Lightweight Deep Neural Network for Imbalanced Binary Dataset Classification with Dice-Loss in NLP

Yunhu Kim
Purdue University
West Lafayette, Indiana, USA

ABSTRACT

Many text dataset is skewed. Traditional models often struggle with imbalanced data, a common issue in applications like cyberbully detection and spam classification. We introduce a lightweight deep neural network[6] optimized with a differentiable approximation of F1 score for binary text classification, handling skewed datasets.

ACM Reference Format:
Yunhu Kim. 2018. Lightweight Deep Neural Network for Imbalanced Binary Dataset Classification with Dice-Loss in NLP. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX)*. ACM, New York, NY, USA, 4 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

The advancement of the Large Language Model (LLM) brought various advantages in text classification across multiple fields including customer service, and spam detection. These models identify, interpret, and classify the textual data.

However, the LLM in real-world applications can be challenging due to the limited computational resources and latency. This is especially true for applications requiring real-time processing or operating within resource-constrained environments, such as mobile devices, IoT devices, and embedded systems. Furthermore, many Natural Language Processing (NLP) problems are imbalanced. In this study we address imbalanced binary classification datasets, such as cyberbullying detection and spam classification. These tasks demand high accuracy and a balanced consideration of majority and minority classes.

We introduce an optimization strategy by utilizing a differentiable approximation of the F1 score in the training process. The F1 score measures the balance between precision and recall, addressing a common shortfall in models optimized for accuracy. We will compare various loss functions including Mean Squared Error[1], Negative Log-Likelihood[9], and differentiable approximation of the F1 score[2]. We will address various problems including cyberbullying detection and spam classification.

2 PREVIOUS WORKS

Many natural language processing (NLP) tasks are imbalanced, and there has been many works to address the problem. Henning et al. [3] surveyed a efficient ways to address imbalanced problem in NLP including Re-sampling, data augmentation, and loss-function

2.1 Re-Sampling

Re-sampling can increase the importance of the minority class in training by changing the label distribution. For instance, random oversampling duplicates the randomly picked minority class instances, while random undersampling removes the randomly selected majority class instances. These methods set a more balanced label distribution, which can improve the performance of machine learning models by giving equal weight to all classes during training.

2.2 Loss-Function

One of the widely used method is adjusting weight of the loss function. Adjusting the loss function can address the imbalanced problem. For instance, Weighted Cross Entropy assigns weights to each class based on the distribution of the true labels. It can increase the importance of the minority class in training.

3 METHOD

3.1 loss function for imbalanced dataset

The loss function in this study focus on increasing the F1 score. The F1 score is a measure that balances between precision and recall. It is calculated as the mean of these two metrics, where precision represents the model's ability to identify only the relevant data points. At the same time, recall measures the model's capacity to identify all relevant instances within the dataset. Thus increasing F1 score of the model can solve the imbalanced data problem.

3.2 Differentiable Approximated F1 Loss

The F1 score is expressed mathematically as:

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{1}$$

Here, TP, FP, and FN represent true positives, false positives, and false negatives, respectively.

The **Sørensen–Dice coefficient** is a statistic used to compare the similarity of two samples.[8]

The Sørensen-Dice coefficient when applied to boolean variable, equation can be written as:

$$\text{Dice} = \frac{2TP}{2TP + FP + FN} = F_1 \tag{2}$$

Therefore the Sørensen-Dice coefficient and F_1 score is equal. Sørensen's original formula was intended to be applied to discrete data. Given two sets, X and Y, it is defined as

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

$$\text{Dice}(X, Y) = \frac{2|X \cap Y|}{|X| + |Y|} \tag{3}$$

where |X| and |Y| are the cardinalities of the two sets (i.e. the number of elements in each set). The Sørensen index equals twice the number of elements common to both sets divided by the sum of the number of elements in each set.

From Set Theory to Predictive Modeling

In predictive modeling, we often deal with predictions ($\hat{y}$) and actual outcomes (y) represented as binary vectors. Here, each element of y and $\hat{y}$ indicates the presence (1) or absence (0) of a characteristic.

Intersection as Hadamard Product

The intersection $|X \cap Y|$ of two sets in set theory is analogous to counting true positives in binary classification. This is achieved by the Hadamard product $y \odot \hat{y}$, which performs element-wise multiplication of y and $\hat{y}$:

$$\sum (y \odot \hat{y}) = TP \tag{4}$$

where the TP means True Positive, and the multiplication $y_i \times \hat{y}_i$ equals 1 if both y_i and $\hat{y}_i$ are 1 (true positive), and 0 otherwise.

Dice Coefficient in Predictive Modeling

The Dice coefficient, when adapted for binary data, can be redefined as follows for use as a loss function in model evaluation:

$$L_{\text{Dice}}(y, \hat{y}) = 1 - \frac{2 \sum (y \odot \hat{y})}{\sum y + \sum \hat{y}} \tag{5}$$

This equation employs:

- $2 \sum (y \odot \hat{y})$: twice the sum of the element-wise product, representing twice the number of true positives.
- $\sum y + \sum \hat{y}$: the total number of positives predicted or actually present, akin to the cardinalities in the set-based formula.

APPLICATION OF THE SIGMOID FUNCTION IN PREDICTIVE MODELING

After computing the raw prediction outputs ($\hat{y}$) from a model, such as a neural network or logistic regression, it needs to be transformed for binary classification. This transformation is achieved using the sigmoid function, defined as:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{6}$$

Reason for Using the Sigmoid Function

The sigmoid function is crucial for several reasons:

- **Output Range**: The sigmoid function maps any real-valued number into the (0, 1) range, which is a probability. This is useful in binary classification where the output needs to represent the probability that a given input belongs to the positive class (1).
- **Differentiability**: The sigmoid function is continuously differentiable, which can be used for gradient-based optimization methods used in training models. The smooth gradient of the sigmoid function allows adjustments during backpropagation.

Application in Dice Coefficient Calculation

Once $\hat{y}$ is transformed into $\sigma(\hat{y})$ using the sigmoid function, the predictions can be used to compute the Dice coefficient (or Dice loss) in binary classification tasks. The predicted probabilities ($\hat{y}$) allow for the assessment of model performance based on how well the predicted probabilities align with the actual binary labels (y).

$$L_{\text{Dice}}(y, \hat{y}) = 1 - \frac{2 \sum (y \odot \sigma(\hat{y}))}{\sum y + \sum \sigma(\hat{y})} \tag{7}$$

In this context, the element-wise product $y \odot \hat{y}$ is influenced by how probabilities are shaped by the sigmoid function, impacting the calculation of true positives and the overall Dice loss.

4 METHOD

We focus on binary classification datasets to evaluate the lightweight deep neural network optimized with a differentiable F1 score and other loss functions. Specifically, we will use two datasets from Kaggle: the SMS Spam Collection dataset and the Cyberbullying Classification dataset

4.1 Preprocessing

4.1.1 Modifying label for Cyberbulling Classification Dataset. The Cyberbullying Classification dataset is multi-class labeled. If it is not cyberbullying the label is 'not_cyberbulling', else it is labeled with type of the cyberbullying such as 'religion', 'age, and 'gender'. To make binary classification dataset, every cyberbullying label are transformed 1, while 'not_cyberbulling' is 0.

4.2 Word Embedding

For the word embedding, we will use GloVe(Global Vectors for Word Representation) embedding [7] to obtain the vocabulary. GloVe with low dimensions will be enough to manage the dataset for the lightweight neural network.

4.3 Models Under Consideration

We will use a 3-layer Deep Averaging Network (DAN)[5]. DAN is great for handling textual data with a low model complexity. This fits for the lightweight neural network.

4.4 Loss Functions

To optimize these models, we will experiment with three distinct loss functions. These are:

(1) Weighted Binary Cross Entropy (BCE) [4], which adjusts the loss contribution from each class to handle class imbalance.
(2) Mean Squared Error (MSE), traditionally used for regression but adapted here for classification to evaluate its impact on model stability and convergence.
(3) Dice Loss, which is a differentiable approximation of the F1 score [2], which directly integrates the F1 metric into the training process, potentially improving performance on imbalanced datasets.

4.5 Model Selection Process

We will divide our dataset into training, validation, and testing sets. We will tune the hyperparameters of the models based on the validation set

4.6 Evaluation Metrics

We will evaluate each model with the following metrics: Accuracy, and the F1 score. Selected metrics will measure the accuracy and how the model handles the imbalanced dataset.

5 RESULT

5.1 Learning Curve

Each figure represent learning curve for two dataset: SMS Spam dataset and Cyberbilling dataset respectively

Figure 1 and Figure 2 show that the model achieves high accuracy quite rapidly within the first few epochs, which suggests a fast learning rate. The training and validation accuracy converge closely, with the validation accuracy slightly lower than the training accuracy throughout the training process. This indicates that the model generalizes well to the validation set without significant overfitting. However, since the validation accuracy does not surpass the training accuracy at any point, there may still be room for improvement in model generalization.

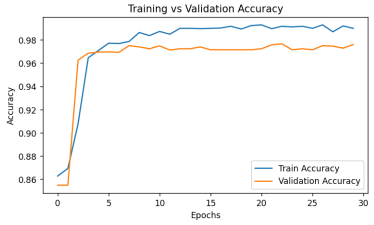

Figure 1: Training and Validation accuracy of DAN with Dice loss on SMS Spam Dataset

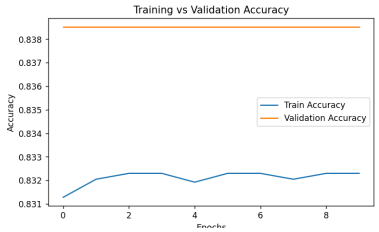

Figure 2: Training and Validation accuracy of DAN with Dice loss on Cyberbulling Dataset

Figure 3 and Figure 4 shows the f1 score comparison between the training and validation dataset. The F1 score, which measures accuracy and considers both the precision and the recall, also shows a similar pattern to accuracy. The convergence of training and validation F1 scores suggests that the model is consistent in its predictions across both datasets.

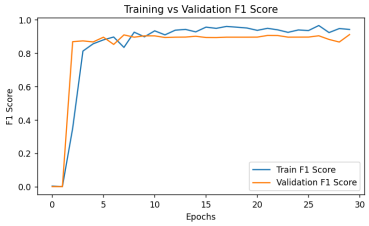

Figure 3: Training and Validation f1 score of DAN with Dice loss on SMS Spam Dataset

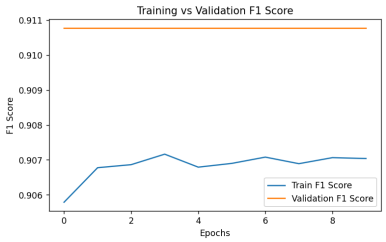

Figure 4: Training and Validation f1 score of DAN with Dice loss on Cyberbulling Dataset

Figure 5 and Figure 6 shows the loss learning plot for the training and validation dataset. The loss for training and validation decreases converges quickly, which indicates effective learning. After the initial epochs, the loss remains relatively flat, suggesting that the model may have reached its capacity for learning from the data provided. The closeness of the training and validation loss lines shows good generalization.

Overall, The model shows excellent performance with high accuracy and F1 scores and low loss on both the training and validation datasets. The rapid convergence and stable performance across the metrics indicate that the model's capacity is well-suited to the complexity of the task. Nevertheless, the fact that the validation scores do not surpass the training scores suggests that the model could be improved with further tuning, data augmentation, or experimentation with different architectures or regularization techniques.

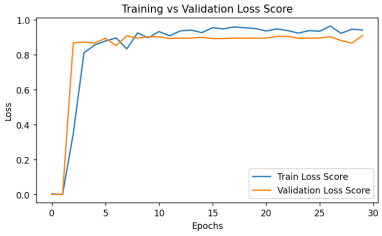

Figure 5: Training and Validation loss of DAN with Dice loss on SMS Spam Dataset

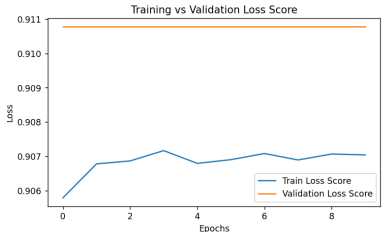

Figure 6: Training and Validation loss of DAN with Dice loss on Cyberbulling Dataset

5.2 Evaluation

My experiment is designed to answer the following question: How much does our algorithm works compared to existing algorithm

As mentioend in **Section 4.3** The table 1 shows the result for F1, Weighted Binary Cross Entropy, and Mean Squared Error loss using lightweight Deep Averaging Neural network.

Dataset	Loss Function	Accuracy	F1 Score
SMS Spam	Dice	0.9677	0.8676
	Weighted BCE	0.9740	0.8945
	MSE	0.9624	0.8409
Cyberbullying	Dice	0.8326	0.9086
	Weighted BCE	0.8703	0.9259
	MSE	0.8556	0.9125

Table 1: Classification Results on SMS Spam and Cyberbullying Datasets

Weighted Binary Cross Entropy tends to perform better than any other algorithm in both dataset. Dice Loss perform better than MSE for SMS Spam dataset, but MSE perform better for Cyberbulling dataset.

5.3 Limitation

The study found that Dice Loss is very sensitive to the learning rate and other hyper parameter that affects learning curve. Changing the learning curve can crush the algorithm and result in predicting only majority class. This takes a lot of time to fine tune the hyper-parameter compared to other loss function.

6 FUTURE WORK

While the study can solve binary-class classification problems for NLP problems, it can not address multi-class classification problems. Future work can expand binary classification dice loss to multi-class classification problems. Also, the F1 score is a sub-optimal score function as it does not consider false negatives during computation. Future work can find an approximate differentiable loss function of the more globally optimal score function such as the Geometric Mean or the AUROC score.

Also I found that the dataset was not completely fit for my experiment as Mean Squared Error Loss without any adjustment could handle imbalanced dataset very well. I believe there need to be better dataset to compute the performance of the Dice loss for imbalanced dataset.

7 WHAT I LEARNED

This project was one of the hardest, but also most fun task I have ever done. I worked alone, which gave me a lot of tasks to do including calculating f1 score to dice loss, planning, and setting up the experiment. But all those tasks were very fun.

Through the process of this research, I learned better understanding of language model, especially for lightweight neural network. It was kind of disappointing that traditional loss function: Weighted Cross Entropy works better for the imbalanced binary classification task even for specifying f1 score, which our research paper addressed. I believe this happened because our model was kind of overfitted while Weighted Cross Entropy method did not, and also f1 score is sub-optimal score function.

I also enhanced my knowledge on neural network loss function and back-propagation. Also beside loss function, I learned how to handle Word Embedding better. Unlike homework, I used the averaged weight for the unknown word, which Author of Glove recommended. I believe there are many other methods that can improve Word Embedding. For example, FastText can handle unknown word, which I believe we can also apply in Glove.

REFERENCES

[1] C. Bishop. *Pattern Recognition and Machine Learning*, volume 16, pages 140–155. 01 2006.
[2] T. FEL. Approximate differentiation of f1 score with the sørensen-dice coefficient. 01 2020.
[3] S. Henning, W. Beluch, A. Fraser, and A. Friedrich. A survey of methods for addressing class imbalance in deep-learning based natural language processing, 2023.
[4] Y. Ho and S. Wookey. The real-world-weight cross-entropy loss function: Modeling the costs of mislabeling. *IEEE Access*, 8:4806–4813, 2020.
[5] M. Iyyer, V. Manjunatha, J. Boyd-Graber, and H. Daumé III. Deep unordered composition rivals syntactic methods for text classification. In C. Zong and M. Strube, editors, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1681–1691, Beijing, China, July 2015. Association for Computational Linguistics.
[6] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436, 2015.
[7] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.
[8] T. Sørensen. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons. *Kongelige Danske Videnskabernes Selskab*, 5:1–34, 1948.
[9] D. Zhu, H. Yao, B. Jiang, and P. Yu. Negative log likelihood ratio loss for deep neural network classification, 2018.