Supplement: Details of Algorithms, Data Analysis, Evaluation Metrics, Baselines for "An Ensemble Learning Approach with Gradient Resampling for Class-imbalance Problems"

Appendix A: Relationship between Algorithms

Figure 1 explains the working flows and relationship between our three algorithms more clearly. Specifically, BCWF adopts the HEM algorithm to obtain the probable difficulty information of each example in training set. Please note that HEM is only used to determine the extent to which a example becomes a potentially hard example *PHE/NHE*, not to directly remove hard examples based on it. Then, as a boosting manner, BCWF detects the classification difficulty of each sample in the current iteration and deletes the samples according to different strategies (i.e., BCWF_h or BCWF_s). In other words, when an example is marked as a hard sample in both the HEM algorithm before training and the current iteration, it will be removed from the training set.

Appendix B: Data Analysis

To show the meaning of hard samples more clearly, we select the 'haberman' data set as an example to draw the sample distribution. The data set has three features (i.e., age, node and year), suitable for three-dimensional graphics display. As shown in Figure 2, the decision boundary of this data set is not obvious. In other words, the data set has class overlap, and the hard examples are mostly distributed in the overlap area, which is an important reason for the failure of the classifier. If we can effectively identify and remove these hard samples, the efficiency of the classifier will be improved.

Table 1 reports some statistical indicators for each data set. Note that the hard sample means that it is misclassified by at least half of the sub-classifiers, otherwise it is a easy sample. Further, in order to explore the relationship between various indicators and model performance, we checked the highest AUC value (Rank 1) on various data sets. From the table, we can conclude that *Ratio3* is highly related, i.e.,

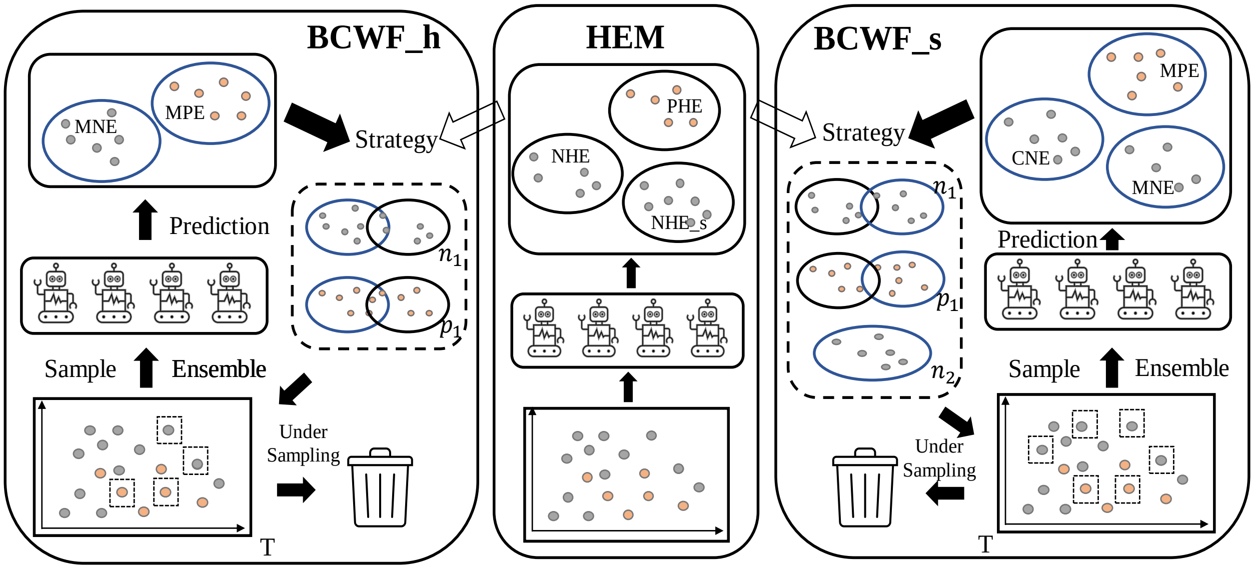

Figure 1 **The working flows and relationship between algorithms.**

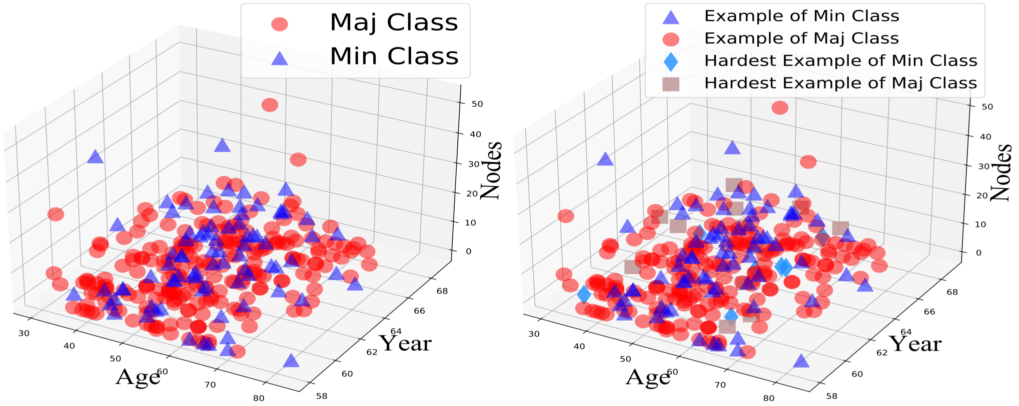

Figure 2 **Hardest examples of haberman data set. Maj and Min are abbreviations for majority and minority respectively.**

negative correlation, to the model's classifier performance, where the correlation coefficient between *Ratio3* and AUC is -0.8210. For the data sets 'wdbc', 'yeast' and 'letter', all of them have a high AUC score with a lower hard examples ratio. We can notice here that the imbalance ratio of the data set 'wdbc' is 1.7, while the imbalance ratio for 'letter' is 24.3. However, the 'letter' data set achieves the highest AUC score out of all the ten data sets. The 'balance' and 'haberman' data sets have low AUC scores, with a hard examples ratio of more than 0.3. All these prove that the hard samples ratio has a significant impact on model performance.

Appendix C: Evaluation Metrics

Generally, the traditional metrics such as error rate can not reflect the learning model's true classification performance on imbalanced data set problems. In this paper, we use *AUC* (Bradley 1997), *Positive f1-score* (Roy et al. 2019), *G-mean* (Xie et al. 2020) as the performance evaluation measures. Before introducing AUC, we first need to introduce the Receiver Operating Characteristics (ROC) curve. The ROC curve

Table 1 Hard examples ratio and AUC results

Data sets	*Ratio*	*PHS*	*PES*	*NHS*	*NES*	*Ratio1* (%)	*Ratio2*(%)	*Ratio3*(%)	AUC
wdbc	1.7	8	172	20	283	4.44	6.60	**5.80**	**0.9632**
pima	1.9	87	140	159	265	38.33	37.50	37.79	0.7803
haberman	2.8	15	54	66	125	21.74	34.55	31.15	0.6437
wpbc	3.2	1	39	62	66	2.50	48.44	37.50	0.7220
credit card	3.5	2,023	3,618	54,61	14,398	35.86	27.50	29.35	0.7021
housing	3.8	14	76	149	190	15.56	43.95	38.00	0.6965
yeast	8.1	6	133	89	1,033	4.32	7.93	**7.53**	**0.9429**
abalone	9.7	26	307	1,025	2,192	7.81	31.86	29.61	0.7952
balance	11.8	10	32	392	96	23.81	80.16	75.71	0.6528
letter	24.3	29	642	694	15,635	4.32	4.25	**4.25**	**0.9946**

Notes. *PHS* denotes positive hard samples, *PES* means positive easy samples. *NHS* refers to negative hard samples and *NES* means negative easy samples. Moreover, *Ratio1* refers to the percentage of positive hard examples in the positive class, *Ratio2* indicates the percentage of negative hard examples in the negative class, and *Ratio3* means the total percentage of hard examples.

comprises two coordinates, i.e., the horizontal coordinate represents the false positive rate, while the vertical coordinate represents the true positive rate (Fawcett 2004),

$$False\ positive\ rate\ (fpr) = FP/(FP + TN),$$

$$True\ positive\ rate\ (tpr) = TP/(TP + FN),$$

where the *TP, FP, TN, FN* represent the cases of predictive examples which are defined in Table 2. Then, AUC is defined as the area under the ROC curve. The larger the AUC, the better the model's classification ability.

The Positive f1-score is defined as a harmonic mean of precision and recall, where a positive f1-score attains its best value at 1 (perfect precision and recall) and its worst value at 0. The definition is as follows,

$$True\ negative\ rate\ (tnr) = TN/(TN + FP),$$

$$Positive\ class\ precision\ (pp) = TP/(TP + FP),$$

$$Negative\ class\ precision\ (np) = TN/(FN + TN),$$

$$Positive\ f1 - score\ (pf1) = \frac{2 * pp * tpr}{pp + tpr}.$$

G-mean is based on the classification accuracy of the minority class and the majority class. It is usually used as an evaluation metric to measure the overall classification performance of an imbalanced data set. A higher G-mean value indicates that the classifier has a good performance on both the majority class and the minority class samples.

$$G - mean = \sqrt{fpr * tnr}$$

The above three metrics are widely used to measure the performance of the algorithm on imbalanced data sets (Lutu and Engelbrecht 2013, Xie et al. 2020, Roy et al. 2019, Razavi-Far et al. 2019). Due to the different focuses of each metric, the use of the three metrics to measure together can better illustrate the superiority of our algorithm.

Appendix D: Baseline Definition

We have selected eleven typical baselines that are widely used to deal with imbalanced problems for comparison. The details are as follows,

Table 2 Confusion matrix

	Predicted Positive Class	Predicted Negative Class
Actual Positive Class	TP (true positives)	FN (false negatives)
Actual Negative Class	FP (false positives)	TN (true negatives)

(1) **MDO**. Mahalanobis Distance-based Over-sampling (MDO) (Abdi and Hashemi 2015) uses the entire data set after the minority examples are over-sampled for training. Classification and regression trees (CART) is used to train weak classifiers. The total number of iterations is 150.

(2) **SmoteSvm** (Veganzones and Séverin 2018). The hybrid method combining smote and support vector machine has achieved significant effects in dealing with imbalanced data sets for bankruptcy prediction.

(3) **KNSMOTE**. Kmeans-Smote (KNSMOTE) (Xu et al. 2021) is a simple and effective over-sampling method based on k-means clustering and smote, which avoids the generation of noise and both inter- and intra-class of imbalance.

(4) **Bagging**. Bagging (Breiman 1996) uses the entire training data set for training. CART is used to train weak classifiers. The number of iterations is 150.

(5) **AdaBoost**. AdaBoost (Schapire 1999) uses the entire training data set for training. CART is used to train weak classifiers. The number of iterations is 150.

(6) **Focal Loss** (Lin et al. 2017). Focal loss is modified on the basis of the standard cross entropy loss, which can reduce the weight of easy-to-classify samples to make the model focus more on difficult-to-classify samples during training.

(7) **FIRE-DES++**. FIRE-DES++ (Cruz et al. 2019) is an enhanced dynamic ensemble system that removes noise and reduces class overlap in the validation set.

(8) **RUSBoost** (Seiffert et al. 2009). RUSBoost incorporates random under-sampling into boosting process of Adaboost, using a balanced sub-dataset for training of weak classifiers in each iteration.

(9) **EasyEnsemble** (Liu et al. 2009). In EasyEnsemble, CART is used to train weak classifiers. The number of sampled subsets T is 15, the number of rounds in each AdaBoost ensemble, i.e., si is 10.

(10) **Self Paced Ensemble** (Liu et al. 2020a). Self paced (SP) ensemble not only considers the imbalance ratio of the sample, but also considers the classification hardness, which can effectively identify the noise in the sample and solve the problem of class overlap.

(11) **MESA** (Liu et al. 2020b). MESA uses a meta-sampler to adaptively resample the training set in iterations to obtain multiple classifiers and form a cascaded ensemble mode, and decouple training and meta-training to adapt to new tasks.

(12) **Hard Examples Mining-Adaboost** (**HEM-AdaBoost**). HEM-AdaBoost is an assembled method which is designed by combining our propose HEM algorithm and AdaBoost. Specifically, we first use the HEM algorithm to find the hardest examples of both classes and remove these samples from the train set. The left examples of train set are used to train AdaBoost classifier. In AdaBoost classifier, CART is used to train weak classifiers, where the total number of iterations is 150.

(13) **Balanced cascade with hard filter (BCWF_h)**. BCWF_h is the proposed Algorithm 2. CART is used to train weak classifiers, the number of subsets T is set as 15, while the number of rounds in each AdaBoost ensemble, i.e, si is 10.

(14) **Balanced cascade with soft filter (BCWF_s)**. BCWF_s is the proposed Algorithm 3 whose parameters are the same with those in BCWF_h.

Appendix E: Algorithm Comparison

Table 3 Comparison between algorithms

Algorithm	Number of subsets	Classifiers	Delete strategy	Pos/Neg ratio
HEM	Variable	Not independent	Random negative sample	Fixed
BCWF_h	Fixed	Not independent	Most hard samples	Uncertain
BCWF_s	Fixed	Not independent	Hardest positive sample Harder negative sample Easy negative sample	Balanced

Table 3 shows the difference between the three algorithms. The main differences between the three algorithms are as follows: (i) The number of subsets of the HEM algorithm is automatically determined according to imbalanced degree of the data, while the other two algorithms are determined by artificial T. Besides, the BCWF algorithms actually include the HEM algorithm, which is used as a means of sample difficulty estimation before training. (ii) BCWF_h removes the hardest negative examples, as these examples are misclassified by all the subclassifiers; by contrast, BCWF_s removes negative examples with a gradient larger than 0, which means that these examples are misclassified by more than half of the subclassifiers. (iii) Since we think that too many easy examples will dominate the optimization direction of the model, BCWF_s will also discard some correctly classified negative examples *CHE* during the iteration process, making the data set balanced after the iteration.

References

Abdi L, Hashemi S (2015) To combat multi-class imbalanced problems by means of over-sampling techniques. *IEEE Transactions on Knowledge and Data Engineering* 28(1):238–251.

Bradley AP (1997) The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition* 30(7):1145–1159.

Breiman L (1996) Bagging predictors. *Machine Learning* 24(2):123–140.

Cruz RM, Oliveira DV, Cavalcanti GD, Sabourin R (2019) Fire-des++: Enhanced online pruning of base classifiers for dynamic ensemble selection. *Pattern Recognition* 85:149–160.

Fawcett T (2004) Roc graphs: Notes and practical considerations for researchers. *Machine Learning* 31(1):1–38.

Lin TY, Goyal P, Girshick R, He K, Doll¨¢r P (2017) Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PP(99):2999–3007.

Liu XY, Wu J, Zhou ZH (2009) Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems Man and Cybernetics Part B* 39(2):539–550.

Liu Z, Cao W, Gao Z, Bian J, Chen H, Chang Y, Liu TY (2020a) Self-paced ensemble for highly imbalanced massive data classification. *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, 841–852 (IEEE).

Liu Z, Wei P, Jiang J, Cao W, Bian J, Chang Y (2020b) Mesa: boost ensemble imbalanced learning with meta-sampler. *Advances in Neural Information Processing Systems* 33:14463–14474.

Lutu PE, Engelbrecht AP (2013) Positive-versus-negative classification for model aggregation in predictive data mining. *INFORMS Journal on Computing* 25(4):792–807.

Razavi-Far R, Farajzadeh-Zanjani M, Wang B, Saif M, Chakrabarti S (2019) Imputation-based ensemble techniques for class imbalance learning. *IEEE Transactions on Knowledge and Data Engineering* .

Roy A, Qureshi S, Pande K, Nair D, Gairola K, Jain P, Singh S, Sharma K, Jagadale A, Lin YY, et al. (2019) Performance comparison of machine learning platforms. *INFORMS Journal on Computing* 31(2):207–225.

Schapire RE (1999) A brief introduction to boosting. *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, volume 99, 1401–1406.

Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A (2009) Rusboost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 40(1):185–197.

Veganzones D, Séverin E (2018) An investigation of bankruptcy prediction in imbalanced datasets. *Decision Support Systems* 112:111–124.

Xie Y, Qiu M, Zhang H, Peng L, Chen Z (2020) Gaussian distribution based oversampling for imbalanced data classification. *IEEE Transactions on Knowledge and Data Engineering* .

Xu Z, Shen D, Nie T, Kou Y, Yin N, Han X (2021) A cluster-based oversampling algorithm combining smote and k-means for imbalanced medical data. *Information Sciences* 572:574–589.