Optimizing Imbalanced Data Classification The OWA-ELM Approach

K.Triveni1, Mr.N.Sravankumar2
1 PG Student, Vemu Institute of Technology, P. Kothakota
2 Assistant Professor, Vemu Institute of Technology, P. Kothakota, AP

ABSTRACT

In the realm of machine learning, the Extreme Learning Machine (ELM) excels in classification and regression but falters with imbalanced data. To address this, we introduce OWA-ELM, an enhanced ELM algorithm with output weight adjustment. OWA-ELM strategically adjusts connection weights to favor minority classes, improving their classification accuracy without compromising majority class performance. Furthermore, we integrate OWA-ELM with the deep learning SMOTE algorithm to automatically synthesize similar data, addressing data imbalance more effectively. Our extended OWA-ELM with SMOTE outperforms other algorithms, achieving higher FSCOREs and demonstrating superior classification performance on imbalanced datasets.

Keywords: Imbalanced Data, OWA-ELM

INTRODUCTION

In machine learning classification applications, imbalanced datasets pose significant challenges across diverse fields such as computer vision, medical science, and industry. Traditional classifiers often exhibit bias towards majority classes, overlooking minority classes due to their inherent assumption of balanced sample distribution. Extreme Learning Machine (ELM), a rapid and efficient neural network, despite its widespread use in various applications, struggles with imbalanced data, leading to suboptimal classification results favoring majority classes. Several algorithms like WELM, LW-ELM, and CCR-ELM have been proposed to enhance ELM's performance on imbalanced datasets by introducing weighted schemes, regularization parameters, and
cost-sensitive approaches. However, these approaches either lack efficiency, introduce complexity, or require extensive parameter tuning. Addressing these limitations, our project aims to introduce a novel algorithm that leverages the strengths of ELM while effectively handling imbalanced data for improved classification accuracy.

**LITERATURE SURVEY**

**S. Dhar and V. Cherkasskyet al**

As The authors introduce an extension to the Universum-support vector machine (U-SVM), a learning methodology tailored for analyzing high-dimensional data with a high feature-to-sample ratio. While U-SVM has shown promise in handling sparse high-dimensional datasets, previous studies primarily focused on balanced datasets with uniform misclassification costs. This paper innovatively extends U-SVM to accommodate varying misclassification costs, enhancing its applicability in real-world scenarios with imbalanced data. Practical conditions for optimizing this cost-sensitive U-SVM are delineated, supported by empirical comparisons demonstrating the efficacy of the proposed approach in diverse learning settings.

**Z. Li, K. Kamnitsas, and B. Glockeret al**

As The authors address the issue of class imbalance affecting the accuracy of predictive models, particularly in image segmentation where neural networks may overfit to under-represented foreground samples. They identify a bias in logit activations at test time, causing a shift across the decision boundary for under-represented classes, leading to systematic under-segmentation of small structures. To mitigate this, the study introduces asymmetric variants of popular loss functions and regularization techniques, including large margin loss, focal loss, adversarial training, mixup, and data augmentation. Empirical evaluations across various segmentation tasks confirm that these modifications significantly enhance segmentation accuracy compared to conventional methods.

**Q. Liu, G. Ma, and C. Chenge t al**

As the study addresses the challenge of imbalanced class distributions in fault diagnosis of rotating machinery, which can compromise diagnostic accuracy using conventional machine learning algorithms due to uniform cost weights across fault classes. The research introduces a novel diagnostic framework leveraging adversarial neural networks (GAN) and multi-sensor data fusion to generate synthetic data for compensation. Two distinct practice modes, Pre-fusion GAN and Post-fusion GAN, are designed based on data fusion positioning logic. Without requiring data pre-processing, the framework generates synthetic data challenging the discriminator, effectively compensating for the minority class. Validated on two imbalanced datasets from rotating machinery, the proposed approach demonstrates superior performance compared to existing methods, showcasing its efficacy in imbalanced fault diagnosis.

**PROBLEM STATEMENT:**

Extreme Learning Machine algorithm is deep learning algorithm which will filter data by using multiple neurons and this algorithm is using everywhere for classification such as Face
recognition, health disease prediction and many more. This algorithm performance may go down while handling imbalance data (data which contains more instances in one CLASS and fewer instances in another CLASS which raise major and minor problem).

**PROPOSED METHOD:**

To overcome from above problem and to handle imbalance data with ELM author of this paper has enhance ELM by adding lesser BETA value. In ELM algorithm, beta is required as small as possible to avoid the structural risk of the model and resulting in weak generalization ability. However, if only slightly increase each element in the column vector that determines the output value of the minority output neuron in the output weight matrix Beta, it may not cause too much structural risk, but helps to improve the classification accuracy of minority class samples. So by enhancing beta value ELM performance can be increase and name as ‘output weight adjustment ELM’ (OWA-ELM).

In propose paper author has evaluate performance of OWA-ELM in terms of accuracy, precision, recall and FSCORE. Author has compare OWA-ELM with traditional ELM algorithm. To measure performance author has used 22 datasets and its bit difficult to use all those datasets so we have use Cleveland, PIMA and Wisconsin dataset.
DATASETS:

Displaying PIMA dataset

<table>
<thead>
<tr>
<th>Preg</th>
<th>Gluc</th>
<th>BP</th>
<th>SkinT</th>
<th>Insu</th>
<th>BMI</th>
<th>DP</th>
<th>Age</th>
<th>Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>6</td>
<td>148</td>
<td>72</td>
<td>35</td>
<td>0</td>
<td>33.6</td>
<td>0.627</td>
<td>59</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>85</td>
<td>66</td>
<td>29</td>
<td>0</td>
<td>26.6</td>
<td>0.351</td>
<td>31</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>133</td>
<td>64</td>
<td>0</td>
<td>0</td>
<td>23.3</td>
<td>0.672</td>
<td>32</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>89</td>
<td>66</td>
<td>23</td>
<td>94</td>
<td>28.1</td>
<td>0.167</td>
<td>21</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>137</td>
<td>40</td>
<td>35</td>
<td>188</td>
<td>43.1</td>
<td>2.288</td>
<td>33</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>703</td>
<td>10</td>
<td>101</td>
<td>76</td>
<td>48</td>
<td>180</td>
<td>32.9</td>
<td>0.171</td>
<td>63</td>
</tr>
<tr>
<td>704</td>
<td>2</td>
<td>122</td>
<td>70</td>
<td>27</td>
<td>0</td>
<td>36.8</td>
<td>0.340</td>
<td>27</td>
</tr>
<tr>
<td>705</td>
<td>5</td>
<td>121</td>
<td>72</td>
<td>23</td>
<td>112</td>
<td>26.2</td>
<td>0.245</td>
<td>30</td>
</tr>
<tr>
<td>706</td>
<td>1</td>
<td>126</td>
<td>60</td>
<td>0</td>
<td>510</td>
<td>30.1</td>
<td>0.349</td>
<td>47</td>
</tr>
<tr>
<td>707</td>
<td>1</td>
<td>93</td>
<td>70</td>
<td>31</td>
<td>0</td>
<td>30.4</td>
<td>0.315</td>
<td>23</td>
</tr>
</tbody>
</table>

768 rows x 9 columns

Displaying Wisconsin dataset

<table>
<thead>
<tr>
<th>Id</th>
<th>Diagnosis</th>
<th>radius_mean</th>
<th>texture_mean</th>
<th>perimeter_mean</th>
<th>area_mean</th>
<th>smoothness_mean</th>
<th>compactness_mean</th>
<th>concave points_mean</th>
<th>concave points_mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>M</td>
<td>17.09</td>
<td>10.38</td>
<td>122.80</td>
<td>1805.1</td>
<td>0.16840</td>
<td>0.27760</td>
<td>0.30016</td>
<td>0.14710</td>
</tr>
<tr>
<td>1</td>
<td>M</td>
<td>20.57</td>
<td>17.77</td>
<td>132.00</td>
<td>1320.0</td>
<td>0.08474</td>
<td>0.07864</td>
<td>0.08560</td>
<td>0.07864</td>
</tr>
<tr>
<td>2</td>
<td>M</td>
<td>19.66</td>
<td>21.26</td>
<td>130.00</td>
<td>1203.0</td>
<td>0.10596</td>
<td>0.15960</td>
<td>0.19740</td>
<td>0.12790</td>
</tr>
<tr>
<td>3</td>
<td>M</td>
<td>11.42</td>
<td>20.20</td>
<td>77.56</td>
<td>195.1</td>
<td>0.14250</td>
<td>0.25300</td>
<td>0.23500</td>
<td>0.10020</td>
</tr>
<tr>
<td>4</td>
<td>M</td>
<td>20.20</td>
<td>14.84</td>
<td>135.10</td>
<td>1297.0</td>
<td>0.10030</td>
<td>0.13280</td>
<td>0.10800</td>
<td>0.10430</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>564</td>
<td>M</td>
<td>21.56</td>
<td>22.39</td>
<td>142.00</td>
<td>1479.0</td>
<td>0.11100</td>
<td>0.11590</td>
<td>0.24360</td>
<td>0.13090</td>
</tr>
<tr>
<td>565</td>
<td>M</td>
<td>26.13</td>
<td>28.25</td>
<td>131.20</td>
<td>1251.0</td>
<td>0.09780</td>
<td>0.10340</td>
<td>0.14400</td>
<td>0.09791</td>
</tr>
<tr>
<td>566</td>
<td>M</td>
<td>16.00</td>
<td>28.08</td>
<td>108.30</td>
<td>878.1</td>
<td>0.04525</td>
<td>0.10230</td>
<td>0.05251</td>
<td>0.05302</td>
</tr>
<tr>
<td>567</td>
<td>M</td>
<td>20.66</td>
<td>25.53</td>
<td>140.10</td>
<td>1265.0</td>
<td>0.11790</td>
<td>0.27705</td>
<td>0.35140</td>
<td>0.15260</td>
</tr>
<tr>
<td>568</td>
<td>B</td>
<td>7.76</td>
<td>24.54</td>
<td>47.92</td>
<td>181.0</td>
<td>0.05253</td>
<td>0.04362</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
</tbody>
</table>

569 rows x 33 columns
Displaying Cleveland dataset

**METHODOLOGY:**

1.1 Importing Required Python Packages:

The initial step involves importing essential libraries such as pandas, numpy, and scikit-learn. These libraries play a pivotal role in data manipulation, preprocessing, model evaluation, and visualization, facilitating streamlined development and analysis.

1.2 Reading and Displaying Datasets:

This project incorporates three distinct datasets: PIMA, Wisconsin, and Cleveland. These datasets encompass vital attributes including glucose levels, blood pressure, and age, which are instrumental in heart disease prediction. Data is loaded into pandas DataFrames, providing a comprehensive view of its structure and content.

1.3 Handling Missing Values:

Addressing missing values is crucial to maintain data integrity. In this project, missing values within the datasets are replaced with zeros, ensuring uniformity and facilitating subsequent processing steps.

1.4 Encoding Categorical Variables:

To facilitate model training, categorical variables within the datasets are encoded into numeric format. Label encoding techniques are employed to transform non-numeric data into numerical representations, ensuring compatibility with machine learning algorithms.

1.5 Feature Scaling:

To optimize model performance and prevent dominance by features with larger scales, Min-Max scaling is applied. This normalization technique ensures that all features contribute proportionally to model training, enhancing predictive accuracy.
1.6 Algorithm Selection:

Two distinct algorithms are implemented in this project: Existing Extreme Learning Machine (ELM) and Proposed OWA-ELM (Ordered Weighted Averaging Extreme Learning Machine). ELM is characterized by a single hidden layer neural network, while OWA-ELM incorporates an additional layer to address imbalanced data. Additionally, SMOTE (Synthetic Minority Over-sampling Technique) is applied to augment performance on imbalanced datasets.

1.7 Model Training and Evaluation:

The dataset is partitioned into training and testing subsets using a conventional split ratio. Both ELM and OWA-ELM models are trained on the training data and subsequently evaluated using a range of metrics including accuracy, precision, recall, and F1-score. Confusion matrices are generated to visualize the performance of these models in predicting diverse classes, with extension techniques like SMOTE employed to enhance model performance on imbalanced data.

1.8 Comparative Analysis:

A comprehensive comparative analysis is conducted to assess the performance of each algorithm across different datasets. Performance metrics, including accuracy, precision, recall, and F1-score, are calculated and juxtaposed to evaluate the efficacy of each algorithm in predicting heart disease.

1.9 Visualization:

To facilitate intuitive understanding and interpretation, bar graphs are plotted to visually depict the performance of each algorithm across various datasets. These graphical representations serve as valuable tools for assessing the comparative effectiveness and performance of the implemented algorithms.

1.10 Test Data Preparation:

Test data is subjected to similar preprocessing steps as the training data to ensure compatibility with the trained models, maintaining consistency and facilitating accurate predictions.

1.11 Prediction:

Utilizing the trained models, predictions regarding the likelihood of heart disease for the test data samples are made. Predictions are based on the highest probability class inferred from the models, providing valuable insights into potential heart disease risks.

**EVOLUTION:**

**Precision:**

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \times 100$$

Code: ```python
precision = precision_score(testY, predict, average='macro') * 100
```

**Recall (Sensitivity):**

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \times 100$$

Code: ```python
recall = recall_score(testY, predict, average='macro') * 100
```

**F1 Score:**

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Code: ```python
f1 = f1_score(testY, predict, average='macro') * 100
```
Accuracy:

Formula: \[ \text{Accuracy} = \frac{\text{Correct Predictions}}{\text{Total Predictions}} \]

Code: `accuracy = accuracy_score(testY, predict) * 100`
RESULTS:

In above screen now train PIMA dataset with existing ELM, propose OWA-ELM and extension SMOTE-OWA-ELM and with existing ELM we got 72% accuracy and in confusion matrix graph x-axis represents PREDICTED labels and y-axis represents TRUE labels where yellow top blue colour boxes represents correct prediction count and other boxes represents incorrect prediction count. Propose OWA-ELM we got 78% accuracy and with extension we got 80% accuracy.

Training existing ELM with WISCONSIN dataset and we got 86% accuracy. With propose OWA-ELM on Wisconsin dataset we got 97.36% accuracy and with extension smote we got 97.41% accuracy.
Training ELM with Cleveland dataset and got 76.66% accuracy. Propose OWA-ELM got 88% accuracy and extension got 89% accuracy.

In above screen we can see accuracy graph of existing ELM, propose OWA-ELM and Extension SMOTE where x-axis represents algorithm names and y-axis represents accuracy and in all algorithms extension SMOTE got high accuracy.

In above screen we can see precision graph of existing ELM, propose OWA-ELM and Extension SMOTE where x-axis represents algorithm names and y-axis represents Precision and in all algorithms extension SMOTE got high Precision.
In above screen we can see recall graph of existing ELM, propose OWA-ELM and Extension SMOTE where x-axis represents algorithm names and y-axis represents recall and in all algorithms extension SMOTE got high recall.

In above screen we can see FSCORE graph of existing ELM, propose OWA-ELM and Extension SMOTE where x-axis represents algorithm names and y-axis represents FSCORE and in all algorithms extension SMOTE got high FSCORE.

<table>
<thead>
<tr>
<th>Dataset &amp; Algorithm Name</th>
<th>Precision</th>
<th>Recall</th>
<th>FScore</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ELM Pima Dataset</td>
<td>66.236559</td>
<td>62.010340</td>
<td>62.770563</td>
</tr>
<tr>
<td>1</td>
<td>OWA-ELM Pima Dataset</td>
<td>74.851190</td>
<td>73.245178</td>
<td>73.926428</td>
</tr>
<tr>
<td>2</td>
<td>Extension SMOTE-OWA-ELM Pima Dataset</td>
<td>80.303639</td>
<td>80.279898</td>
<td>80.276035</td>
</tr>
<tr>
<td>3</td>
<td>ELM Wisconsin Dataset</td>
<td>86.363636</td>
<td>86.538042</td>
<td>86.465165</td>
</tr>
<tr>
<td>4</td>
<td>OWA-ELM Wisconsin Dataset</td>
<td>97.442455</td>
<td>97.126072</td>
<td>97.275986</td>
</tr>
<tr>
<td>5</td>
<td>Extension SMOTE-OWA-ELM Wisconsin Dataset</td>
<td>97.427892</td>
<td>97.413793</td>
<td>97.413793</td>
</tr>
<tr>
<td>6</td>
<td>ELM Cleveland Dataset</td>
<td>77.678571</td>
<td>79.665072</td>
<td>76.430976</td>
</tr>
<tr>
<td>7</td>
<td>OWA-ELM Cleveland Dataset</td>
<td>87.142857</td>
<td>88.875598</td>
<td>87.758671</td>
</tr>
<tr>
<td>8</td>
<td>Extension SMOTE-OWA-ELM Cleveland Dataset</td>
<td>89.930556</td>
<td>89.855072</td>
<td>89.850276</td>
</tr>
</tbody>
</table>

In above we can see all algorithms performance in tabular format for all datasets.

Prediction:
In above screen we are performing prediction on test data and in square bracket we can see test data and after arrow symbol \( \Rightarrow \) we can see classification output as ‘No heart disease detected’ or ‘Heart disease detected’

**CONCLUSION**

This project focuses on enhancing the Extreme Learning Machine (ELM) algorithm, crucial for various classifications like face recognition and health disease prediction. ELM's performance falters with imbalanced data, posing major and minor problems. To address this, the author enhances ELM by adjusting the beta value, resulting in 'Output Weight Adjustment ELM' (OWA-ELM). By slightly increasing beta, the classification accuracy of minority class samples improves without significant structural risk. OWA-ELM's performance is evaluated against traditional ELM using datasets like Cleveland, PIMA, and Wisconsin. As an extension, OWA-ELM is combined with the deep learning SMOTE algorithm, boosting FSCORE significantly compared to other algorithms, providing a robust solution to data imbalance.

**REFERENCES:**


