Deep Learning and Thresholding with Class-Imbalanced Big Data

Justin M. Johnson, Taghi M. Khoshgoftaar
College of Engineering and Computer Science
Florida Atlantic University
Boca Raton, Florida 33431
jjohn273@fau.edu, khoshgof@fau.edu

***Abstract*—Class imbalance is a regularly occurring problem in machine learning that has been studied extensively over the last two decades. Various methods for addressing class imbalance have been introduced, including algorithm-level methods, data-level methods, and hybrid methods. While these methods are well studied using traditional machine learning algorithms, there are relatively few studies that explore their application to deep neural networks. Thresholding, in particular, is rarely discussed in the deep learning with class imbalance literature. This paper addresses this gap by conducting a systematic study on the application of thresholding with deep neural networks using a *Big Data* Medicare fraud data set. We use random over-sampling (ROS), random under-sampling (RUS), and a hybrid ROS-RUS to create 15 training distributions with varying levels of class imbalance. With the fraudulent class size ranging from 0.03%–60%, we identify optimal classification thresholds for each distribution on random validation sets and then score the thresholds on a 20% holdout test set. Through repetition and statistical analysis, confidence intervals show that the default threshold is never optimal when training data is imbalanced. Results also show that the optimal threshold outperforms the default threshold in nearly all cases, and linear models indicate a strong linear relationship between the minority class size and the optimal decision threshold. To the best of our knowledge, this is the first study to provide statistical results that describe optimal classification thresholds for deep neural networks over a range of class distributions.**

***Keywords*—Artificial Neural Networks, Deep Learning, Class Imbalance, Thresholding, Data Sampling, Fraud Detection**

1. Introduction

Class imbalance is a common occurrence in many real-world data problems, e.g. medical diagnosis [1] and fraud detection [2]. Given a binary data set where samples are assigned to one of two distinct groups, class imbalance exists when one class is significantly larger than the other. These groups are often referred to as the majority and minority groups, or classes, respectively, and in many problems the minority group is the class of interest, or positive class [3], [4], [5], [6]. One popular example is the task of predicting the presence of a medical disease, where the majority of the patients are healthy and detecting the disease is of greater interest. In this example, the majority group of healthy patients is referred to as the negative class.

Training effective predictive models with these imbalanced data sets can be very difficult, and non-standard machine learning methods are often required to achieve desirable results. When class imbalance exists within training data, learners will typically over-classify the majority group due to its increased prior probability [7]. As a result, the instances belonging to the minority group are misclassified more often than those belonging to the majority group. These adverse effects are often compounded by the challenges of working with big data [8], [9] and class rarity [10]. Studies show that performance generally degrades as the level of class imbalance increases [11], i.e. as the relative size of the minority class becomes smaller. We denote this level of class imbalance using n_{neg}:n_{pos}, where n_{neg} and n_{pos} correspond to the relative number of samples in the negative and positive class. Moreover, popular performance metrics such as accuracy or error rate can mislead analysts with scores that falsely indicate good performance. For example, given a binary data set with a positive class distribution of 1%, a naïve learner that always outputs the negative class label for all inputs will achieve 99% accuracy.

The challenges and solutions associated with modeling class-imbalanced data have been studied extensively with traditional machine learning algorithms [12], [13], [14], [15], but deep learning with class-imbalanced data has received relatively little attention [16]. This gap can be attributed to deep learning, arguably popularized circa 2013 [17], being relatively immature compared to more traditional machine learning techniques. Furthermore, unlike data sets *in the wild*, many of the popular benchmark data sets used to evaluate deep models contain mostly balanced class distributions. Our literature review on deep learning with class imbalance shows that existing research focuses primarily on solving class-imbalanced image tasks with the convolutional neural network (CNN). These existing studies address class imbalance with data sampling [18], [19], [20], loss functions that balance class-wise weight updates [21], [22], [23], or complex hybrid methods that leverage deep representation learning [24], [25], [26]. The thresholding method for addressing class imbalance, however, is rarely discussed. Moreover, the few studies that do employ thresh-

978-1-7281-4550-1/19/$31.00 ©2019 IEEE
DOI 10.1109/ICMLA.2019.00134

olding do not provide clear implementation details and, in our opinion, do not adequately emphasize the importance of thresholding.

While it does not appear in deep learning research, thresholding with traditional machine learning algorithms has been widely used to improve classification results [27], [28], [29]. Thresholding is an algorithm-level method that increases or decreases the bias towards a particular class by changing the classification threshold that is used to assign class labels to probability estimates [30]. Given a classifier with the default decision threshold of 0.5, the positive class label is assigned when the classifier estimates a posterior probability greater than or equal to 0.5. Decreasing this threshold allows learners to assign the positive class label to observations with lower confidence, potentially increasing the total number of correctly classified positive samples. For example, a perfect true positive rate (TPR) can be achieved by using a decision threshold of 0.0. This model would be useless of course, because it is guaranteed to misclassify all negative samples as positive. Consequently, the tradeoff between class-wise performance scores must be considered when selecting an optimal threshold. Since neural networks and deep models estimate Bayesian posterior probabilities [31], we believe thresholding should be used whenever training data is imbalanced. Provost [32] goes as far as stating that using classifiers with imbalanced data and a default threshold may be a "critical mistake".

To address this gap in deep learning research, we conduct a systematic study that uses thresholding to address class imbalance in a severely imbalanced Medicare data set. The Medicare data set [33] has a class distribution of 99.97:0.03, i.e. the positive fraudulent samples makes up just 0.03% of the entire data set. Through random over-sampling (ROS), random under-sampling (RUS), and a hybrid ROS-RUS we create 15 new training distributions with positive class sizes in the range 0.1%–60%. We present a thresholding procedure that identifies optimal classification thresholds on a validation set and use a 20% holdout test set to evaluate the performance of these thresholds. Through repetition and statistical analysis, we provide 95% confidence intervals for optimal decision thresholds and find that the default threshold of 0.5 is never optimal when networks are trained with class-imbalanced data. In addition, we show that class-wise performance scores achieved with the optimal threshold are significantly better than the default threshold when training data is imbalanced. Finally, linear models are used to illustrate a strong relationship between the minority class size and the optimal decision threshold. To the best of our knowledge, this is the first study to provide a statistical analysis of optimal classification thresholds for deep neural network classification with imbalanced data.

The remainder of this paper is structured as follows. Section 2 discusses related works. Section 3 details the data set and methods used throughout the experiment. Section 4 presents our findings, and Section 5 concludes with suggestions for future work.

2. Related Work

Anand et al. [34] studied the effects of class imbalance on the backpropagation algorithm in shallow networks. The authors show how the optimization process becomes dominated by the majority class. This biased optimization causes the majority class error to reduce rapidly, but this is usually at the expense of increasing the minority class error.

We explored 15 deep learning methods for addressing class imbalance in a survey paper [16] and found that the area of deep learning with class imbalance, and big data, is still relatively understudied. Several authors explored data sampling methods [18], [19], [20] and found ROS to outperform RUS and baseline models most of the time. Others employed cost-sensitive loss functions or proposed new loss functions that reduce the bias towards the majority class [21], [22], [23], [35]. Some of the best results were achieved by more complex hybrid methods that leverage deep feature learning and custom loss functions [24], [25], [26]. Out of all methods explored, however, we found that there was very little mention of thresholding.

Lin et al. [23] proposed the *Focal Loss* function for addressing the severe class imbalance found in object detection problems. While their study is not specifically about thresholding, they do disclose using a threshold of 0.05 to speed up inference. Dong et al. [26] also present a loss function for addressing class imbalance, i.e. the *Class Rectification Loss*. They compare their proposed loss function to a number of alternative methods, including thresholding. Results from Dong et al. show thresholding outperforms ROS, RUS, cost-sensitive learning, and other baseline models on the imbalanced X-Domain [36] image data set. These studies were not intended to showcase thresholding, yet, their results clearly indicate that thresholding plays an important role in classifying imbalanced data with deep models.

To the best of our knowledge, Buda et al. [20] were the only authors to explicitly isolate the thresholding method and study its ability to improve the classification of imbalanced data with deep models. ROS and RUS were used to create training distributions with varying levels of class imbalance from the MNIST [37] and CIFAR-10 [38] benchmarks, and the authors evaluated minority class sizes between 0.02%–50%. Thresholding was achieved by dividing CNN outputs by prior class probabilities, and the accuracy performance metric was used to show how thresholding improves class-wise performance in nearly all cases. In addition to outperforming ROS, RUS, and the baseline CNN, the authors show that combining thresholding with ROS performs exceptionally well and outperforms plain ROS. It is difficult to properly interpret these results, however, as the accuracy summary metric is misleading when classes are imbalanced, and the accuracy metric does not give any insight into class-specific tradeoffs. We expand on this work by reporting multiple complementary performance metrics and by estimating the significance of observed results with repetition and statistical analysis.

The Medicare data set used in this study is taken from a big data study by Herland et al. [33]. The authors used

the List of Excluded Individuals/Entities (LEIE) [39] to label claims data from the Centers of Medicare and Medicaid Services (CMS) [40] as fraudulent or non-fraudulent. Additional details about the data set will be discussed in Section 3.1. They use Part B, Part D, and DMEPOS claims data to perform cross-validation with logistic regression (LR), Random Forest (RF), and Gradient Boosted Tree (GBT) learners. Herland et al. also create a fourth data set that combines all three data sets to determine if learners should be trained on each data set independently, or on all available data. Results show that the combined and Part B data sets score significantly better on the *area under the Receiver Operating Characteristic curve* (ROC AUC) [41] metric than the other data sets. The LR learner is also shown to outperform the GBT and RF learners with a max AUC of 0.816. We employ deep neural network (DNN) models with data sampling and thresholding onto the Part B data set and improve upon these results significantly.

3. Methods

The value of thresholding is evaluated by fitting DNN models to an 80% training set and scoring performance on a 20% test set. Similar to hyperparameters, optimal decision thresholds are discovered by maximizing class-wise performance on a validation set. Experiments are carried out using the Keras [42] open-source deep learning library with its default backend, i.e. TensorFlow [43]. Statistical analysis is used to estimate the significance of optimal decision thresholds and classification results.

3.1. Data Sets

This study uses the 2012–2016 Medicare Part B data sets provided by CMS [40]. The Medicare Part B claims data describes the services and procedures that healthcare professionals provide to Medicare's Fee-For-Service beneficiaries. Records contain various provider-level attributes, including a unique 10-digit identification number for providers, i.e. the National Provider Identifier (NPI) [44], and the provider specialty type. Other attributes describe the provider's activity within Medicare over a single year, e.g. procedures performed, average charges submitted to Medicare, and average payments by Medicare. Procedures rendered are encoded using the Healthcare Common Procedures Coding System (HCPCS) [45]. CMS releases data annually and aggregates the data over: (1) provider NPI, (2) HCPCS code, and (3) place of service. This produces one record for each provider, HCPCS code, and place of service combination over a given year.

The LEIE data set lists providers that are prohibited from practicing and is used to label providers within the Medicare Part B data set as fraudulent or non-fraudulent. The Office of Inspector General has the authority to exclude providers from Federally funded healthcare programs for a variety of reasons. Following the work by Bauder and Khoshgoftaar [46], a subset of exclusion types that are indicative of fraud are used to label Medicare providers. We label providers in the Medicare data as fraudulent by matching on NPI numbers, and we consider all claims prior to the provider's exclusion date to be fraudulent.

For each year, records are grouped by NPI and provider type. Each group is converted into a single record of summary statistics, i.e. minimum, maximum, sum, median, mean, and standard deviation. Categorical features are one-hot encoded, and stratified random sampling is used to set aside a 20% test set. A min-max scaler is fit to the training data and used to transform the attributes of the training and test sets to the range $[0, 1]$. Training and test set details are illustrated in Table 1.

TABLE 1. TRAINING AND TEST DATA SETS' DETAILS

Data Set	Total Samples	Fraudulent Samples	% Fraudulent
Training Data	3,753,896	1206	0.032%
Test Data	938,474	302	0.032%

Additional details about the data set and the preprocessing steps can be read in the original paper by Herland et al. [33].

3.2. Hyperparameters

Hyperparameters are defined through a random search procedure by averaging validation results over 10 runs. All models are trained using the Adam optimizer [47] with mini-batch sizes of 256 and default moment estimate decay rates. The Rectified Linear Unit (ReLU) activation function [48] is used in all hidden layer neurons, and the sigmoid activation function is used at the output layer.

The ROC AUC metric is used to monitor training and validation performance, because traditional loss and accuracy metrics can be misleading when classes are highly imbalanced. Validation results show that two hidden layers composed of 32 neurons provides sufficient capacity to overfit the training data. We apply batch normalization [49] before hidden-layer activations to speed up training and improve performance on the validation set. Dropout is applied after hidden-layer activations to reduce overfitting [50]. The details of this baseline architecture are listed in Table 2. To determine how network depth affects the optimal decision threshold and model performance, we extend this model to four hidden layers following the same pattern.

3.3. Class Imbalance Methods

We study how the minority class size affects the optimal decision threshold and classification performance by creating 15 new Medicare training distributions. We vary the levels of class imbalance by applying ROS, RUS, and ROS-RUS with varying sampling rates. The range of distributions selected demonstrate the effect of class imbalance and allow for the interpolation of additional results. The details of each distribution, including the size of the positive and negative class after sampling, are listed in Table 3. The first row describes the training data prior to data sampling, and the

TABLE 2. BASELINE ARCHITECTURE

Layer Type	# of Neurons	# of Parameters
Input	125	0
Dense	32	4032
Batch Normalization	32	128
ReLU Activation	32	0
Dropout $P = 0.5$	32	0
Dense	32	1056
Batch Normalization	32	128
ReLU activation	32	0
Dropout $P = 0.5$	32	0
Dense	1	33
Sigmoid activation	1	0

remaining rows provide the size of the positive and negative classes after applying data sampling. The total number of samples in the training set is denoted by N_{train}, which is equal to $n_{neg} + n_{pos}$.

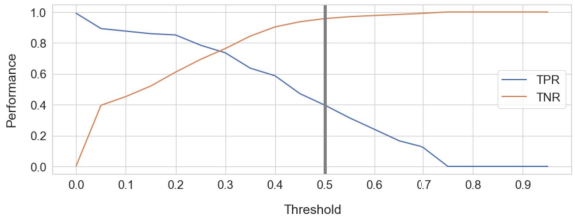
TABLE 3. TRAINING DATA DISTRIBUTIONS

Distribution	n_{neg}	n_{pos}	N_{train}	n_{neg}:n_{pos}
Baseline	3,377,421	1,085	3,378,506	99.97:0.03
RUS-1	975,737	1,085	976,822	99.9:0.1
RUS-2	107,402	1,085	108,487	99:1
RUS-3	4,390	1,085	5,475	80:20
RUS-4	1,620	1,085	2,705	60:40
RUS-5	1,085	1,085	2,170	50:50
RUS-6	710	1,085	1,795	40:60
ROS-1	3,377,421	3,381	3,380,802	99.9:0.1
ROS-2	3,377,421	33,635	3,411,046	99:1
ROS-3	3,377,421	844,130	4,221,551	80:20
ROS-4	3,377,421	2,251,375	5,628,796	60:40
ROS-5	3,377,421	3,377,421	6,754,842	50:50
ROS-6	3,377,421	5,064,780	8,442,201	40:60
ROS-RUS-1	1,688,710	1,688,710	3,377,420	50:50
ROS-RUS-2	844,355	844,355	1,688,710	50:50
ROS-RUS-3	337,742	337,742	675,484	50:50

RUS decreases class imbalance by randomly sampling from the majority class without replacement. Large reduction rates are required to balance the Medicare fraud classes because the original data set is severely imbalanced. For example, RUS-2 combines the positive group with a negative class sample that is just 3.18% of the original negative group, and the resulting data set is still considered highly imbalanced at 99:1. One advantage of RUS is that it can drastically reduce the size of the training data. Smaller training data reduces memory overhead and allows for faster training times, making it ideal for hyperparameter tuning and big data problems [51]. If the sample taken from the majority group is too small, however, RUS risks under representing the majority class. Training models with a majority class sample that is not representative of the population may degrade test performance [52].

ROS constructs class-balanced training sets by randomly duplicating samples from the minority class. In addition to reducing class imbalance, ROS has the advantage of using all available training data. Since there are many more non-fraud cases than there are fraud in the Medicare data set, the fraud cases must be over-sampled at high rates to balance out class distributions. For example, creating a 50:50

Figure 1. Performance Tradeoffs with Varying Thresholds (ROS-3)



class-balanced training set with ROS requires sampling the minority class at a rate of 3112%. This means that ROS nearly doubles the size of the Medicare training set in order to balance the classes.

Finally, ROS and RUS are combined (ROS-RUS) to increase the total number of balanced distributions that are evaluated. Three new distributions are created by reducing the majority group by 90%, 75%, and 50% while simultaneously over-sampling the minority group until classes are balanced. Higher reduction rates improve efficiency by reducing the size of the training set, whereas lower reduction rates improve the representation of the majority group. This hybrid method balances the training data, reduces the risk of under representing the majority group, and decreases training costs with lower over-sampling rates. We do not create imbalanced distributions with ROS-RUS because the results of ROS and RUS sufficiently demonstrate how class imbalance affects the optimal decision threshold.

3.4. Thresholding Procedure

There is a clear tradeoff between the TPR and TNR that must be taken into account when selecting an optimal classification threshold. We illustrate this tradeoff in Figure 1 using a portion of the validation results from the ROS-3 80:20 distribution. By varying the decision threshold and plotting the resulting TPR and TNR scores, we can see that TPR scores increase as the decision threshold approaches zero. In other words, lowering the threshold decreases the confidence that is required to label a sample as being from the positive class and allows the model to correctly classify more positive samples. There is a tradeoff, however, and the total number of false positives will only increase as more samples are assigned to the positive class with lower confidence. Consequently, lowering the threshold usually harms TNR performance. This TPR to TNR tradeoff is not unique to this specific problem, or neural networks in general, but occurs whenever threshold-based classification rules are used to assign class labels to probability estimates.

Selecting an optimal decision threshold should be driven by the problem definition and requirements. For example, a cancer detection system will usually maximize recall because false negatives are life-threatening. In our Medicare fraud detection system we prefer a high TPR over a high TNR, as detecting fraud is more important than detecting non-fraud. Additionally, we wish to approximately balance the TPR and TNR rates in order to maximize the model's

```
input : targets y, probability estimates p
output: optimal threshold
best_thresh ← curr_thresh ← max_gmean ← 0;
delta_thresh ← 0.0005;
while curr_thresh < 1.0 do
    ŷ ← ApplyThreshold(p, curr_thresh);
    tpr, tnr, gmean ← CalcPerformance(y, ŷ);
    if tpr < tnr then
        return best_thresh;
    end
    if gmean > max_gmean then
        max_gmean ← gmean;
        best_thresh ← curr_thresh;
    end
    curr_thresh ← curr_thresh + delta_thresh;
end
return best_thresh;
```

Algorithm 1: Calculating Decision Thresholds

total predictive power. This will maximize the total number of fraudulent samples captured without introducing an overwhelming number of false positives. We implement a procedure (Algorithm 1) that identifies an optimal decision threshold based on these requirements.

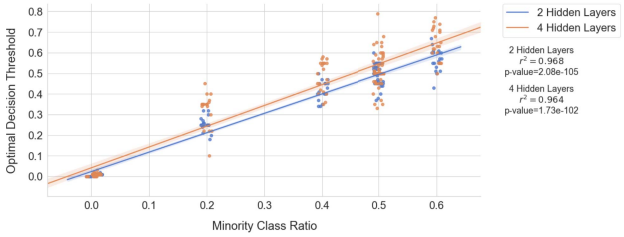
Optimal classification thresholds are defined by validating models on random samples of the training data. More specifically, we create validation data by holding out 10% of the training data using stratified random sampling without replacement. Models are trained for 50 epochs using the remaining 90% of the training data, and our threshold selection procedure is used to find the best threshold on the validation set. To account for variance within the random samples, we repeat this process 10 times using a new validation set for each iteration and then average the thresholds. The average decision thresholds are then evaluated on the unseen test data. Models are retrained using all available training data for 50 epochs, and the previously calculated thresholds are used to assign class labels to the test observations. We chose to retrain models with all available training data because, for many distributions, there are very few fraudulent samples (1085). Alternatively, we could have applied the models that were trained on the subset of the training data, i.e. excluding the validation samples, but we believe class rarity justifies retraining with all available data.

The average thresholds selected during validation would be useless if they did not also maximize class-wise performance on the test set. Therefore, we take the TPR to TNR tradeoff into consideration and report both metrics on the test set. We also report the geometric mean (G-Mean) and use it as the primary metric for comparing optimal threshold results to default threshold results. To add rigor and enable statistical analysis, classification results are averaged over 30 repetitions.

4. Results and Discussion

We begin by presenting the optimal decision thresholds that were calculated during the validation phase. Confidence intervals are used to estimate the significance of the calculated thresholds, and linear models are used to describe the relationship between the selected threshold and the minority

Figure 2. Minority Class Ratio vs Optimal Decision Threshold



class size. Finally, classification results on the test set are presented using both the optimal threshold and the default threshold. Due to space constraints, we only include the two-hidden-layer network results, but note that the four-hidden-layer results are very similar. A significance level of 0.05 is used for all statistical analysis.

4.1. Optimal Decision Thresholds

The average decision thresholds calculated during the validation phase show that the default threshold of 0.5 is never optimal when the training data is imbalanced. To illustrate this, Table 4 lists the average optimal thresholds along with their 95% confidence intervals. The confidence intervals listed in bold are significantly different from the default threshold. Even with relatively low class imbalance levels, e.g. 80:20 or 60:40, class-wise performance is maximized by selecting a decision threshold that is significantly different from the default threshold.

We observe a close relationship between the minority class size and the optimal decision threshold. For example, the Baseline distribution has a minority class ratio of 0.0003, and the average optimal decision boundary calculated on the trained model is 0.0002. Similarly, the ROS-2 distribution has a minority class ratio of 0.01, i.e. 1%, and the average optimal decision boundary was found to be 0.0110. We visualize this relationship by plotting the minority class size against the optimal decision threshold across all distributions in Figure 2. Plots are enhanced with a horizontal jitter of 0.01, and linear models are fit to the data using *Ordinary Least Squares* [53] and 95% confidence bands. To illustrate how network depth affects the optimal threshold, linear models are fit to the two network architectures separately, i.e. two hidden layers versus four hidden layers. The resulting r^2 and p values indicate a strong linear relationship between the minority class size and the average optimal decision threshold. This relationship further suggests the use of thresholding when training deep models with imbalanced data.

4.2. Classification Results

We report the average TPR, TNR, and G-Mean scores that were calculated on the test set over 30 repetitions. All training distributions, even those severely imbalanced, produce reasonable test results with G-Mean scores greater

TABLE 4. COMPARING OPTIMAL THRESHOLDS AND CLASSIFICATION RESULTS WITH TWO HIDDEN LAYERS

Distribution	n_{neg}:n_{pos}	Average Threshold	Threshold Confidence Interval	ROC AUC	Optimal Threshold			Default Threshold			G-Mean Z-test p-value
					TPR	TNR	G-Mean	TPR	TNR	G-Mean	
Baseline	99.97:0.03	0.0002	**(0.0002, 0.0003)**	0.8058	0.8280	0.6099	0.7088	0.0000	1.0000	0.0000	**<0.05**
RUS-1	99.9:0.1	0.0009	**(0.0007, 0.0011)**	0.8102	0.7965	0.6597	0.7243	0.0000	1.0000	0.0000	**<0.05**
RUS-2	99:1	0.0110	**(0.0095, 0.0125)**	0.8124	0.7807	0.6987	0.7383	0.0000	1.0000	0.0000	**<0.05**
RUS-3	80:20	0.2680	**(0.2502, 0.2858)**	0.8076	0.7521	0.7163	0.7338	0.0392	0.9955	0.1421	**<0.05**
RUS-4	60:40	0.4200	**(0.3959, 0.4441)**	0.8043	0.7783	0.6700	0.7212	0.7013	0.7643	0.7320	**<0.05**
RUS-5	50:50	0.4970	(0.4704, 0.5236)	0.8027	0.7864	0.6601	0.7195	0.7853	0.6602	0.7190	0.8546
RUS-6	40:60	0.5730	**(0.5400, 0.6060)**	0.7994	0.7802	0.6588	0.7154	0.8791	0.5102	0.6670	**<0.05**
ROS-1	99.9:0.1	0.0007	**(0.0005, 0.0009)**	0.8114	0.8168	0.6348	0.7193	0.0000	1.0000	0.0000	**<0.05**
ROS-2	99:1	0.0110	**(0.0087, 0.0132)**	0.8383	0.8572	0.6334	0.7338	0.0000	1.0000	0.0000	**<0.05**
ROS-3	80:20	0.2410	**(0.2135, 0.2685)**	0.8484	0.8282	0.6926	0.7549	0.4221	0.9416	0.6268	**<0.05**
ROS-4	60:40	0.4080	**(0.3691, 0.4469)**	0.8454	0.8056	0.7198	0.7582	0.7258	0.8052	0.7629	0.5547
ROS-5	50:50	0.4530	**(0.4150, 0.4910)**	0.8505	0.8084	0.7324	0.7692	0.7977	0.737	0.7657	0.2978
ROS-6	40:60	0.5630	**(0.5169, 0.6091)**	0.8503	0.8163	0.7272	0.7701	0.8483	0.6768	0.756	**<0.05**
ROS-RUS-1	50:50	0.4850	(0.4554, 0.5146)	0.8500	0.8029	0.7354	0.7665	0.8066	0.7242	0.7616	0.5261
ROS-RUS-2	50:50	0.5218	(0.4940, 0.5497)	0.8509	0.7876	0.7553	0.7710	0.8047	0.7272	0.7623	0.1744
ROS-RUS-3	50:50	0.5090	(0.4771, 0.5409)	0.8477	0.8104	0.7209	0.7625	0.8051	0.7224	0.7599	0.7601

than 0.70 when an optimal decision threshold is used. These results imply that the thresholds identified during model validation can be effectively applied to the test set.

Not surprisingly, the classification results reveal a close relationship between ROC AUC scores and G-Mean scores. For example, the distribution with the lowest ROC AUC score (Baseline) has the lowest G-Mean score, while the distribution with the highest ROC AUC score (ROS-RUS-2) has the highest G-Mean score. This relationship intuitively makes sense, as a high ROC AUC score can only be achieved if there exists a threshold that yields a high TPR and TNR. In other words, a high ROC AUC score implies that a relatively high TPR and TNR can be achieved, but only if we identify the required threshold.

To better showcase thresholding performance, Table 4 also provides the classification results achieved using the default threshold. Similar to the optimal threshold results, default threshold results are obtained by fitting models to the training data, scoring on the test set, and averaging over 30 repetitions. On average, the default threshold produces TPR and TNR scores of 0.0 and 1.0 when imbalance levels are high, e.g. 99:1 or worse. A two-sample z-test is used to compare the G-Mean results from the two threshold groups, i.e. optimal versus default, and results that are significantly different are indicated with bold p-values. Based on these results, the default threshold never performs significantly better than the optimal threshold.

On average, the 40:60, 50:50, and 60:40 distributions are the only distributions that perform well with the default threshold of 0.5. While these mostly balanced distributions achieve reasonable G-Mean scores with the default threshold, this threshold does not explicitly prefer one class over the other. For example, the TPR achieved with RUS-4 and the default threshold (0.7013) is significantly less than the TPR achieved with the selected optimal threshold (0.7783). Unlike the default threshold, optimal thresholds defined by our threshold selection procedure specifically prefer a high TPR over a high TNR. The results reflect this design choice, as the TPR score is always greater than the TNR score when using the optimal threshold. Therefore, we conclude that thresholding is valuable even when class distributions are mostly balanced.

5. Conclusion

Thresholding is a popular algorithm-level method for addressing class imbalance. While it is commonly used to address class imbalance with traditional machine learning algorithms, there are very few studies that evaluate its use in deep learning. We address this gap in the literature by applying thresholding to a severely class-imbalanced Medicare fraud data set with two- and four-hidden-layer neural networks. Data sampling methods are used to create 15 training distributions across a range of class imbalance levels, and optimal classification thresholds are calculated for each distribution using a validation set. Thresholds are then evaluated by scoring on a test set and comparing results to those achieved with the default threshold of 0.5. Statistical results show that the default threshold is never optimal when networks are trained with imbalanced data. A strong linear relationship between the minority class size and the optimal decision threshold further supports using non-default thresholds to classify imbalanced data. Finally, class-wise performance results show that even when the default threshold performs well, the optimal threshold produces more desirable results that favor the positive class and maximize the TPR.

There are several opportunities for future work available. A similar thresholding procedure can be evaluated across a range of domains to show that thresholding is equally effective with alternative data types. Other methods for selecting optimal thresholds, such as dividing by class prior probabilities, should be compared to the validation-based method used in this study. Finally, the effects of various hyperparameters should be explored to determine how they change the optimal decision threshold.

Acknowledgments

The authors would like to thank the reviewers in the Data Mining and Machine Learning Laboratory at Florida Atlantic University.

References

[1] A. Jain, S. Ratnoo, and D. Kumar, "Addressing class imbalance problem in medical diagnosis: A genetic algorithm approach," in *2017 International Conference on Information, Communication, Instrumentation and Control (ICICIC)*, Aug 2017, pp. 1–8.

[2] R. A. Bauder and T. M. Khoshgoftaar, "Medicare fraud detection using random forest with class imbalanced big data," in *2018 IEEE International Conference on Information Reuse and Integration (IRI)*, July 2018, pp. 80–87.

[3] R. B. Rao, S. Krishnan, and R. S. Niculescu, "Data mining for improved cardiac care," *SIGKDD Explor. Newsl.*, vol. 8, no. 1, pp. 3–10, Jun. 2006. [Online]. Available: http://doi.acm.org/10.1145/1147234.1147236

[4] W. Wei, J. Li, L. Cao, Y. Ou, and J. Chen, "Effective detection of sophisticated online banking fraud on extremely imbalanced data," *World Wide Web*, vol. 16, no. 4, pp. 449–475, Jul 2013. [Online]. Available: https://doi.org/10.1007/s11280-012-0178-0

[5] D. A. Cieslak, N. V. Chawla, and A. Striegel, "Combating imbalance in network intrusion datasets," in *2006 IEEE International Conference on Granular Computing*, May 2006, pp. 732–737.

[6] M. Kubat, R. C. Holte, and S. Matwin, "Machine learning for the detection of oil spills in satellite radar images," *Machine Learning*, vol. 30, no. 2, pp. 195–215, Feb 1998. [Online]. Available: https://doi.org/10.1023/A:1007452223027

[7] N. V. Chawla, N. Japkowicz, and A. Kotcz, "Editorial: Special issue on learning from imbalanced data sets," *SIGKDD Explor. Newsl.*, vol. 6, no. 1, pp. 1–6, Jun. 2004. [Online]. Available: http://doi.acm.org/10.1145/1007730.1007733

[8] R. A. Bauder, T. M. Khoshgoftaar, and T. Hasanin, "An empirical study on class rarity in big data," in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec 2018, pp. 785–790.

[9] J. L. Leevy, T. M. Khoshgoftaar, R. A. Bauder, and N. Seliya, "A survey on addressing high-class imbalance in big data," *Journal of Big Data*, vol. 5, no. 1, p. 42, Nov 2018. [Online]. Available: https://doi.org/10.1186/s40537-018-0151-6

[10] C. Seiffert, T. M. Khoshgoftaar, J. V. Hulse, and A. Napolitano, "Mining data with rare events: A case study," in *19th IEEE International Conference on Tools with Artificial Intelligence(ICTAI 2007)*, vol. 2, Oct 2007, pp. 132–139.

[11] G. Weiss and F. Provost, "The effect of class distribution on classifier learning: An empirical study," *Tech Rep*, 09 2001.

[12] P. Branco, L. Torgo, and R. P. Ribeiro, "A survey of predictive modeling on imbalanced domains," *ACM Comput. Surv.*, vol. 49, no. 2, pp. 31:1–31:50, Aug. 2016. [Online]. Available: http://doi.acm.org/10.1145/2907070

[13] A. Ali, S. M. Shamsuddin, and A. Ralescu, "Classification with class imbalance problem: A review," in *SOCO 2015*, vol. 7, 01 2015, pp. 176–204.

[14] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent Data Analysis*, pp. 429–449, 2002.

[15] J. Van Hulse, T. M. Khoshgoftaar, and A. Napolitano, "Experimental perspectives on learning from imbalanced data," in *Proceedings of the 24th International Conference on Machine Learning*, ser. ICML '07. New York, NY, USA: ACM, 2007, pp. 935–942.

[16] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *Journal of Big Data*, vol. 6, no. 1, p. 27, Mar 2019. [Online]. Available: https://doi.org/10.1186/s40537-019-0192-5

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Neural Information Processing Systems*, vol. 25, 01 2012.

[18] D. Masko and P. Hensman, "The impact of imbalanced training data for convolutional neural networks," 2015, kTH, School of Computer Science and Communication (CSC).

[19] H. Lee, M. Park, and J. Kim, "Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning," in *2016 IEEE International Conference on Image Processing (ICIP)*, Sept 2016, pp. 3713–3717.

[20] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, pp. 249 – 259, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608018302107

[21] H. Wang, Z. Cui, Y. Chen, M. Avidan, A. B. Abdallah, and A. Kronzer, "Predicting hospital readmission via cost-sensitive deep learning," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, pp. 1–1, 2018.

[22] S. H. Khan, M. Hayat, M. Bennamoun, F. A. Sohel, and R. Togneri, "Cost-sensitive learning of deep feature representations from imbalanced data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, pp. 3573–3587, 2018.

[23] T.-Y. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2999–3007, 2017.

[24] C. Huang, Y. Li, C. C. Loy, and X. Tang, "Learning deep representation for imbalanced classification," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 5375–5384.

[25] S. Ando and C. Y. Huang, "Deep over-sampling framework for classifying imbalanced data," in *Machine Learning and Knowledge Discovery in Databases*, M. Ceci, J. Hollmén, L. Todorovski, C. Vens, and S. Džeroski, Eds. Cham: Springer International Publishing, 2017, pp. 770–785.

[26] Q. Dong, S. Gong, and X. Zhu, "Imbalanced deep learning by minority class incremental rectification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2018.

[27] Q. Zou, S. Xie, Z. Lin, M. Wu, and Y. Ju, "Finding the best classification threshold in imbalanced classification," *Big Data Research*, vol. 5, pp. 2–8, 2016.

[28] V. Ganganwar, "An overview of classification algorithms for imbalanced datasets," *International Journal of Emerging Technology and Advanced Engineering*, vol. 2, pp. 42–47, 01 2012.

[29] A. D. Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi, "Calibrating probability with undersampling for unbalanced classification," in *2015 IEEE Symposium Series on Computational Intelligence*, Dec 2015, pp. 159–166.

[30] J. J Chen, C.-A. Tsai, H. Moon, H. Ahn, J. J Young, and C.-H. Chen, "Decision threshold adjustment in class prediction," *SAR and QSAR in environmental research*, vol. 17, pp. 337–52, 07 2006.

[31] R. P. Lippmann, "Neural networks, bayesian a posteriori probabilities, and pattern classification," in *From Statistics to Neural Networks*, V. Cherkassky, J. H. Friedman, and H. Wechsler, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 83–104.

[32] F. Provost, "Learning with imbalanced data sets 101," *Papers from the AAAI workshop. Technical report WS-00-05*, pp. 1–4, 01 2000.

[33] M. Herland, T. M. Khoshgoftaar, and R. A. Bauder, "Big data fraud detection using multiple medicare data sources," *Journal of Big Data*, vol. 5, no. 1, p. 29, Sep 2018. [Online]. Available: https://doi.org/10.1186/s40537-018-0138-3

[34] R. Anand, K. G. Mehrotra, C. K. Mohan, and S. Ranka, "An improved algorithm for neural network classification of imbalanced training sets," *IEEE Transactions on Neural Networks*, vol. 4, no. 6, pp. 962–969, Nov 1993.

[35] S. Wang, W. Liu, J. Wu, L. Cao, Q. Meng, and P. J. Kennedy, "Training deep neural networks on imbalanced data sets," in *2016 International Joint Conference on Neural Networks (IJCNN)*, July 2016, pp. 4368–4374.

[36] Q. Chen, J. Huang, R. Feris, L. M. Brown, J. Dong, and S. Yan, "Deep domain adaptation for describing people based on fine-grained clothing attributes," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 5315–5324.

[37] Y. LeCun and C. Cortes. (2010) MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/. Accessed: 2018-11-15. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[38] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research)." [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html

[39] Office of Inspector General. (2019) Leie downloadable databases. [Online]. Available: https://oig.hhs.gov/exclusions/exclusions_list.asp

[40] Centers For Medicare & Medicaid Services. (2018) Medicare provider utilization and payment data: Physician and other supplier. [Online]. Available: https://www.cms.gov/research-statistics-data-and-systems/statistics-trends-and-reports/medicare-provider-charge-data/physician-and-other-supplier.html

[41] F. Provost and T. Fawcett, "Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions," *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, vol. 43-48, 12 1999.

[42] F. Chollet *et al.*, "Keras," https://keras.io, 2015. [Online]. Available: https://keras.io

[43] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. [Online]. Available: http://tensorflow.org/

[44] Centers for Medicare & Medicaid Services. (2019) National provider identifier standard (npi). [Online]. Available: https://www.cms.gov/Regulations-and-Guidance/Administrative-Simplification/NationalProvIdentStand/

[45] Centers For Medicare & Medicaid Services. (2018) Hcpcs general information. [Online]. Available: https://www.cms.gov/Medicare/Coding/MedHCPCSGenInfo/index.html

[46] R. A. Bauder and T. M. Khoshgoftaar, "A novel method for fraudulent medicare claims detection from expected payment deviations (application paper)," in *2016 IEEE 17th International Conference on Information Reuse and Integration (IRI)*, July 2016, pp. 11–19.

[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2015.

[48] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010.

[49] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, pp. 448–456. [Online]. Available: http://dl.acm.org/citation.cfm?id=3045118.3045167

[50] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2627435.2670313

[51] T. Hasanin, T. M. Khoshgoftaar, J. L. Leevy, and N. Seliya, "Examining characteristics of predictive models with imbalanced big data," *Journal of Big Data*, vol. 6, no. 1, p. 69, Jul 2019. [Online]. Available: https://doi.org/10.1186/s40537-019-0231-2

[52] T. M. Khoshgoftaar, C. Seiffert, J. V. Hulse, A. Napolitano, and A. Folleco, "Learning with limited minority class data," in *Sixth International Conference on Machine Learning and Applications (ICMLA 2007)*, Dec 2007, pp. 348–353.

[53] B. Zdaniuk, *Ordinary Least-Squares (OLS) Model*. Dordrecht: Springer Netherlands, 2014, pp. 4515–4517.