A Skew-Sensitive Evaluation Framework for Imbalanced Data Classification

Min Du * Nesime Tatbul * Brian Rivers Akhilesh Kumar Gupta Lucas Hu
Wei Wang Ryan Marcus Shengtian Zhou Insup Lee Justin Gottschlich
(Correspondence to: Min Du <min.du.email@gmail.com>, Nesime Tatbul <tatbul@csail.mit.edu>)

Scan me!

Abstract

Class distribution skews in imbalanced datasets may lead to models with prediction bias towards majority classes. In this paper, we propose a simple and general-purpose evaluation framework for imbalanced data classification that is sensitive to arbitrary skews in class cardinalities and importances.

Key Design Principles

- ***Simplicity:*** It should be intuitive and easy to use and interpret.
- ***Generality:*** It should be general-purpose, i.e., (i) extensible to an arbitrary number of classes and (ii) customizable to any application domain.

Skew-Sensitive Evaluation Framework

Weighted Balanced Accuracy (WBA)

Suppose we are given a test dataset with N data items and C distinct classes: $N = \sum_{i=1}^{C} n_i$

Assume a classifier correctly predicts p_i out of n_i:

$$Accuracy = \frac{\sum_{i=1}^{C} p_i}{N}$$

Macro-average of $Accuracy_i$:

$$BalancedAccuracy = \frac{1}{C} \times \sum_{i=1}^{C} Accuracy_i$$

Generalize into $WeightedBalancedAccuracy$ by extending it with per-class importance weights w_i:

$$WeightedBalancedAccuracy = \sum_{i=1}^{C} w_i \times Accuracy_i$$

Weight Customization

Importance criteria = User-defined

Importance criteria = Rarity

$$w_i = r_i = \frac{1}{f_i \times \sum_{j=1}^{C} \frac{1}{f_j}}$$

Multiple importance criteria

$$w_i = \frac{\prod_{j=1}^{M} m_{i,j}}{\sum_{k=1}^{C} \prod_{j=1}^{M} m_{k,j}}$$

Partially-defined importance criteria: support the case when not all of the class weights are supplied by the user

Metric Customization

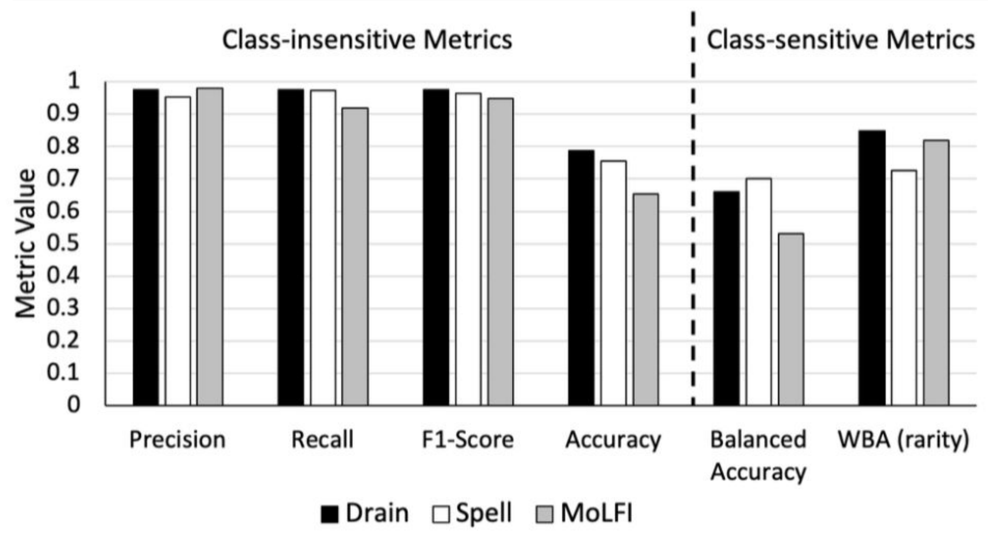
Our framework can be easily extended to other metrics such as *Precision, Recall, and F-Score.*

Model Training Improvement

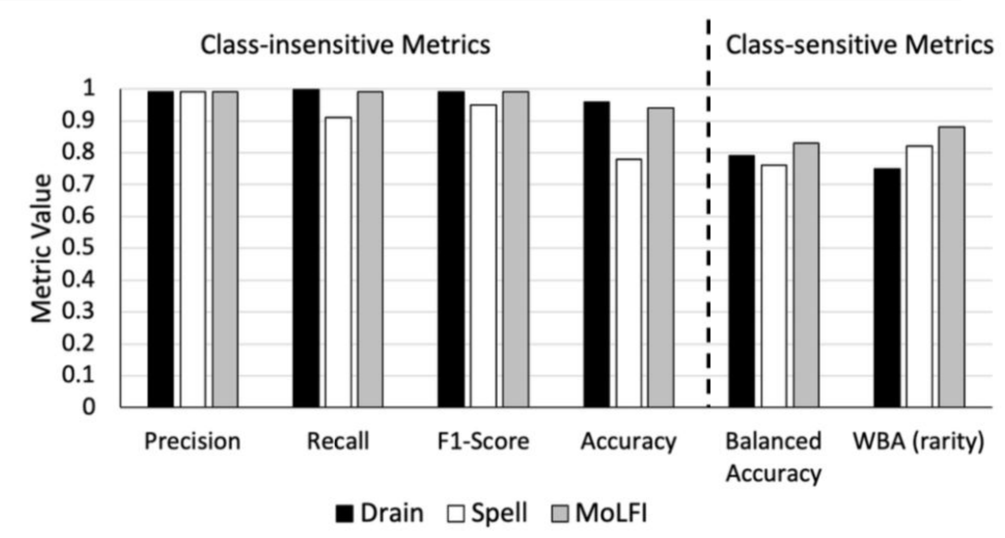
Using $Loss_i$ to denote the total loss incurred by all samples within class i, with our proposed class weights w_i, the model training loss: $\mathcal{L} = \sum_{i=1}^{C} Loss_i$

$$\mathcal{L} = \sum_{i=1}^{C} w_i \times Loss_i$$

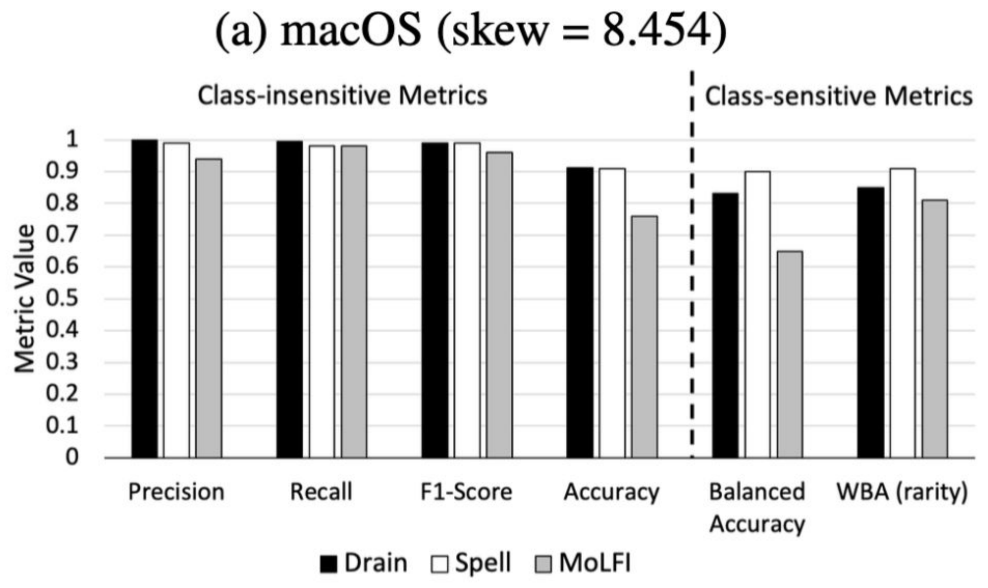
Use Case 1: Learned Log Parsing



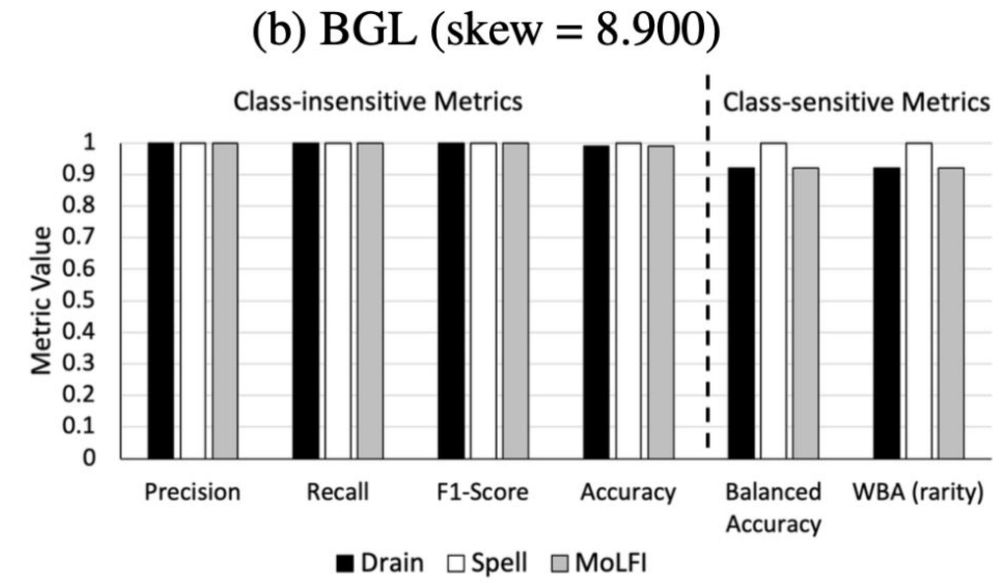
(a) macOS (skew = 8.454)



(b) BGL (skew = 8.900)



(c) Android (skew = 4.822)

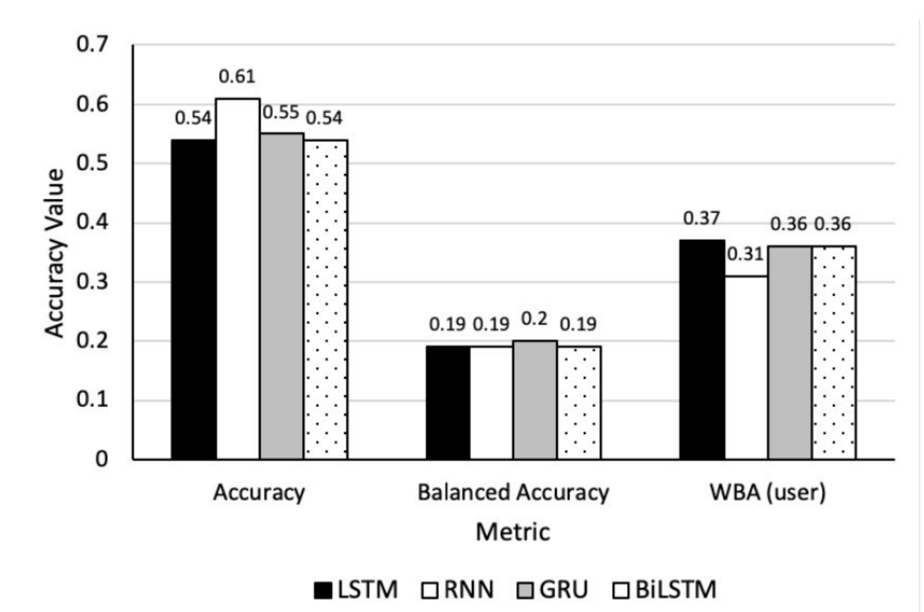


(d) HDFS (skew = 0.202)

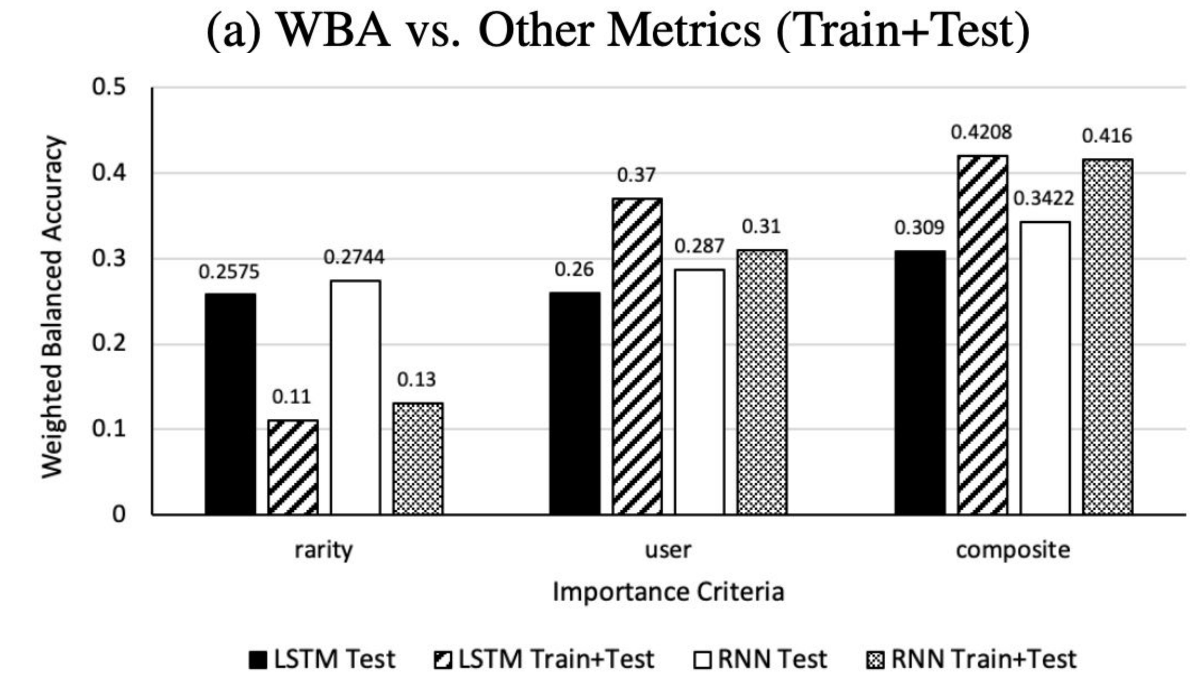
Use Case 2: Sentiment Analysis

Table: Amazon per-class breakdown

Class	Frequency	Weights	LSTM	RNN	GRU	BiLSTM
1	0.092	0.7	0.19	0.04	0.16	0.17
2	0.052	0	0	0	0	0
3	0.075	0	0	0	0	0
4	0.142	0	0	0	0	0
5	0.639	0.3	0.81	0.96	0.84	0.83



(a) WBA vs. Other Metrics (Train+Test)



(b) WBA: Test vs. Train+Test

Use Case 3: URL Classification

Category	#URLs	Rarity w_i	User w_i	Classification Accuracy: Service A	Service B	Service C	Service D	Ranking
benign	16762	0.04	0.05	0.761	0.815	0.661	0.853	DBAC
NSFW	5276	0.14	0.05	0.965	0.804	0.533	0.767	ABDC
malware	1913	0.38	0.8	0.890	0.845	0.602	0.872	ADBC
phishing	1675	0.44	0.1	0.968	0.811	0.521	0.771	ABDC
Accuracy				**0.826**	0.815	0.621	**0.831**	DABC
Balanced Accuracy				**0.896**	0.819	0.579	**0.816**	ABDC
WBA_{rarity}				**0.929**	**0.823**	0.559	**0.812**	ABDC
WBA_{user}				**0.895**	**0.838**	0.593	**0.856**	ADBC

Above: Evaluating and ranking the URL classification services

Right: Training and evaluating a URLNet model using WBA

Category	Dataset Statistics: Train #URLs	Test #URLs	Rarity w_i	User w_i	Classification Accuracy: Train with no WBA	Train with rarity w_i	Train with user w_i
benign	10000	6762	0.04	0.05	0.981	0.300	0.458
NSFW	3150	2126	0.14	0.15	**0**	**0.608**	0.418
malware	1143	770	0.38	0.45	**0.705**	**0.839**	**0.895**
phishing	1000	675	0.44	0.35	0.782	0.788	0.754
Test with Accuracy					0.745	0.435	0.502
Test with Balanced Accuracy					0.617	0.634	0.631
Test with WBA_{rarity}					**0.653**	**0.761**	NA
Test with WBA_{user}					**0.640**	NA	**0.752**