is no threshold value (or (TPR, FPR) pair) that corresponds to the lower/upper envelope for the given $P(+)$. In these cases, the interpolation of the adjacent threshold values is returned as the optimal threshold (the cost lines and F-measure curves corresponding to all three points are shown in the figures).

In the third column of Figure 5.14, the optimal threshold values (or (TPR, FPR) pairs) obtained from cost and F-measure curves are then shown and compared in ROC space. In Figure 5.14, it is observed that for any $P(+) < 0.03$, the best (TPR, FPR) pair found using the Cost space is close to $(0,0)$ in the ROC space, which is not optimal since it corresponds to a classifier that does not classify any positive class sample correctly. For example in the first row, $P(+)$ is set to 0.015 which corresponds to the optimal point of $(TPR, FPR) = (0.82, 0.022)$ from the F-measure space. When $P(+)$ is set to any value between 0.03 and 0.09 (e.g. Figure 5.14(b), (f) and (j)), the optimal point of (TPR, FPR) obtained from the cost curve is different than the one obtained from the F-measure space. When $P(+)$ is set to any value higher than 0.09, the optimal thresholds obtained from Cost and F-measure spaces are identical. The problem of not being able to find an optimal threshold from cost space is not encountered in the results with the FIA dataset (Figure 5.15).

5.5.2 Experiments for Iterative Boolean Combination of Classifiers in F-measure Space

It was explained in sections 5.1.1 and 5.2.2 that finding the upper envelope of F-measure curves that correspond to the available (TPR, FPR) points results in finding the best collection of classifiers across a range of possible P(+) during deployment. We used this idea to modify the IBC algorithm in 5.2.2. In the experiments of this section, this algorithm is put to test to see if this method can result in a better performance in terms of F-measure when the classifiers that are stored for the test time are those that have been selected and combined during validation using the proposed IBC algorithm in the F-measure space.

In this experiment, a pool of 20 classifiers is trained using Bagging algorithm (Barandela *et al.*, 2003) that randomly samples balanced subsets of data to train each classifier in the pool. Then, the classifiers in the pool are tested with the validation set and a subset of the classifiers is selected and combined using Algorithm 8. At test time, the combination that was selected from the validation step is used.

Both FIA and COX datasets are used for the experiments. As explained in section 2.1.2, for this experiment, face captures from one individual (that is randomly selected as target) are considered as the positive class and the face captures from a number of randomly selected individuals are considered as the negative class. In order to consider two cases where $P_{\text{train}}(+) = P_{\text{validation}}(+) = P_{\text{test}}(+) = 0.1$ and $P_{\text{train}}(+) = P_{\text{validation}}(+) = P_{\text{test}}(+) = 0.04$, 9 and 24 negative class individuals are selected randomly, respectively.

The F-measure curves resulting from these experiments are averaged over the overall $10 \times 10 = 100$ overall rounds of experiments and the results are shown in Figure 5.16.

In Figure 5.16 , the F-measure curve of resulting combined classifier (shown as adaptive IBC) is compared to the original Bagging ensemble (shown as adaptive Bag) that combines all classifiers in the pool by score averaging. It is observed that the proposed IBC picks and combines the classifiers better and improves the performance in terms of the F-measure, over a range of imbalance levels. In Figure 5.16 (a), that range is $P(+) < 0.25$ when $P_{\text{train}}(+) = P_{\text{validation}}(+) =$

$P_{\text{test}}(+) = 0.1$, and it is $P(+) < 0.4$ when $P_{\text{train}}(+) = P_{\text{validation}}(+) = P_{\text{test}}(+) = 0.04$. In Figure 5.16 (b), that range is $0.04 > P(+)$ when $P_{\text{train}}(+) = P_{\text{validation}}(+) = P_{\text{test}}(+) = 0.1$, and it is all $P(+)$s when $P_{\text{train}}(+) = P_{\text{validation}}(+) = P_{\text{test}}(+) = 0.04$. This could be due to the fact that when the classifiers are trained, validated and tested on higher imbalance levels they become more efficient to handle higher imbalance levels during deployment time (as observed in previous chapters of this thesis).

5.6 Discussion

Here we summarize the discussion related to the results of analysis and experiments throughout the paper that support the the main contribution of the paper. We proposed a new global performance evaluation space that simply allow one to directly evaluate performance in terms of the scalar metric; the F-measure with the following properties:

- Possibility of visualizing the performance of any classifier (soft or crisp) under different imbalance levels of test data. In section 5.1.1 (Figures 5.1 and 5.2), we showed how a crisp classifier corresponds to a single curve in the F-measure space and a soft classifier corresponds to the upper envelope of several curves (each of which corresponds to a single threshold). We also showed that one gets different curves for a single classifier for different preference level between classes (different values of α).

- Possibility of selecting the best threshold of a classifier under the given imbalance level and preference between precision and recall. This idea was presented in section 5.2.1. In section 5.5.1 (Figure 5.14), we carried out an experiment to select the best optimal decision threshold of SVM classifier, for the given $P(+)$ values.

- Since each classifier corresponds to a curve in the proposed F-measure space, it becomes possible to compare more than two classifiers over different decision thresholds and under different imbalance levels of test data and preference between classes (see section 5.1.2). This also provides us the possibility of selecting the best classifier among others for the given imbalance level ($P(+)$) and preference between precision and recall (α). In sec-

tion 5.5.1 (Figure 8), we carried out an experiment to compare the performance of three classifiers with a real-world video dataset.

- This space can also be used to select the best combination of a set of classifiers. As the second contribution of the paper in section 5.2.2, the proposed F-measure space is used to modify the Iterative Boolean Combination (IBC) method to adapt the selection and combination of classifiers in the ensemble for an optimal performance under different operating conditions (imbalance levels). In section 5.5.2 (Figure 5.16), an experiment is carried out on the video dataset to select the best combination of classifiers generated using the Bagging algorithm. The result of the experiment shows a significant improvement of performance in the F-measure space.

- The proposed F-measure space is preferred to the ROC and Precision-Recall spaces. ROC space is not focused on a specific performance measure, and is also not sensitive to class imbalance, which makes it unsuitable to classification problems with skewed class distributions. The PR space is analogous to ROC space when precision and recall are of interest, instead of TPR and FPR. Although this space is sensitive to imbalance, it does not allow to easily visualize how the F-measure behaves as a function of class skew. In Figures 5.2 and 5.12 experiments are used to show the advantage of the F-measure space to the ROC and Precision-Recall spaces.

- The F-measure space can be preferable to cost space in some applications when precision-recall is preferred to the misclassification cost like in information retrieval. In addition, the F-measure space can be preferable to cost space in some scenarios of imbalanced data classification when no specific performance measure can be defined with regard to the preference between expected cost and precision-recall. The reason is that the F-measure space is more sensitive to class imbalance and tuning the preference between classes results in a visible difference in performance in the F-measure space compared to the cost space. In section 5.1.3, we analyzed and compared the F-measure and cost spaces. We saw that tuning α results in a visible difference in performance while tuning m does not provide the same effect in the conventional cost space that depicts EC against $PC(+)$. In addition in

Figure 5.7 and the related text, the sensitivity of the F-measure to differences in TPR and FPR between two or more classifiers is more significant than the sensitivity of the *EC* to those. In the results of the experiments in Figure 5.14, we also observed that for high level of imbalance the F-measure space is preferred to the cost space for selecting the suitable decision threshold.

5.7 Conclusions

In this paper, the main existing global evaluation measures and visualization tools were overviewed, and a new one was proposed specifically for the scalar F-measure and for class imbalance problems. The scalar F-measure, weighs the ability of a classifier in recognizing the positive class (the minority and the class of interest) versus the misclassification rate of the negative class (the majority class). It is a suitable scalar performance measure to compare classifiers under imbalance and no visualization tool exists to depict it globally for different operational conditions. Therefore, the F-measure space is proposed as a versatile tool to visualize and compare classifiers performance under different operating conditions (i.e. skew level of data and preference between recall and precision). This space can be used to select the best decision threshold for a soft classifier as well as the best soft classifier among a group, for the given operating condition. This space can also be used to select the best classifiers based on Neyman-Pearson criterion. This space can be further used to modify learning algorithms to address imbalance. In this paper, this space is used to modify the Iterative Boolean Combination algorithm. The experiments on a real-world video dataset is carried out in order to show the use of this space to compare and select classifiers as well as the improvement of performance using the modified Iterative Boolean Combination algorithm for the Bagging ensemble learning method.

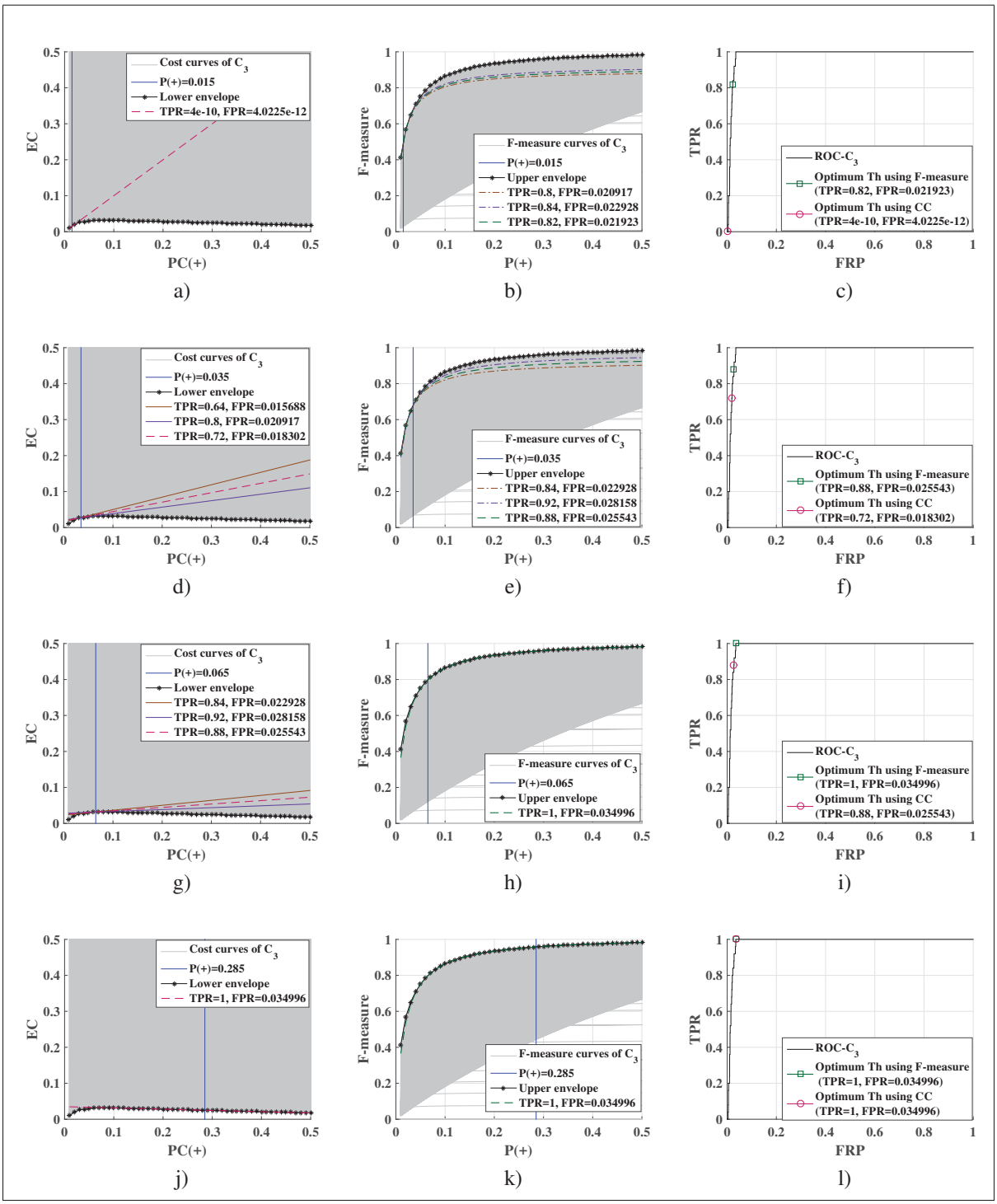


Figure 5.14 Results of experiments on COX dataset to find the optimal decision threshold (and the corresponding TPR, FPR values) in cost and F-measure spaces given a specific operating condition ($P(+)$). First column shows the cost curves, the second column shows the F-measure curves and the third column shows the ROC curves.

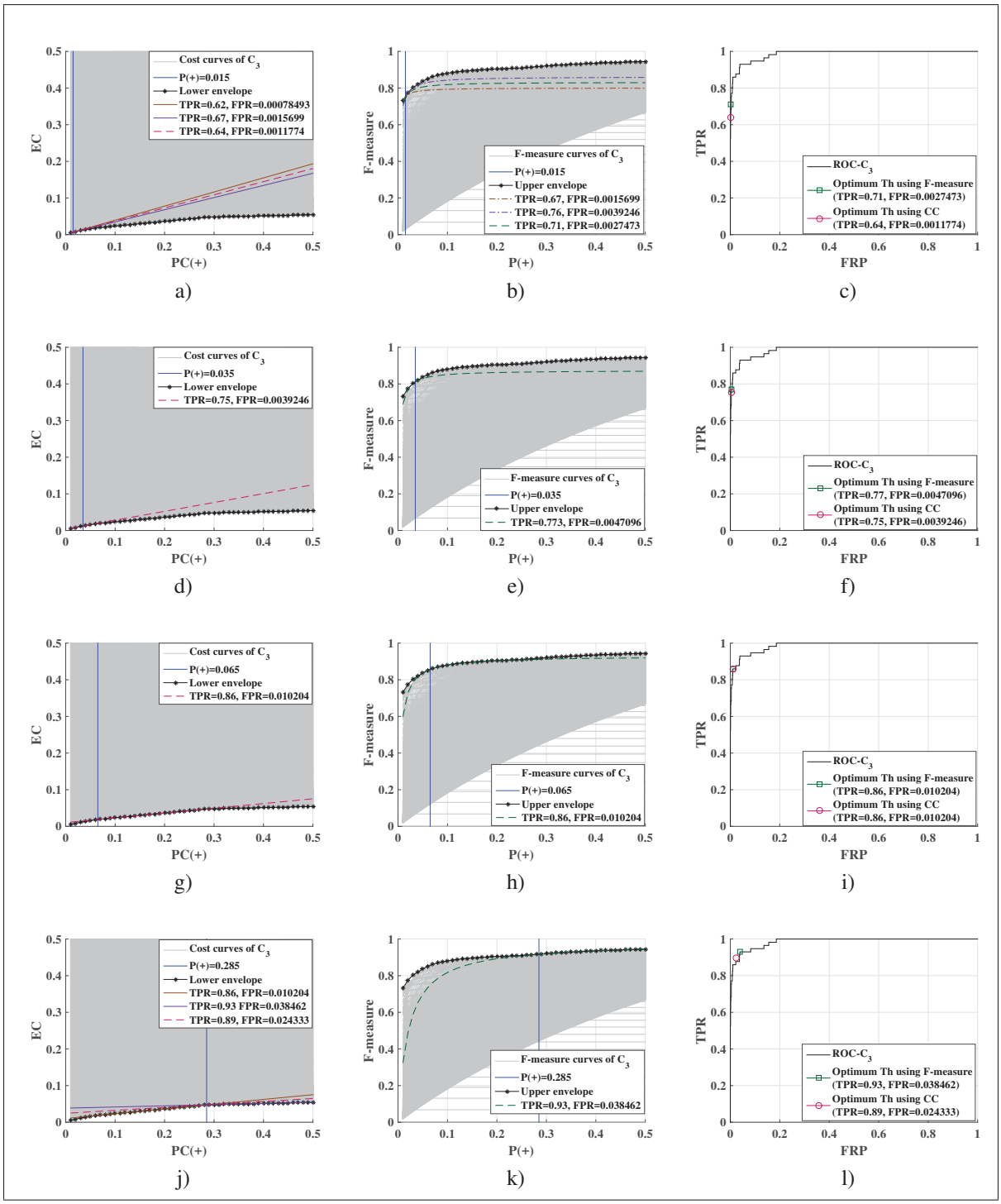


Figure 5.15 Results of experiments on FIA dataset to find the optimal decision threshold (and the corresponding TPR, FPR values) in cost and F-measure spaces given a specific operating condition ($P(+)$). First column shows the cost curves, the second column shows the F-measure curves and the third column shows the ROC curves.

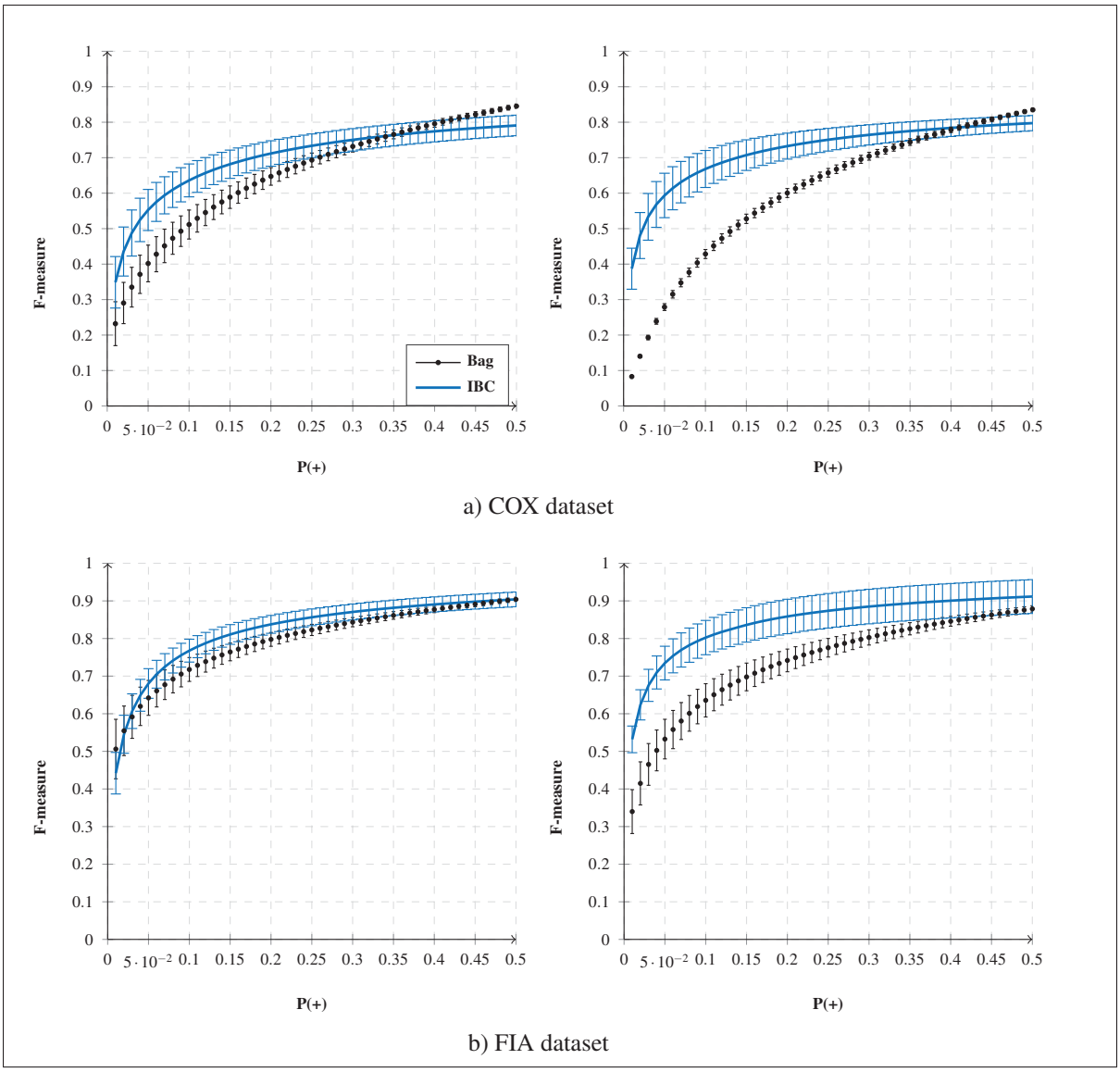


Figure 5.16 F-measure curves of Bagging ensemble method and its modified version using IBC in the F-measure space on COX dataset (first row) and FIA dataset (second row).
First column: $P_{\text{train}}(+) = P_{\text{validation}}(+) = P_{\text{test}}(+) = 0.1$.
Second column: $P_{\text{train}}(+) = P_{\text{validation}}(+) = P_{\text{test}}(+) = 0.04$

CONCLUSION AND RECOMMENDATIONS

Face re-identification, an application of face recognition in video surveillance, is challenging due to variations in capture condition and imbalance. Imbalance is a fundamental problem in many machine learning applications since most of classification algorithms are designed with assumption of balanced data distribution and therefore, when these classification systems are designed using imbalanced data their performance becomes biased towards classification of the majority class. This bias of performance is also due in part to using unsuitable performance metrics in optimization of most classification algorithms. In addition, most of the classification algorithms in the literature that are specialized for imbalanced data consider a fixed imbalance level known a priori. In this thesis three ensemble methods are proposed for robust learning and classification of imbalanced data in face re-identification application. In these ensembles the property of varying imbalance of data is taken into account and a suitable performance measure is used to optimize their learning algorithms. In addition, a versatile performance measurement space is proposed that provide the possibility of comparing classification systems under different operating conditions.

In Chapter 1 a review of literature about the systems for face recognition in video, and specialized classification systems for imbalance is presented. It was found that Bagging and Boosting ensembles of classifiers have been modified to learn from imbalanced data and can be further improved in this thesis by accounting for varying imbalance level of data during operations. In addition, this review showed us that F-measure is a suitable performance metric to evaluate classifiers under imbalance. However, there exists no global performance evaluation tool to compare classifiers in terms of F-measure under different operating conditions.

In Chapter 2 an experimental methodology is presented that involves comparing specialized classification systems using the suitable performance metrics for this kind of problem in different operating conditions using both synthetic and video datasets. A synthetic data set is

generated such that the level of imbalance as well as the overlap between classes is controllable.

In Chapter 3 an under-sampling method (Trajectory Under-Sampling (TUS)) is proposed for Bagging ensemble that is specialized for face re-identification application. The proposed under-sampling method maintains a good level of diversity-accuracy and is used to design Bagging ensembles that use data subsets with different imbalance levels to learn base classifiers of the ensemble with two strategies. One strategy relies on random selection of non-target trajectories and the other relies on sorting non-target trajectories based on their distance from the target class. In the first strategy the F-measure is used to weigh the contribution of classifiers in final prediction of the ensemble. The proposed ensemble method is then further extended by under-sampling trajectories to support vectors using SVM classifiers to improve performance and efficiency. The experiments on both synthetic and video datasets showed that the proposed under-sampling method is more suitable than general-purpose sampling methods for this application and ensembles designed using this sampling strategy outperform ensembles from literature in this application.

In Chapter 4 a new Boosting ensemble (Progressive Boosting (PBoost)) is proposed that accounts for varying imbalance level of data during operations in face re-identification application. In the proposed ensemble the RUSBoost ensemble is improved in three ways. First, the proposed under-sampling method (TUS) replaces the random under sampling to avoid information loss and increase diversity. Second, the loss function of Boosting algorithm is modified using the F-measure to avoid bias of performance towards the majority class. Third, the classifiers in the ensemble are validated on different imbalance levels to improve robustness to varying imbalance level of data during operations. The experiments on both synthetic and video datasets showed that the proposed PBoost algorithm outperform the specialized Boosting ensembles for imbalance in terms of accuracy, robustness, and computation complexity.

In Chapter 5 a global performance evaluation tool is proposed to compare classifiers under different operating conditions in terms of F-measure. F-measure is sensitive to imbalance and includes a controlling parameter to set the preference between classes. In this space each classifier is presented as a curve that shows its performance under different imbalance levels with different decision thresholds for the given preference between classes. The proposed space is then used to adapt specialized ensemble methods from literature to varying imbalance levels to select and combine classifiers at the same time using Iterative Boolean Combination (IBC). The experiments on video datasets showed that the proposed space provides the possibility of comparing classification systems under varying imbalance for the given preference between classes. In addition, experiments showed that the proposed modified IBC improves the robustness of classification systems to imbalance significantly.

Future work:

- The proposed under-sampling method can be applicable in many image classification applications where there exists a natural sub-clasters of data in a one-versus-all classification strategy.

- The proposed TUS ensembles and PBooost can be used in any application to learn from imbalanced data.

- Although the proposed systems in this thesis demonstrated robust and efficient performance for face re-identification application under imbalance, more strategies can be developed. An important open issue is to account for changes in capture condition in videos such as variations in illumination, pose, expression, etc in conjunction with imbalance problem. For example, the training samples can be assigned with weights based on their quality in terms of the capture conditions and these weights can be integrated into the proposed PBoost algorithm.

- As mentioned in section 5.1.3, the accurateness of the F-measure and cost curves of a classifier depend on the quality of the test data used to obtain them and TPR and FPR estimation errors. Therefore, in practice, with different validation sets, different curves may be obtained for the same classifier. Tackling this problem analogous to Brier curves (which take into account the use of suboptimal decision thresholds estimated from validation data) is an open issue for future work.

- There exists other classification algorithms that have not been adapted to imbalance in this thesis. For example, the trending interest in deep neural networks opens the opportunity to design end-to-end recognition systems robust to imbalance for application in real-world problems. As reviewed in section 1.2.2 designing such systems is still a young research area because conventional neural network architectures are not suitable for imbalance problems. There are three reasons for this problem. First, these networks are often trained on balanced datasets that do not reflect the real-world application scenario. Second, the greater gradient component of the class with higher number of samples result in the optimization of the parameters of the network biased towards this class. Third, these networks are optimized based on loss functions that are unsuitable for imbalanced data classification. In addition, the performance of these networks is often measured based on either accuracy or AUC which are known to be unsuitable for imbalanced data classification problems.

- Direct optimization of the learning procedure of the neural network based on the F-measure (as proposed in this thesis) is not applicable since the F-measure based loss function is not differentiable. Therefore, finding a way to use F-measure to optimize a deep CNN architecture for face video recognition under imbalance could be investigated in a future work.

- Using the proposed trajectory under-sampling method and progressively inserting the samples as the mini-batches that are used for training the neural network can be a possible

way to make the neural network architecture robust to imbalance. This algorithm can be investigated in a future work.

BIBLIOGRAPHY

Aghdam, H. H. & Heravi, E. J. (2017). Guide to Convolutional Neural Networks. *New York, NY: Springer. doi*, 10, 978–3.

Ahmed, E., Jones, M. & Marks, T. K. (2015). An improved deep learning architecture for person re-identification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3908–3916.

Ahonen, T., Hadid, A. & Pietikainen, M. (2006). Face description with local binary patterns: Application to face recognition. *IEEE transactions on pattern analysis and machine intelligence*, 28(12), 2037–2041.

Akbani, R., Kwek, S. & Japkowicz, N. (2004). Applying support vector machines to imbalanced datasets. *European conference on machine learning*, pp. 39–50.

Alcalá, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L. & Herrera, F. (2010). KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic and Soft Computing*, 17(2-3), 255–287.

Ambeth Kumar, V. D., Ramya, S., Divakar, H. & Kumutha Rajeswari, G. (2019). A Survey on Face Recognition in Video Surveillance. *Proceedings of the International Conference on ISMAC in Computational Vision and Bio-Engineering 2018 (ISMAC-CVB)*, pp. 699–708.

Anand, R., Mehrotra, K. G., Mohan, C. K. & Ranka, S. (1993). An improved algorithm for neural network classification of imbalanced training sets. *IEEE Transactions on Neural Networks*, 4(6), 962–969.

Artan, Y., Haider, M. A., Langer, D. L., van der Kwast, T. H., Evans, A. J., Yang, Y., Wernick, M. N., Trachtenberg, J. & Yetik, I. S. (2010). Prostate cancer localization with multispectral MRI using cost-sensitive support vector machines and conditional random fields. *IEEE Transactions on Image Processing*, 19(9), 2444–2455.

Barandela, R., Valdovinos, R. M. & Sánchez, J. S. (2003). New applications of ensembles of classifiers. *Pattern Analysis & Applications*, 6(3), 245–256.

Barreno, M., Cardenas, A. & Tygar, J. D. (2008). Optimal ROC curve for a combination of classifiers. *Advances in Neural Information Processing Systems*, pp. 57–64.

Barry, M. & Granger, E. (2007). Face recognition in video using a what-and-where fusion neural network. *Neural Networks, 2007. IJCNN 2007. International Joint Conference on*, pp. 2256–2261.

Bashbaghi, S., Granger, E., Sabourin, R. & Parchami, M. (2019). Deep Learning Architectures for Face Recognition in Video Surveillance. In Jiang, X., Hadid, A., Pang, Y., Granger, E. & Feng, X. (Eds.), *Deep Learning in Object Detection and Recognition* (pp. 133–154). Singapore: Springer Singapore. doi: 10.1007/978-981-10-5152-4_6.

Batuwita, R. & Palade, V. (2009). A new performance measure for class imbalance learning. application to bioinformatics problems. *Machine Learning and Applications, 2009. ICMLA'09. International Conference on*, pp. 545–550.

Bazzani, L., Cristani, M., Perina, A. & Murino, V. (2012). Multiple-shot person re-identification by chromatic and epitomic analyses. *Pattern Recognition Letters*, 33(7), 898–903.

Bedagkar-Gala, A. & Shah, S. K. (2014). A survey of approaches and trends in person re-identification. *Image and Vision Computing*, 32(4), 270–286.

Black, M. A. & Craig, B. A. (2002). Estimating disease prevalence in the absence of a gold standard. *Statistics in medicine*, 21(18), 2653–2669.

Błaszczyński, J., Deckert, M., Stefanowski, J. & Wilk, S. (2010). Integrating selective preprocessing of imbalanced data with Ivotes ensemble. *Rough Sets and Current Trends in Computing*, pp. 148–157.

Branco, P., Torgo, L. & Ribeiro, R. P. (2016). A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys (CSUR)*, 49(2), 31.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2), 123–140.

Breitenstein, M. D., Reichlin, F., Leibe, B., Koller-Meier, E. & Gool, L. V. (2009). Robust tracking-by-detection using a detector confidence particle filter. *2009 IEEE 12th International Conference on Computer Vision*, pp. 1515-1522.

Britto Jr, A. S., Sabourin, R. & Oliveira, L. E. (2014). Dynamic selection of classifiers A comprehensive review. *Pattern Recognition*, 47(11), 3665 - 3680.

Buda, M., Maki, A. & Mazurowski, M. A. (2017). A systematic study of the class imbalance problem in convolutional neural networks. *arXiv preprint arXiv:1710.05381*.

Buda, M., Maki, A. & Mazurowski, M. A. (2018). A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106, 249–259.

Chang, C.-C. & Lin, C.-J. (2011). LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3), 27.

Chawla, N. V., Bowyer, K. W., Hall, L. O. & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16(1), 321–357.

Chawla, N. V., Lazarevic, A., Hall, L. O. & Bowyer, K. W. (2003). SMOTEBoost: Improving prediction of the minority class in boosting. *European Conference on Principles of Data Mining and Knowledge Discovery*, pp. 107–119.

Chen, W. (2017). Deep Embedding for Face Recognition in Public Video Surveillance. *Biometric Recognition: 12th Chinese Conference, CCBR 2017, Shenzhen, China, October 28-29, 2017, Proceedings*, 10568, 31.

Chopra, S., Hadsell, R. & LeCun, Y. (2005, June). Learning a similarity metric discriminatively, with application to face verification. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 1, 539-546 vol. 1. doi: 10.1109/CVPR.2005.202.

Chopra, S., Hadsell, R. & LeCun, Y. (2005). Learning a similarity metric discriminatively, with application to face verification. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 1, 539–546.

Chung, Y.-A., Lin, H.-T. & Yang, S.-W. (2015). Cost-aware pre-training for multiclass cost-sensitive deep learning. *arXiv preprint arXiv:1511.09337*.

Cohen, G., Hilario, M., Sax, H., Hugonnet, S. & Geissbuhler, A. (2006). Learning from imbalanced data in surveillance of nosocomial infection. *Artificial Intelligence in Medicine*, 37(1), 7–18.

Connolly, J.-F., Granger, E. & Sabourin, R. (2012). An adaptive classification system for video-based face recognition. *Information Sciences*, 192, 50–70.

Dass, S. C., Nandakumar, K. & Jain, A. K. (2005). A Principled Approach to Score Level Fusion in Multimodal Biometric Systems. *Audio- and Video-Based Biometric Person Authentication*, pp. 1049–1058.

Davis, J. & Goadrich, M. (2006). The relationship between Precision-Recall and ROC curves. *Proceedings of the 23rd international conference on Machine learning*, pp. 233–240.

De-la Torre, M., Granger, E., Radtke, P., Sabourin, R. & Gorodnichy, D. O. (2014). Self-updating with facial trajectories for video-to-video face recognition. *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pp. 1669–1674.

De-la Torre, M., Granger, E., Radtke, P. V., Sabourin, R. & Gorodnichy, D. O. (2015a). Partially-supervised learning from facial trajectories for face recognition in video surveillance. *Information Fusion*, 24, 31–53.

De-la Torre, M., Granger, E. & Sabourin, R. (2015b). Adaptive Skew-Sensitive Fusion of Ensembles and their Application to Face Re-Identification. *Pattern Recognition*, 48, 3385-3406.

De-la Torre, M., Granger, E., Sabourin, R. & Gorodnichy, D. O. (2015c). Adaptive skew-sensitive ensembles for face recognition in video surveillance. *Pattern Recognition*, 48(11), 3385–3406.

Dembczynski, K. J., Waegeman, W., Cheng, W. & Hüllermeier, E. (2011). An exact algorithm for F-measure maximization. *Advances in neural information processing systems*, pp. 1404–1412.

Díez-Pastor, J. F., Rodríguez, J. J., García-Osorio, C. & Kuncheva, L. I. (2015). Random Balance: Ensembles of variable priors classifiers for imbalanced data. *Knowledge-Based Systems*, 85, 96–111.

Ding, C. & Tao, D. (2017). Pose-invariant face recognition with homography-based normalization. *Pattern Recognition*, 66, 144–152.

Ding, W., Huang, D.-Y., Chen, Z., Yu, X. & Lin, W. (2017). Facial action recognition using very deep networks for highly imbalanced class distribution. *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 1368–1372.

Drummond, C. & Holte, R. C. (2006). Cost curves: An improved method for visualizing classifier performance. *Machine learning*, 65(1), 95–130.

Dunn, J. C. (1973). A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters.

Edgar, G. (2007). *Measure, topology, and fractal geometry*. Springer Science & Business Media.

Fan, W., Stolfo, S. J., Zhang, J. & Chan, P. K. (1999). AdaCost: misclassification cost-sensitive boosting. *Machine Learning and Applications, 1999. ICMLA'99 International Conference on*.

Fawcett, T. (2004). ROC graphs: Notes and practical considerations for researchers. *Machine learning*, 31(1), 1–38.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern recognition letters*, 27(8), 861–874.

Ferri, C., Hernández-Orallo, J. & Modroiu, R. (2009). An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, 30(1), 27–38.

Ferri, C., Hernández-orallo, J. & Flach, P. A. (2011). Brier curves: a new cost-based visualisation of classifier performance. *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 585–592.

Flach, P. (2014). Classification in context, Adapting to changes in class and cost distribution. *LMCE-2014*.

Flach, P. & Kull, M. (2015). Precision-Recall-Gain Curves: PR Analysis Done Right. *Advances in Neural Information Processing Systems*, pp. 838–846.

Flach, P. A. (2003). The geometry of ROC space: understanding machine learning metrics through ROC isometrics. *Machine Learning and Applications, 2003. ICMLA'03. International Conference on*, pp. 194–201.

Freund, Y. (1995). Boosting a weak learning algorithm by majority. *Information and computation*, 121(2), 256–285.

Freund, Y. & Schapire, R. E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. *Computational learning theory*, pp. 23–37.

Freund, Y., Schapire, R. E. et al. (1996). Experiments with a new boosting algorithm. *Machine Learning and Applications, 1996. ICMLA'96 International Conference on*.

Galar, M., Fernández, A., Barrenechea, E., Bustince, H. & Herrera, F. (2011). An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. *Pattern Recognition*, 44(8), 1761–1776.

Galar, M., Fernandez, A., Barrenechea, E., Bustince, H. & Herrera, F. (2012). A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 42(4), 463–484.

Galar, M., Fernández, A., Barrenechea, E., Bustince, H. & Herrera, F. (2013a). Dynamic classifier selection for one-vs-one strategy: avoiding non-competent classifiers. *Pattern Recognition*, 46(12), 3412–3424.

Galar, M., Fernández, A., Barrenechea, E. & Herrera, F. (2013b). EUSBoost: Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling. *Pattern Recognition*, 46(12), 3460–3471.

García, S. & Herrera, F. (2009). Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy. *Evolutionary computation*, 17(3), 275–306.

Garcıa, V., Mollineda, R. & Sánchez, J. (2010). Theoretical analysis of a performance measure for imbalanced data. *Proceedings of the 20th International Conference on Pattern Recognition*, pp. 617–620.

Gheissari, N., Sebastian, T. B. & Hartley, R. (2006). Person reidentification using spatiotemporal appearance. *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, 2, 1528–1535.

Goh, R., Liu, L., Liu, X. & Chen, T. (2005). The CMU face in action (FIA) database. In *Analysis and Modelling of Faces and Gestures* (pp. 255–263).

Goswami, G., Bhardwaj, R., Singh, R. & Vatsa, M. (2014). MDLFace: Memorability augmented deep learning for video face recognition. *Biometrics (IJCB), 2014 IEEE International Joint Conference on*, pp. 1–7.

Guo, G. & Zhang, N. (2019). A survey on deep learning based face recognition. *Computer Vision and Image Understanding*, 189, 102805. doi: https://doi.org/10.1016/j.cviu.2019.102805.

Guo, H. & Viktor, H. L. (2004). Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *ACM SIGKDD Explorations Newsletter*, 6(1), 30–39.

Guo, Y. & Zhang, L. (2017). One-shot face recognition by promoting underrepresented classes. *arXiv preprint arXiv:1707.05574*.

Hadid, A. & Pietikäinen, M. (2009). Combining appearance and motion for face and gender recognition from videos. *Pattern Recognition*, 42(11), 2818–2827.

Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H. & Bing, G. (2016). Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications*.

Haker, S., Wells III, W. M., Warfield, S. K., Talos, I.-F., Bhagwat, J. G., Goldberg-Zimring, D., Mian, A., Ohno-Machado, L. & Zou, K. H. (2005). Combining classifiers using their receiver operating characteristics and maximum likelihood estimation. *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 506–514.

Hanczar, B. & Nadif, M. (2013). Precision-recall space to correct external indices for biclustering. *Proceedings of the 30th international conference on machine learning (ICML-13)*, pp. 136–144.

He, H. & Garcia, E. A. (2009). Learning from imbalanced data. *Knowledge and Data Engineering, IEEE Transactions on*, 21(9), 1263–1284.

Hernández-Orallo, J., Flach, P. & Ferri, C. (2012). A unified view of performance metrics: Translating threshold choice into expected classification loss. *Journal of Machine Learning Research*, 13(Oct), 2813–2869.

Ho, T. K. (1995). Random decision forests. *Proceedings of 3rd international conference on document analysis and recognition*, 1, 278–282.

Hochberg, Y. (1988). A sharper Bonferroni procedure for multiple tests of significance. *Biometrika*, 75(4), 800–802.

Hu, G., Yang, Y., Yi, D., Kittler, J., Christmas, W., Li, S. Z. & Hospedales, T. (2015). When face recognition meets with deep learning: an evaluation of convolutional neural networks for face recognition. *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 142–150.

Hu, J., Lu, J. & Tan, Y. (2014, June). Discriminative Deep Metric Learning for Face Verification in the Wild. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1875-1882. doi: 10.1109/CVPR.2014.242.

Hu, S., Liang, Y., Ma, L. & He, Y. (2009). MSMOTE: improving classification performance when training data is imbalanced. *Proceedings of the 2009 Second International Workshop on Computer Science and Engineering*, 2, 13–17.

Hu, Y., Wu, X. & He, R. (2017). Attention-Set Based Metric Learning for Video Face Recognition. *2017 4th IAPR Asian Conference on Pattern Recognition (ACPR)*, pp. 97-102. doi: 10.1109/ACPR.2017.43.

Huang, C., Li, Y., Change Loy, C. & Tang, X. (2016). Learning deep representation for imbalanced classification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5375–5384.

Huang, L., Zhao, X. & Huang, K. (2019). Bridging the Gap Between Detection and Tracking: A Unified Approach. *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3999–4009.

Huang, Z., Shan, S., Wang, R., Zhang, H., Lao, S., Kuerban, A. & Chen, X. (2015). A benchmark and comparative study of video-based face recognition on COX face database. *IEEE Transactions on Image Processing*, 24(12), 5967–5981.

Imam, T., Ting, K. M. & Kamruzzaman, J. (2006). z-SVM: an SVM for improved classification of imbalanced data. *Australasian Joint Conference on Artificial Intelligence*, pp. 264–273.

Iman, R. L. & Davenport, J. M. (1980). Approximations of the critical region of the fbietkan statistic. *Communications in Statistics-Theory and Methods*, 9(6), 571–595.

Japkowicz, N. (2001). Concept-learning in the presence of between-class and within-class imbalances. *Conference of the Canadian Society for Computational Studies of Intelligence*, pp. 67–77.

Jeatrakul, P., Wong, K. W. & Fung, C. C. (2010). Classification of imbalanced data by combining the complementary neural network and SMOTE algorithm. *International Conference on Neural Information Processing*, pp. 152–159.

Johnson, J. M. & Khoshgoftaar, T. M. (2019). Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1), 27.

Joshi, M. V., Kumar, V. & Agarwal, R. C. (2001). Evaluating boosting algorithms to classify rare classes: Comparison and improvements. *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*, pp. 257–264.

Khan, S. H., Hayat, M., Bennamoun, M., Sohel, F. A. & Togneri, R. (2017). Cost-Sensitive learning of deep feature representations from imbalanced data. *IEEE transactions on neural networks and learning systems*.

Khreich, W., Granger, E., Miri, A. & Sabourin, R. (2010). Iterative Boolean combination of classifiers in the ROC space: An application to anomaly detection with HMMs. *Pattern Recognition*, 43(8), 2732–2752.

Kim, M.-J., Kang, D.-K. & Kim, H. B. (2015). Geometric mean based boosting algorithm with over-sampling to resolve data imbalance problem for bankruptcy prediction. *Expert Systems with Applications*, 42(3), 1074–1082.

Kiran, M., Tiwari, V., Morin, L.-A. B., Granger, E. et al. (2019). On the Interaction Between Deep Detectors and Siamese Trackers in Video Surveillance. *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 1–8.

Kittler, J., Hater, M. & Duin, R. P. W. (1996, Aug). Combining classifiers. *Proceedings of 13th International Conference on Pattern Recognition*, 2, 897-901 vol.2. doi: 10.1109/ICPR.1996.547205.

Ko, A. H., Sabourin, R. & Britto Jr, A. S. (2008). From dynamic classifier selection to dynamic ensemble selection. *Pattern Recognition*, 41(5), 1718 - 1731.

Kraipeerapun, P., Fung, C. C. & Nakkrasae, S. (2009). Porosity prediction using bagging of complementary neural networks. *International Symposium on Neural Networks*, pp. 175–184.

Krawczyk, B. (2016a). Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4), 221–232. doi: 10.1007/s13748-016-0094-0.

Krawczyk, B. (2016b). Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4), 221–232.

Kubat, M., Matwin, S. et al. (1997). Addressing the curse of imbalanced training sets: one-sided selection. *Machine Learning and Applications, 1997. ICMLA'97 International Conference on*.

Kubat, M., Holte, R. C. & Matwin, S. (1998). Machine learning for the detection of oil spills in satellite radar images. *Machine learning*, 30(2-3), 195–215.

Kuncheva, L. I. & Whitaker, C. J. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2), 181–207.

Kuncheva, L. I., Bezdek, J. C. & Duin, R. P. (2001). Decision templates for multiple classifier fusion, an experimental comparison. *Pattern Recognition*, 34(2), 299 - 314.

Landgrebe, T. C., Paclik, P. & Duin, R. P. (2006). Precision-recall operating characteristic (P-ROC) curves in imprecise environments. *18th International Conference on Pattern Recognition (ICPR'06)*, 4, 123–127.

Lee, H., Park, M. & Kim, J. (2016). Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning. *2016 IEEE international conference on image processing (ICIP)*, pp. 3713–3717.

Leng, Q., Hu, R., Liang, C. & Wang, Y. (2013). Person re-identification based on contextual characteristic. *Electronics Letters*, 49(17), 1074–1076.

Li, Q., Yang, B., Li, Y., Deng, N. & Jing, L. (2013). Constructing support vector machine ensemble with segmentation for imbalanced datasets. *Neural Computing and Applications*, 22(1), 249–256.

Li, X., Wang, L. & Sung, E. (2008). AdaBoost with SVM-based component classifiers. *Engineering Applications of Artificial Intelligence*, 21(5), 785–795.

Lin, T.-Y., Goyal, P., Girshick, R., He, K. & Dollár, P. (2017). Focal loss for dense object detection. *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988.

Lipton, Z. C., Elkan, C. & Naryanaswamy, B. (2014). Optimal thresholding of classifiers to maximize F1 measure. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 225–239.

Littlestone, N. & Warmuth, M. K. (1994). The Weighted Majority Algorithm. *Information and Computation*, 108(2), 212 - 261.

Liu, X.-Y., Wu, J. & Zhou, Z.-H. (2009). Exploratory undersampling for class-imbalance learning. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(2), 539–550.

Liu, Y., Yu, X., Huang, J. X. & An, A. (2011). Combining integrated sampling with SVM ensembles for learning from imbalanced datasets. *Information Processing & Management*, 47(4), 617–631.

López, V., Fernández, A., García, S., Palade, V. & Herrera, F. (2013). An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, 250, 113–141.

Margineantu, D. D. & Dietterich, T. G. (1997). Pruning adaptive boosting. *Machine Learning and Applications, 1997. ICMLA'97 International Conference on*.

Martinel, N. & Foresti, G. L. (2012). Multi-signature based person re-identification. *Electronics Letters*, 48(13), 765–767.

Masko, D. & Hensman, P. (2015). The impact of imbalanced training data for convolutional neural networks.

Matta, F. & Dugelay, J.-L. (2007). Video face recognition: A physiological and behavioural multimodal approach. *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, 4, IV–497.

Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2), 442–451.

Mazzon, R., Tahir, S. F. & Cavallaro, A. (2012). Person re-identification in crowd. *Pattern Recognition Letters*, 33(14), 1828–1837.

Mease, D., Wyner, A. & Buja, A. (2007). Cost-weighted boosting with jittering and over/under-sampling: JOUS-boost. *J. Machine Learning Research*, 8, 409–439.

Mitra, S., Savvides, M. & Kumar, B. V. (2006). Human face identification from video based on frequency domain asymmetry representation using hidden Markov models. *International Workshop on Multimedia Content Representation, Classification and Security*, pp. 26–33.

Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R. & Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data*, 2(1), 1.

Ojala, T., Pietikainen, M. & Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7), 971–987.

Pagano, C., Granger, E., Sabourin, R. & Gorodnichy, D. O. (2012). Detector ensembles for face recognition in video surveillance. *The 2012 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8.

Pagano, C., Granger, E., Sabourin, R., Marcialis, G. L. & Roli, F. (2014). Adaptive ensembles for face recognition in changing video surveillance environments. *Information Sciences*, 286, 75–101.

Parambath, S. P., Usunier, N. & Grandvalet, Y. (2014). Optimizing F-measures by cost-sensitive classification. *Advances in Neural Information Processing Systems*, pp. 2123–2131.

Parkhi, O. M., Vedaldi, A., Zisserman, A. et al. (2015). Deep Face Recognition. *BMVC*, 1(3), 6.

Pastor-Pellicer, J., Zamora-Martínez, F., España-Boquera, S. & Castro-Bleda, M. J. (2013). F-measure as the error function to train neural networks. *International Work-Conference on Artificial Neural Networks*, pp. 376–384.

Pillai, I., Fumera, G. & Roli, F. (2017). Designing multi-label classifiers that maximize F measures: State of the art. *Pattern Recognition*, 61(Supplement C), 394 - 404. doi: https://doi.org/10.1016/j.patcog.2016.08.008.

Pouyanfar, S., Tao, Y., Mohan, A., Tian, H., Kaseb, A. S., Gauen, K., Dailey, R., Aghajanzadeh, S., Lu, Y.-H., Chen, S.-C. et al. (2018). Dynamic sampling in convolutional neural networks for imbalanced data classification. *2018 IEEE conference on multimedia information processing and retrieval (MIPR)*, pp. 112–117.

Prati, R. C., Batista, G. E. & Monard, M. C. (2011). A survey on graphical methods for classification predictive performance evaluation. *IEEE Transactions on Knowledge and Data Engineering*, 23(11), 1601–1618.

Prodromidis, A. L. & Stolfo, S. J. (2001). Cost Complexity-Based Pruning of Ensemble Classifiers. *Knowledge and Information Systems*, 3(4), 449–469.

Radtke, P., Granger, E., Sabourin, R. & Gorodnichy, D. (2013). Adaptive ensemble selection for face re-identification under class imbalance. *International Workshop on Multiple Classifier Systems*, pp. 95–108.

Radtke, P. V., Granger, E., Sabourin, R. & Gorodnichy, D. O. (2014). Skew-sensitive boolean combination for adaptive ensembles–An application to face recognition in video surveillance. *Information Fusion*, 20, 31–48.

Raj, V., Magg, S. & Wermter, S. (2016). Towards effective classification of imbalanced data with convolutional neural networks. *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*, pp. 150–162.

Ranawana, R. & Palade, V. (2006). Optimized Precision-A new measure for classifier performance evaluation. *2006 IEEE International Conference on Evolutionary Computation*, pp. 2254–2261.

Rokach, L. (2009). Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography. *Computational Statistics & Data Analysis*, 53(12), 4046–4072.

Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2), 1–39.

Saeed, U., Matta, F. & Dugelay, J.-L. (2006). Person recognition based on head and mouth dynamics. *Multimedia Signal Processing, 2006 IEEE 8th Workshop on*, pp. 29–32.

Sammon, J. W. (1969). A nonlinear mapping for data structure analysis. *IEEE Transactions on computers*, 18(5), 401–409.

Satta, R., Fumera, G., Roli, F., Cristani, M. & Murino, V. (2011). A multiple component matching framework for person re-identification. *International Conference on Image Analysis and Processing*, pp. 140–149.

Schroff, F., Kalenichenko, D. & Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 815–823.

Seiffert, C., Khoshgoftaar, T. M., Van Hulse, J. & Napolitano, A. (2010). RUSBoost: A hybrid approach to alleviating class imbalance. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 40(1), 185–197.

Smeulders, A. W. M., Chu, D. M., Cucchiara, R., Calderara, S., Dehghan, A. & Shah, M. (2014). Visual Tracking: An Experimental Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7), 1442-1468. doi: 10.1109/TPAMI.2013.230.

Sohn, K., Liu, S., Zhong, G., Yu, X., Yang, M.-H. & Chandraker, M. (2017). Unsupervised domain adaptation for face recognition in unlabeled videos. *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3210–3218.

Soleymani, R., Granger, E. & Fumera, G. (2016a). Classifier Ensembles with Trajectory Under-Sampling for Face Re-Identification. *Proceedings of the International Conference on Pattern Recognition Applications and Methods-Volume 1*, pp. 97-108.

Soleymani, R., Granger, E. & Fumera, G. (2016b). Classifier ensembles with trajectory undersampling for face re-identification.

Soleymani, R., Granger, E. & Fumera, G. (2016c). Loss factors for learning Boosting ensembles from imbalanced data. *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pp. 204–209.

Soleymani, R., Granger, E. & Fumera, G. (2018a). F-measure curves for visualizing classifier performance with imbalanced data. *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*, pp. 165–177.

Soleymani, R., Granger, E. & Fumera, G. (2018b). Progressive boosting for class imbalance and its application to face re-identification. *Expert Systems with Applications*, 101, 271–291.

Soleymani, R., Granger, E. & Fumera, G. (2019). F-Measure Curves: A Tool to Visualize Classifier Performance Under Imbalance. *Pattern Recognition*, 107146.

Stallkamp, J., Ekenel, H. K. & Stiefelhagen, R. (2007). Video-based face recognition on real-world data. *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pp. 1–8.

Stefanowski, J. & Wilk, S. (2008). Selective pre-processing of imbalanced data for improving classification performance. In *Data Warehousing and Knowledge Discovery* (pp. 283–292). Springer.

Sun, Y., Kamel, M. S., Wong, A. K. & Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12), 3358–3378.

Sun, Y., Chen, Y., Wang, X. & Tang, X. (2014a). Deep learning face representation by joint identification-verification. *Advances in neural information processing systems*, pp. 1988–1996.

Sun, Y., Wang, X. & Tang, X. (2014b). Deep learning face representation from predicting 10,000 classes. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1891–1898.

Sun, Y., Liang, D., Wang, X. & Tang, X. (2015a). Deepid3: Face recognition with very deep neural networks. *arXiv preprint arXiv:1502.00873*.

Sun, Y., Wang, X. & Tang, X. (2015b). Deeply learned face representations are sparse, selective, and robust. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2892–2900.

Sze-To, A. & Wong, A. K. (2017). A Weight-Selection Strategy on Training Deep Neural Networks for Imbalanced Classification. *International Conference Image Analysis and Recognition*, pp. 3–10.

Szegedy, C., Ioffe, S., Vanhoucke, V. & Alemi, A. A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. *Thirty-First AAAI Conference on Artificial Intelligence*.

Taigman, Y., Yang, M., Ranzato, M. & Wolf, L. (2014a, June). DeepFace: Closing the Gap to Human-Level Performance in Face Verification. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Taigman, Y., Yang, M., Ranzato, M. & Wolf, L. (2014b). Deepface: Closing the gap to human-level performance in face verification. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1701–1708.

Tang, Y., Zhang, Y.-Q., Chawla, N. V. & Krasser, S. (2009). SVMs modeling for highly imbalanced classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(1), 281–288.

Tao, Q. & Veldhuis, R. (2013). Robust Biometric Score Fusion by Naive Likelihood Ratio via Receiver Operating Characteristics. *IEEE Transactions on Information Forensics and Security*, 8(2), 305-313. doi: 10.1109/TIFS.2012.2231862.

Tao, Q. & Veldhuis, R. (2009). Threshold-optimized decision-level fusion and its application to biometrics. *Pattern Recognition*, 42(5), 823–836.

Ting, K. M. (2000). A comparative study of cost-sensitive boosting algorithms. *Machine Learning and Applications, 2000. International Conference on*.

Trigueros, D. S., Meng, L. & Hartnett, M. (2018). Face Recognition: From Traditional to Deep Learning Methods. *arXiv preprint arXiv:1811.00116*.

Van Rijsbergen, C. (1979). Information retrieval: theory and practice. *Proceedings of the Joint IBM/University of Newcastle upon Tyne Seminar on Data Base Systems*, pp. 1–14.

Viola, P. & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, 1, I–511.

Wang, H., Cui, Z., Chen, Y., Avidan, M., Abdallah, A. B. & Kronzer, A. (2018). Predicting hospital readmission via cost-sensitive deep learning. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 15(6), 1968–1978.

Wang, S., Liu, W., Wu, J., Cao, L., Meng, Q. & Kennedy, P. J. (2016). Training deep neural networks on imbalanced data sets. *Neural Networks (IJCNN), 2016 International Joint Conference on*, pp. 4368–4374.

Wang, S. & Yao, X. (2009). Diversity analysis on imbalanced data sets by using ensemble models. *Computational Intelligence and Data Mining, 2009. CIDM'09. IEEE Symposium on*, pp. 324–331.

Wang, S. & Yao, X. (2012). Multiclass imbalance problems: Analysis and potential solutions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(4), 1119–1130.

Wang, Y., Bao, T., Ding, C. & Zhu, M. (2017). Face recognition in real-world surveillance videos with deep learning method. *2017 2nd International Conference on Image, Vision and Computing (ICIVC)*, pp. 239–243.

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

Wong, Y., Chen, S., Mau, S., Sanderson, C. & Lovell, B. C. (2011). Patch-based probabilistic image quality assessment for face selection and improved video-based face recognition. *CVPR 2011 WORKSHOPS*, pp. 74–81.

Wu, E. Y. G. & Chang, K. (2006). Kernel boundary alignment considering unbalanced data distribution. *IEEE Trans. Knowl. Data Eng*, 17(6), 786–796.

Wu, X., He, R. & Sun, Z. (2015). A Lightened CNN for Deep Face Representation. *CoRR*, abs/1511.02683. Consulted at http://arxiv.org/abs/1511.02683.

Xiao, J., Xie, L., He, C. & Jiang, X. (2012). Dynamic classifier ensemble model for customer classification with imbalanced class distribution. *Expert Systems with Applications*, 39(3), 3668–3675.

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

Yan, R., Liu, Y., Jin, R. & Hauptmann, A. (2003). On predicting rare classes with SVM ensembles in scene classification. *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, 3, III–21.

Yang, S., Chen, L.-F., Yan, T., Zhao, Y.-H. & Fan, Y.-J. (2017). An ensemble classification algorithm for convolutional neural network based on AdaBoost. *Computer and Information Science (ICIS), 2017 IEEE/ACIS 16th International Conference on*, pp. 401–406.

Ye, N. & Sim, T. (2010). Towards general motion-based face recognition. *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 2598–2605.

Yen, S.-J. & Lee, Y.-S. (2009). Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Systems with Applications*, 36(3), 5718–5727.

Zhang, C., Tan, K. C. & Ren, R. (2016). Training cost-sensitive deep belief networks on imbalance data problems. *2016 international joint conference on neural networks (IJCNN)*, pp. 4362–4367.

Zhang, Y., Shuai, L., Ren, Y. & Chen, H. (2018). Image classification with category centers in class imbalance situation. *2018 33rd Youth Academic annual conference of Chinese Association of Automation (YAC)*, pp. 359–363.

Zhao, W., Chellappa, R., Phillips, P. J. & Rosenfeld, A. (2003). Face Recognition: A Literature Survey. *ACM Comput. Surv.*, 35(4), 399–458. doi: 10.1145/954339.954342.

Zhou, S. K., Chellappa, R. & Moghaddam, B. (2004). Visual tracking and recognition using appearance-adaptive models in particle filters. *IEEE Transactions on Image Processing*, 13(11), 1491–1506.

Zhou, Z.-H. & Jiang, Y. (2004). NeC4. 5, neural ensemble based C4. 5. *IEEE Transactions on Knowledge and Data Engineering*, 16(6), 770–773.

Zhu, Z., Luo, P., Wang, X. & Tang, X. (2013). Deep learning identity-preserving face space. *Proceedings of the IEEE International Conference on Computer Vision*, pp. 113–120.

Zou, W., Zhu, S., Yu, K. & Ng, A. Y. (2012). Deep learning of invariant features via simulated fixations in video. *Advances in neural information processing systems*, pp. 3203–3211.