**Data augmentation based on the Wasserstein GAN with gradient penalty to enhance the prediction of genes associated with RNA methylation pathways.**

**Abstract:**

RNA methylation modification influences various processes in the human body, which has gained increasing attention from scholars. Predicting gene associated with RNA methylation pathways can significantly aid biologists in their study of RNA methylation processes. Limited prediction methods have been investigated; their performance is still constrained by the limited sample amount. To improve the prediction of genes associated with RNA methylation pathways, we utilized the dataset collected from Harmonizome and implemented deep learning techniques combined with five well-established machine learning classifiers to oversample the minority data. The experimental results indicate that the synthetic gene samples can complement the original sample distribution effectively, resulting in an improved prediction performance for genes associated with the RNA methylation pathway.

**1.Introduction:**

RNA methylation refers to the chemical process in which methyl groups are selectively added to methyladenine in RNA under the catalytic action of methyltransferase. So far, researchers have discovered over 160 RNA modifications [1] that are extensively distributed across eukaryotes and prokaryotes. Among them, the most prevalent form of RNA modification is methylation. By studying this process, researchers have discovered associations between RNA methylation modifications and various fields, including cancer [2,3], cardiovascular disease [4], embryonic development [5], and cell differentiation [6]. These findings highlight the significance of RNA methylation and suggest their potential implications in diverse biological processes and diseases. Further research in this area may shed more light on the functional roles of RNA methylation and their therapeutic applications [7].

Numerous reports have revealed that specific enzymes or proteins within the RNA methylation pathway exert regulatory effects on distinct functions and biochemical processes in the human body. These studies are founded on the identification of proteins, genes, or specific enzymes associated with RNA methylation. Despite the significant role of the RNA methylation process in various aspects of the human body, researchers have limited knowledge about the pathways involved. This limitation primarily arises from the substantial financial and time investments needed to identify gene functions using experimental wet laboratory methods. Fortunately, due to intensive research of large-scale omics data and artificial intelligence methods in this field, it is now possible to use computational methods that are faster and more cost-effective than wet methods. Researchers can explore the function of target genes through various computational methods.

Several reports have explored the classification of RNA methylation pathways genes [8]. Researchers collect data and conduct experiments using well-established machine learning models on known RNA methylation pathways genes as positive samples. This suggests that ‘dry’ computational methods may be used to identify RNA methylation pathways genes. However, due to the limited number of known samples and the high imbalanced ratio of positive and negative samples, the classifier may incorrectly identify majority class samples as minority class samples. Any misclassification of target genes obtained through computational methods results in increased research costs, including time and funding expenses. Thus, avoiding overfitting and improving the classification method's effectiveness in highly imbalanced and high-dimensional data remains challenging.

Various approaches have emerged to mitigate data imbalance. The most common approach is to oversample the minority class samples. Random oversampling is a widely employed algorithm that duplicates randomly selected minority class samples from the original dataset to balance the sample distribution. However, this algorithm primarily enhances the classifier's weights on the minority class but lacks diversity in generating new samples [9]. Another effective technique, SMOTE [10], generates samples that lie near the minority class in the feature space, introducing diversity through interpolation among neighboring samples. However, SMOTE has limited impact on most classifiers trained on high-dimensional data, particularly in bioinformatics [11] (Blagus, R., & Lusa, L. 2013). This limitation also applies to certain SMOTE variants (e.g., Borderline-SMOTE [12]). Directly applying these algorithms to highly imbalanced datasets may not alleviate overfitting and enhance classification performance.

Deep learning can be used as an alternative approach to address the imbalanced representation of minority samples. Researchers have utilized deep learning techniques to address the challenge of learning from minority samples and generating synthetic samples to supplement the existing dataset. One commonly approach is the use of Generative Adversarial Networks [13] (GANs), which consist of two networks that ‘fight’ against each other: a generator and a discriminator. The generator is responsible for producing synthetic data, while the discriminator evaluates the authenticity of the data. Through an adversarial training process, these 2 networks compete against each other until reaching equilibrium, where the discriminator unable to distinguish real from synthetic data, and the generator cannot improve its sample generation further. This allows the generator to learn the underlying data distribution and generate synthetic samples that can be utilized for downstream tasks. Several studies have confirmed that GANs and their variants can generate synthetic data by learning the distribution of original sample to compensate the original dataset: Ma, Li et al. [14] proposed using deep convolutional generative adversarial network (DC-GAN) to classify white blood cell. Gadermayr et al. [15] use a cycle GAN to translate images with healthy conditions to fake images with affected conditions for augmenting labeled segmentation data. Frid-Adar et al. [16] proposed using GANs to synthesize CT image data and demonstrated that the generated images achieved favorable results on a limited number of datasets, including cysts, metastases, and haemangiomas. Not only the remarkable success of GANs in image synthetic, Xiao, Y et al. [17] utilized WGAN to learn high-dimensional features of cancer gene expression data, which enables GAN to fit the distribution of cancer gene expression data. According to [17], this is the first application of GAN neural networks on a cancer gene expression dataset.

Despite the remarkable success of GANs in balancing datasets, their application in high-dimensional extremely imbalanced gene expression dataset has, surprisingly, not yet been studied, to the best of our knowledge. Thus, we propose a novel approach to address the classification of extreme imbalanced dataset. In this study, we divided the dataset into training set and test set. Using minority class samples from the training set, we trained the Wassestein GAN with gradient penalty (WGAN-GP) [18] as an oversampler and selected the best generator for downstream oversampling using the Classifier Two Sample Test (CTST) [19]. Instead of directly generating synthetic samples to balance the entire dataset, we utilized the selected generator to perform oversampling on multiple data blocks. The balanced data blocks were then used to train the machine model. Finally, we evaluated the effectiveness of our approach by comparing it with several different schemes (SMOTE, Random Oversample, Borderline-SMOTE) on test set using various evaluation metrics. This approach combines the advantages of oversampling and undersampling, compensating the dataset while mitigating information loss caused by undersampling. To the best of our knowledge, this is the first attempt to utilizes a GAN-like algorithm for oversampling in extreme imbalanced datasets.

**2. Method**:

**2.1. Workflow of the study**

Generally, our proposed method consists of four steps. In the first step, we perform feature extraction on the dataset to reduce the dimensionality and train the WGAN-GP using the positive samples in the training set after dividing the dataset into training and test samples. In the second step, all the WGAN-GP weights are evaluated using CTST, and the model weights with the best CTST results are selected for the downstream task oversampling. In the third step, the training set is divided into multiple data blocks, and then each data block is oversampled using the WGAN-GP model selected in the second part to obtain the same number of balanced data blocks. In the fourth step, the machine learning models are trained using the balanced data blocks and the performance of the models is evaluated using the test set. As shown in Fig. 1.

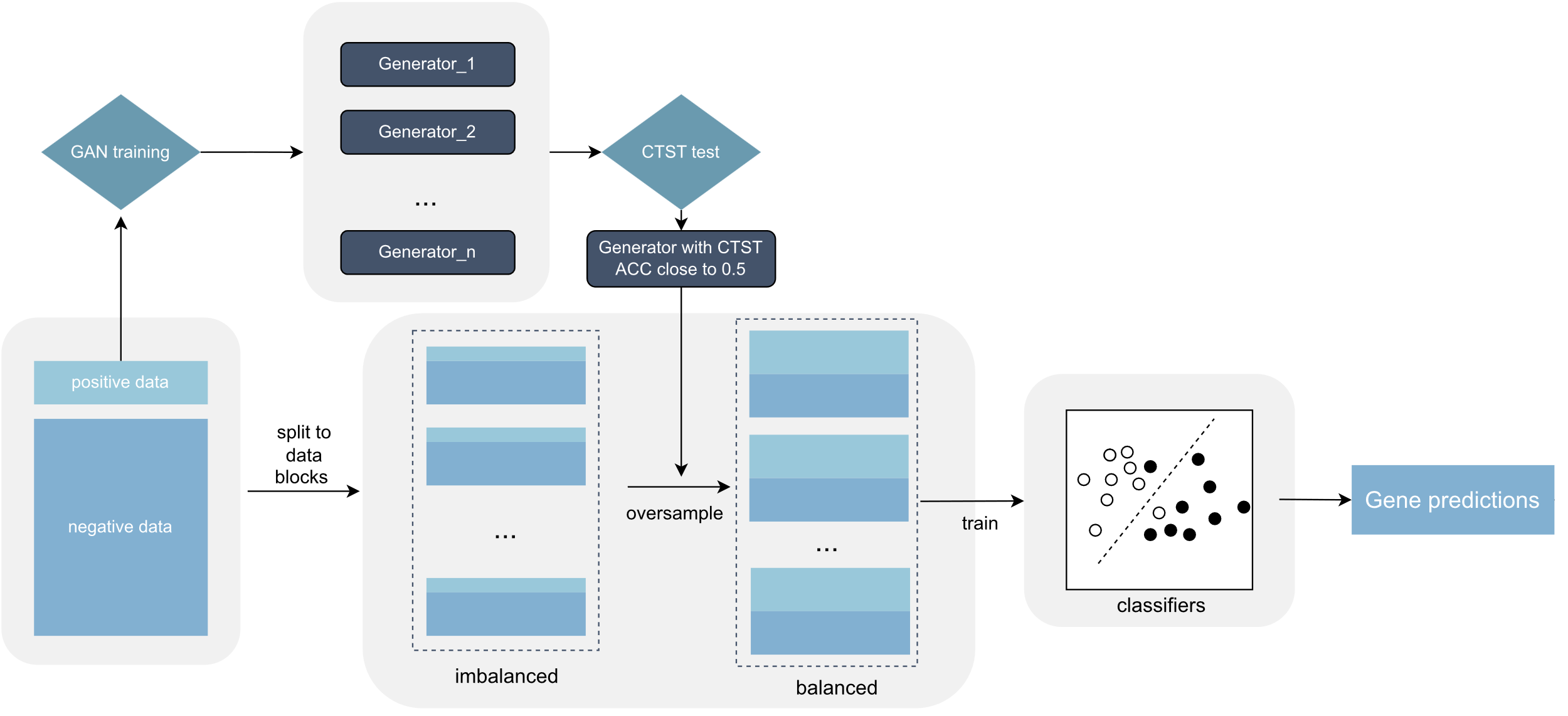


Figure 1: Workflow of our study

**2.2. Data set and feature transformation**

We use the dataset created and made available by Tsagkogeorga *et al*. Original dataset collected from Harmonizome[20] website(<https://maayanlab.cloud/Harmonizome>). 15 one-hot-encoded datasets were selected to construct our dataset. The dataset was initially standardized to continuous-value ranged from 0 to 1, or -1 to 1 where 1 indicated strong positive gene-feature association, 0 indicates no gene-feature association observed and -1 indicates strong negative gene-feature association.

The genes within these 15 datasets, along with their associated features, exhibit variations. The quantity of genes and features differs across datasets. Certain genes can be present in one or multiple datasets, while specific features may be present across various datasets as well. Utilizing pandas to merge the data based on the gene's official symbol results in a dataset comprising 26,936 genes. Each gene encompasses 50,176 features, forming a matrix of dimensions 26,936x50,176. Within the 15 datasets, certain genes lack specific features that are prevalent in other datasets. Consequently, features absent from a gene in this context will be marked as null. Subsequently, the values within the dataset marked as null will be assigned a value of 0. While the presence of 0 within the dataset signifies the absence of gene-feature associations, this will not impact the ultimate outcomes of training and prediction.

To gain high-informative features and reduce feature dimensions, following 2 kinds of features will be removed: (i) zero values more than 70%. (ii)variance less than 16% of whole.

Table 1: Numbers of gene sample for training, CV, and test

|  |  |  |  |
| --- | --- | --- | --- |
|  | Train & CV | Test | Total |
| Positive gene numbers | 74 | 18 | 92 |
| Negative gene numbers | 21044 | 5260 | 26304 |
| Total | 21118 | 5278 | 26396 |

**2.3. The data block constructure balancing.**

Given the significant imbalance between positive and negative samples in our dataset, directly applying oversampling may not enhance classifier performance (will discuss in chapter 3.2). Therefore, we further adopted a data block constructure for oversampling. In short, we divide the training dataset into subsets and then use oversampling to balance these subsets.

Specifically, these subsets are created by combining the same positive samples with different negative samples (The positive samples represent the minority samples), and the number of positive and negative samples in each subset is allocated proportionally based on the hyperparameters. In general, the number of positive samples in each subset is lower than the number of negative samples to achieve a balanced distribution of positive and negative samples after oversampling.

Formally, let (resp., ) be the number of positive samples (resp., negative samples). Let (resp.,) be the positive samples (resp., negative samples). Let *D* = be the entire dataset. Let λ = be the imbalanced ratio of the entire dataset. Assuming hyperparameter α, it is the parameter that controls the oversample rate. Algorithm 1 shows the pseudocodes of balancing data block constructure process.

|  |
| --- |
| **Algorithm 1**: Data block balancing |
| Input: Dataset *D*  Output: set of *K* data block |
| 1. Obtain , based on *D*; 2. λ = , *K* = *INT* (λ \* α); 3. Divide into *K* equal pieces { , , …, }; 4. for *i* = 1 to *K* do 5. = Oversample ( ) ; 6. Add into ; |
| 1. end for 2. return |

**2.4. The small-sample augmentation based on WGAN-GP.**

Generative Adversarial Networks (GANs) are neural networks capable of learning high-dimensional feature distributions from data samples. In general, a GAN consists of two neural networks: generator G and discriminator D. The generator receives a random priori noise as input, the output of the generator is considered to be fake samples, the discriminator's task is to distinguish between the samples generated by the generator and the real samples. To train the GAN, the generator and the discriminator play a minimax two-player game, that is, the generator generates samples that can fool the discriminator as much as possible, while the discriminator distinguishes between fake samples and real samples as much as possible. Ideally, when the generator is unable to generate more true samples and the discriminator is unable to distinguish between false and true samples, a Nash equilibrium will be reached in the minimax game of these two neural networks.

Formally, let be the object function, the minimax game is evaluated as:

(1)

where is sampled from the real data distribution and is sampled from the distribution of a priori noise . represents the expectation, represents the score assigned by the discriminator to , and represents the fake samples generated by using the noise sampled from the priori distribution . In a GAN, the generator and discriminator are trained alternatively. That is, the weights of one network are fixed while the other network is trained. Specifically, when training the discriminator , the generator is fixed first, and the discriminator is optimized by maximizing . Similarly, when training the generator the discriminator is fixed first, and the generator is optimized by minimizing .

However, conventional GANs face the training difficulties primarily due to the Kullback-Leibler (KL) divergence and Jensen-Shannon (JS) divergence loss functions. Specifically, when handling non-overlapping distributions, the JS divergence between and approximate a constant, which lead to gradient vanishing and even mode collapse [21]. WGAN attempts to address the issue of gradient vanishing in GANs by replacing them with the Wasserstein distance. Equation 2 denotes the value function of WGAN:

(2)

where  represents a function that satisfies the K-Lipschitz constraint, and are the parameters of and , respectively. The parameter is constrained within a range that ensures the function has a slope value less than K between any two points. Given the powerful fitting capability of neural networks, we can simply replace and with and , respectively. Thus, the object function of WGAN is represented by Equation 3.

(3)

Given the truth that is still a function satisfying K-Lipschitz constraint, all the parameters in will be clipped to a fixed range after each epoch, e.g., from -0.01 to 0.01. Compared to GAN, WGAN incorporates the K-Lipschitz constraint to address the issue of gradient vanishing and due to its effective smoothing of the value function, WGAN shows improved trainability. Nevertheless, WGAN's approach to the K-Lipschitz constraint is rough, as the direct parameter clipping post-training significantly impacts the network's fitting capability. Consequently, it becomes more difficult to fit complex distributions, resulting in a substantial number of weight values concentrated along the clipped boundaries [22]. This limitation significantly hampers the performance of WGAN. WGAN-GP takes a more 'moderate' approach to enforcing the K-Lipschitz constraint by incorporating a gradient penalty on the discriminator. More specifically, during discriminator's training, a gradient penalty is incorporated into discriminator's loss function. Let GP be the gradient penalty, equation 4 illustrates it.

(4)

And the object function of WGAN-GP becomes:

(5)

where is obtained by interpolating and , is the gradient penalty coefficient, which is an empirical value, typically a positive real. denotes the derivative of at .

Furthermore, the CNN structure in WGAN-GP is primarily designed for image feature extraction and is not well-suited for our gene expression data. Therefore, it is crucial to enhance WGAN-GP to better accommodate gene expression data. Accordingly, we substitute the CNN structure in WGAN-GP with the fully-connected neural network and perform additional optimization of the network parameters in order to tailor it specifically to our dataset. Each fully-connected layer is followed by a Leaky ReLU activation function with a negative slope of 0.2. In the discriminator, the number of neurons in each layer gradually decreases until it reaches 1; In the generator, the number of neurons in each layer progressively increases until it matches the number of features in the dataset.

**2.5. Further generator selection**

In this study, we use the CTST approach to select the generator that produces the optimal synthetic samples. Optimal synthetic samples are samples that can mimic the feature distribution of real training positive samples while not exactly as real training gene expression samples. In brief, when the number of synthetic samples and real samples is the same, CTST involves employing a binary classifier to differentiate between the two types of samples. If cross-validation results demonstrate that the classifier can effectively distinguish between the two types of samples, there will be a substantial distinction between the distributions of synthetic samples and real samples. Specifically, given two sample sets of the same size that follow two distributions P, Q. CTST considers accepting or rejecting the null hypothesis that P is equal to Q. If the null hypothesis is accepted, the classification accuracy of two class samples will be near chance level (50%), on the contrary, 100% [23].

During the training process of WGAN-GP, we incorporated the CTST process after each epoch to promptly assess the impact of network training and retain the necessary network weights for downstream tasks. Due to the time-consuming nature of network training, cross-validation is performed only once, and the network weights are retained based on the accuracy of this validation. However, these weights are susceptible to chance during the individual CTST process. This is because, due to the probabilistic nature of data generation, the network may generate samples that closely resemble a specific distribution, leading to a CTST result close to 0.5. (In reality, the network has started to learn the distribution of the target samples more effectively at this stage.) These weights can significantly influence the downstream task. To evaluate the impact of generator fitting across different epochs, we conducted an additional CTST experiment after WGAN-GP training. Multiple CTSTs were performed using different weights, and the generator with the mean accuracy closest to 0.5 was selected as the data generator for the downstream task. In this study, we conduct the CTST by using Support Vector Machine classification algorithm.

**2.5. Prediction models.**

**2.6. Training and evaluation**

This study includes a total of 92 gene expression data samples (positive samples) related to methylation pathways, with 74 samples allocated for training purposes. Initially, these positive training samples will be employed to train WGAN-GP for acquiring generators that can produce synthetic samples. Subsequently, they will be utilized for training the classification model.

To prevent mode collapse during the training prosses of WGAN-GP, we employ the Early Stopping mechanism. If the CTST result, which measures the dissimilarity between synthetic samples generated by the generator and real samples, falls below the preset threshold of 0.02, the model from that epoch will be saved and offered for selection in downstream tasks. To increase the number of available generator models for selection, the program does not halt immediately upon meeting the Early Stopping condition. Instead, it continues training until reaching the preset number of epochs.

To compare the performance of different classifiers, we employ five machine learning classifiers: Support Vector Machine (SVM), Gradient Boosting (GB), Gaussian Naive Bayes (GNB), Random Forest (RF), and Logistic Regression (LR). Following the division of the data based on the oversampling rate α, these classifiers will be trained using an equal number of data blocks. To determine the optimal hyperparameters for SVM, GB, and RF, we employ the grid-search technique and select the combination of hyperparameters that yields the highest accuracy score through 3-fold cross validation.

Four metrics are utilized for evaluating the classifiers’ performance: Accuracy, Recall, F1 Score, AUROC, and Matthews Correlation Coefficient (MCC). Due to the highly imbalanced ratio of positive and negative samples in the test set, the remaining four metrics, excluding AUROC, will be computed using multiple data blocks based on test set, and the average of their performance metrics across all data blocks will be considered as the test result. In contrast to the training set, the splitting process for the test set data blocks does not contain the oversampling step.

**3. Result:**

**3.1. Influence of hyperparameters and training strategies**

In order to investigate the effect of the oversampling rate α (mentioned in chapter 2.3) on the model performance, we tested the performance using multiple machine learning models based on different oversampling rates. As shown in Figure 3, the horizontal axis is the oversampling rate, the vertical axis is the test Accuracy score, and the lines in different categories indicate different classes of classifiers. Our experiments involved nine α values and five distinct classifiers. Generally, specific classifiers exhibited varied test results with different α values. Fixed α values also produced diverse test results across classifiers. Additionally, inadequate synthetic samples (α close to 1) or excessive samples (α close to 0) led to performance degradation. The optimal test accuracy (0.9367) was achieved with an α of 0.4 paired with the Random Forest (RF) classifier. Conversely, the lowest teste accuracy (0.8929) was observed with an α of 0.2 paired with the RF classifier, indicating a difference of approximately 4.4%. Notably, RF is the classifier most impacted by variations in alpha. The classifier least influenced by α is the Gaussian Naive Bayes (GNB), showing a marginal difference of about 0.6% between the best result (0.9107) and the worst result (0.9039). The poorest results among other classifiers were 1% lower for Support Vector Machine (SVM), 2.2% lower for Gradient Boosting (GB), and 0.8% lower for Logistic Regression (LR) compared to the best results.

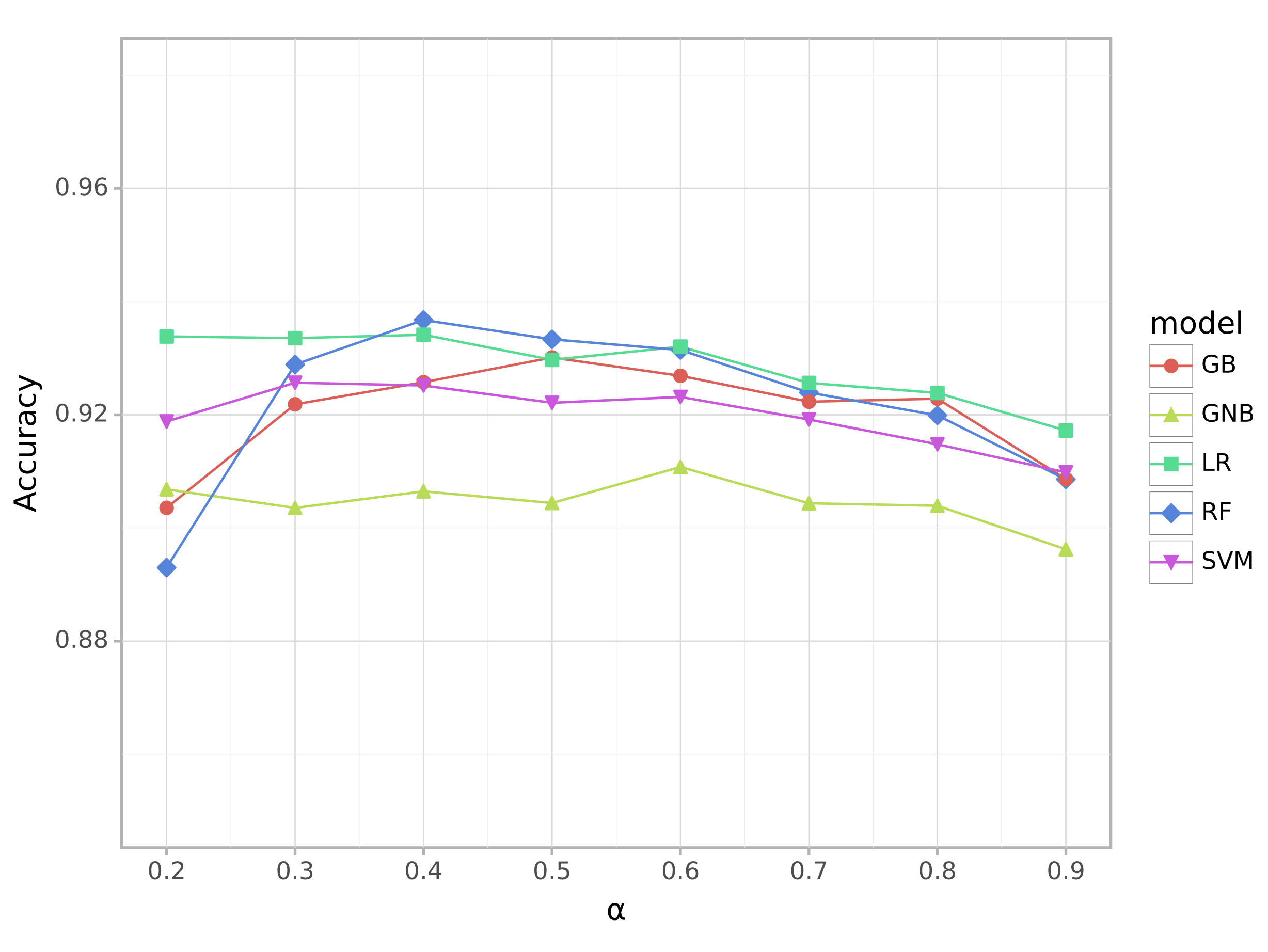
****

Figure 2: Impact of oversample rate α

**3.2 ablation study**

Our approach consists of three components: generator selection, data block constructure, and oversampling process. To study the effectiveness of these components, we removed the data block constructure and oversampling processes to obtain two variants of the scheme. We use these two variants to train five machine learning models respectively. Subsequently, we evaluate their performance using 4 metrics to assess the impact of these two components on performance. Figure 3 depicts the results of the ablation study.

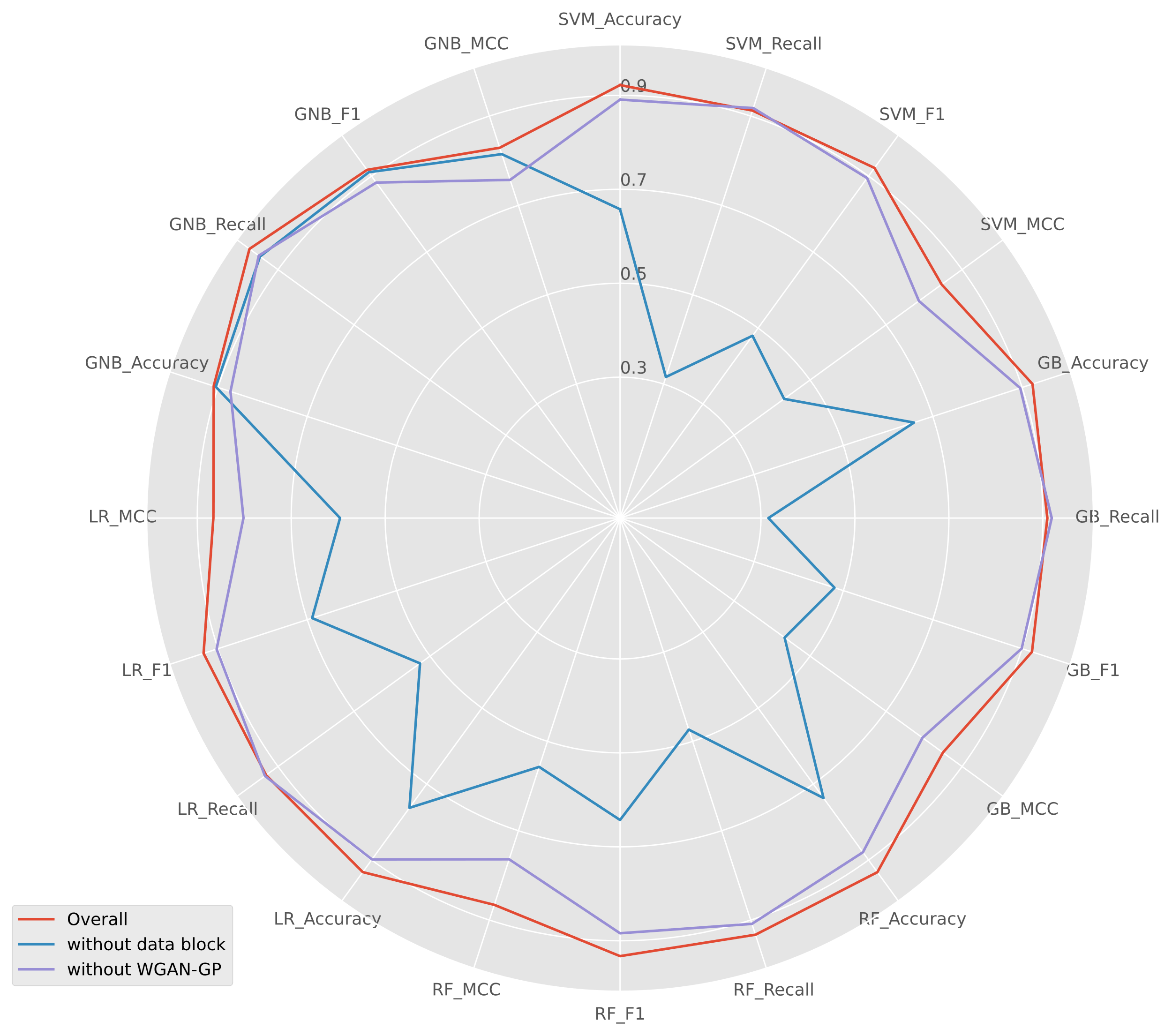


Figure 3: Overview of ablation study.

Fig. 4 illustrates the MCC results of three schemes: our proposed complete scheme (represented by red bars), the variant scheme without data block constructure (represented by green bars), and the variant scheme without the oversampling process (represented by blue bars). The horizontal axis displays the classifiers used, while the vertical axis represents the MCC scores of the models for the three schemes. In general, our proposed complete scheme exhibits the best performance across all five classifiers. However, the absence of data block constructure or the oversampling process results in varying degrees of performance degradation. Among the five classifiers, the GNB classifier shows the least sensitivity to the two components, with a decrease in GNB performance of 1.4% when data block constructure is missing, and a decrease of 7% when WGAN-GP oversampling is missing. In contrast, the other four classifiers demonstrate higher sensitivity to data block constructure. In the absence of data block constructure, GB and SVM experience nearly a 50% performance decrease, while LR and RF experience performance drops of 26.6% and 30.88% respectively. The situation is similar when the oversampling process is absent. In this case, GB, SVM, RF, and LR experience performance decreases of 5.3%, 5.9%, 10.2%, and 6.4% respectively. This experimental result clearly demonstrates that the approach of directly oversampling and generating a large number of synthetic samples to address imbalanced datasets is not effective for classification problems with a high degree of imbalance. In contrast, our proposed scheme combines the advantages of oversampling and undersampling. By incorporating data blocks, it significantly reduces the need for synthetic samples while preserving the original sample distribution and expanding the dataset.

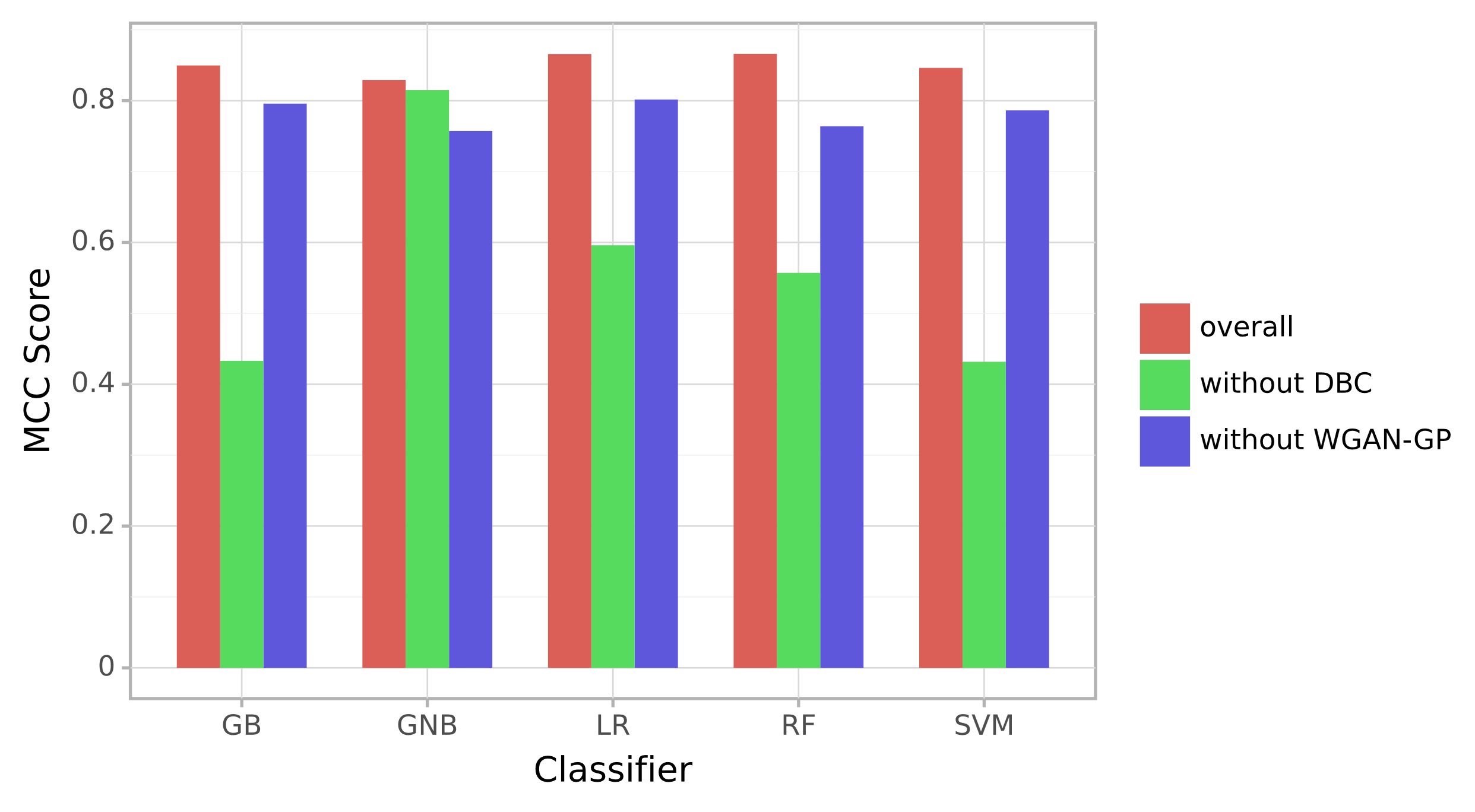


Figure 4: MCC results of ablation study.

A total of 61 weights were saved using the preset hyperparameters. The additional CTST experimental results for these 61 weights are presented in Figure 5, sorted based on their variance. After conducting 200 repetitions of the experiment, slight differences were observed in the mean values of all generator test results. Most of the generator test results still satisfy the preset hyperparameter criteria, falling between 0.498 and 0.502 as denoted by the green dots in the figure. A few generator test results deviate from the preset criteria but meet the requirements of the downstream tasks, ranging from 0.4 to 0.498 and 0.502 to 0.6 as indicated by the orange points in the figure. This indicates an increase in their variance. A few generators, represented by the red dots, exhibit test results significantly deviating from the 0.5 criterion, with an absolute difference greater than 0.1, also higher variance. In such cases, the samples generated by these generators can introduce bias in the training of the classifier model. These test results confirm necessity of adding additional CTST.

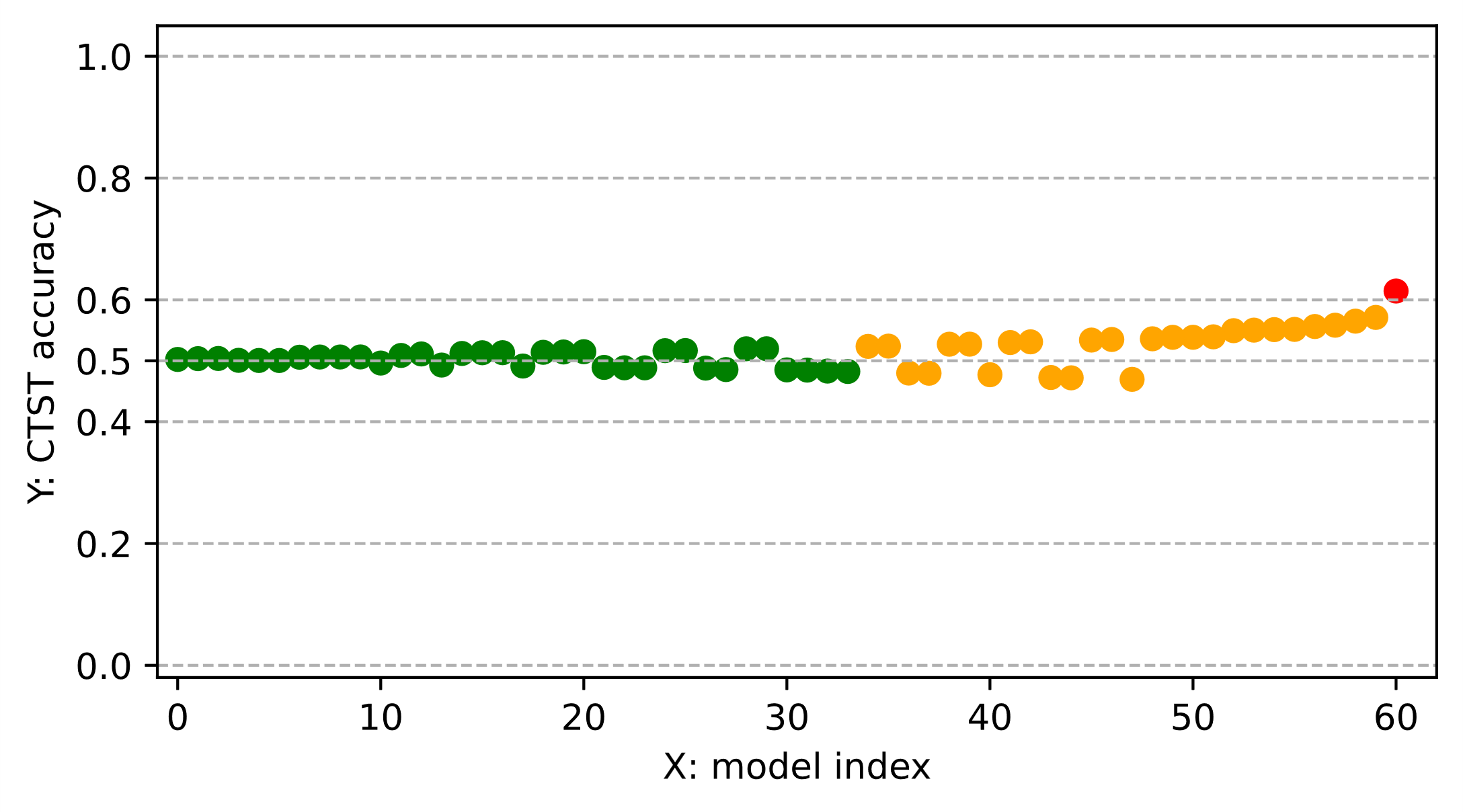


Figure 5: CTST result over 61 saved model weight.

**3.3. WGAN-GP successfully generated high-quality positive gene samples.**

Overall, the samples synthesized through WGAN-GP exhibited a strong alignment with the original positive samples. Following each training epoch, we incorporated supplementary CTST. As shown in Figure 6, at the outset of generator training, the generated samples diverged notably from the distribution of positive samples, yielding a CTST accuracy of 1.0. This observation underscores the effective discrimination achieved by the CTST classifier between the generated and positive samples.

Around the 200th epoch, the generator began capturing the distinctive feature distribution of the positive samples. Consequently, an overlap emerged in the feature space between the generated and positive samples, leading to a decrease in CTST accuracy to approximately 0.7. As training progressed, by the 500th epoch, this overlapping region expanded further, causing the CTST accuracy to diminish to about 0.6.

After over 800 additional training iterations, WGAN-GP successfully internalized the genetic feature distribution of authentic positive samples. Evidently, the CTST accuracy decreased to 0.5 and exhibited fluctuation around this value, signifying the generator's ability to effectively align with the positive sample distribution. Additionally, the training loss curve further indicated that the generator attained a local optimum as the CTST accuracy reached 0.5.

Furthermore, conducting independent CTST evaluations on each saved generator model unveiled that more than half of the generators could consistently uphold a classifier accuracy of around 0.5. This observation implies the challenge of distinguishing between generated samples and positive samples using the classifier. Undoubtedly, the generator excelled at producing high-caliber synthetic samples.

图表, 散点图

描述已自动生成

Figure 6: Distribution changes between positive and synthetic samples

During the early stages of developing generative adversarial networks, researchers faced the challenge of training difficulties. While WGAN-GP partially addressed these difficulties by incorporating gradient penalties, the role of proper neural network design and hyperparameter tuning remained crucial in GAN training. In addition to visualizing samples at various epochs, observing changes in loss during GAN training was also effective. For WGAN-GP, the generator's loss represented the disparity between the generated sample and the real sample. Figure 7 illustrates that the generator exhibited poor sample fitting ability at the beginning of training, resulting in an increase in loss throughout the training process. After adversarial training with the discriminator, the generator started capturing the underlying sample distribution, leading to a decrease in loss. As the training progressed, the generator's loss began to fluctuate, indicating its position in the intermediate stage. Around 500-600 epochs, the generator and discriminator reached a Nash equilibrium in the game, demonstrating a consistent reduction in loss and subsequent stabilization. This indicated that the entire generative adversarial network had reached an approximate optimal state.

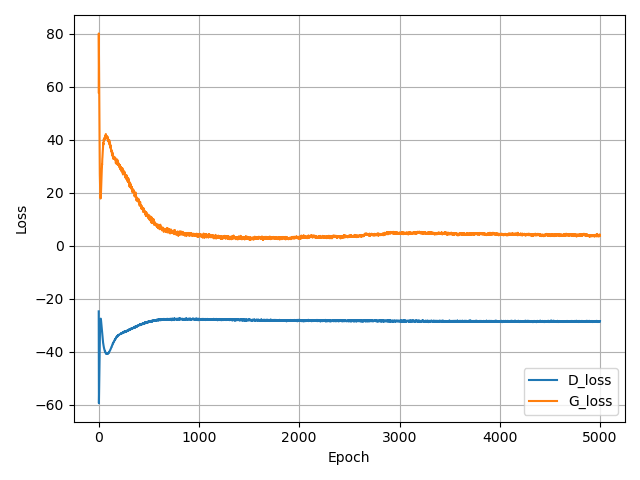


Figure 7: WGAN-GP train loss over 5000 epochs. Blue line indicates discriminator loss and orange line indicates generator loss.

**3.4. Performance comparison**

We conducted the test scheme as illustrated in Chapter 2.6, and the results are presented in Table 2. Overall, among the four schemes, our approach exhibits superior prediction performance with most classifier combinations. The combination with SVM classifier improves accuracy by 16.01%, recall by 38.81%, F1 by 23.34%, and MCC by 25.14%, respectively, compared to the second place. The combination with GB classifier achieves improvements of 13.77%, 33.91%, 19.35%, and 21.87% over the second place. Additionally, the combination with RF classifier surpasses the second place by 16.83%, 40.58%, 24.26%, and 26.73%. The combination with LR classifier outperforms the second place by 11.69%, 30.39%, 15.92%, and 1889%. In terms of performance metrics, our scheme consistently attains leading results across most metrics, securing the best accuracy (0.9320), F1 (0.9354), and MCC (0.9327) values when combined with LR. But, our model doesn't lead in combination with GNB, the RandomOversample scheme achieves the best performance with an average accuracy of 0.9183, a recall of 1.0, F1 of 0.9261, and an MCC of 0.8505. This suggests the method's inclination to identify more positive samples. Nonetheless, our scheme remains compatible as it achieves values for accuracy, recall, F1, and MCC closest to the leading scheme with only a small gap.

Notably, some approaches encountered a significant problem, classifying all the samples in the test set as negative. This resulted in their average accuracy, recall, F1, and MCC all being 0.5, 0, 0, 0, respectively. In such cases, we considered these approaches unusable and presented these values as N/A. Among the three compared schemes, the combination of SMOTE with SVM exhibits invalid values. Similarly, RandomOversample encounters the same issue when combined with SVM, GB and RF. While BorderlineSMOTE mitigates the problem but performs worse than our scheme. In comparison, our method demonstrates better data compatibility and can more effectively accommodate highly unbalanced data.

Table 2: Accuracy, Recall, F1 and MCC for 4 approaches

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method**  **Classifier** | SMOTE | | | | BorderlineSMOTE | | | | RandomOversample | | | | Ours | | | |
| Accuracy | Recall | F1 | MCC | Accuracy | Recall | F1 | MCC | Accuracy | Recall | F1 | MCC | Accuracy | Recall | F1 | MCC |
| SVM | N/A | N/A | N/A | N/A | 0.7630 | 0.5263 | 0.6895 | 0.5973 | N/A | N/A | N/A | N/A | **0.9232** | **0.9342** | **0.9145** | **0.8487** |
| GB | 0.7629 | 0.5263 | 0.6894 | 0.5970 | 0.7891 | 0.5789 | 0.7330 | 0.6375 | 0.7631 | 0.5263 | 0.6896 | 0.5976 | **0.9269** | **0.9379** | **0.9181** | **0.8562** |
| RF | 0.7368 | 0.4736 | 0.6428 | 0.5570 | 0.5789 | 0.1578 | 0.2727 | 0.2927 | 0.5263 | 0.0526 | 0.1 | 0.1644 | **0.9315** | **0.9345** | **0.9321** | **0.8650** |
| LR | 0.8413 | 0.6842 | 0.8118 | 0.7192 | 0.8151 | 0.6315 | 0.7735 | 0.6776 | 0.8153 | 0.6315 | 0.7737 | 0.6781 | **0.9320** | **0.9327** | **0.9354** | **0.8665** |
| GNB | **0.9164** | **0.9473** | **0.9201** | **0.8365** | 0.8094 | 0.6315 | 0.7684 | 0.6627 | 0.8115 | 0.7894 | 0.8088 | 0.6263 | 0.9108 | 0.8673 | 0.9178 | 0.8324 |

To investigate the difference in predictive performance between samples synthesized by WGAN-GP and other schemes, we substitute WGAN-GP with three oversampling schemes in our approach, resulting in three new variants. These three new variants of the scheme will balance the data blocks instead of balancing the entire dataset before training, which is similar to our proposed approach. α is set to 0.6, No oversampling method was set as baseline.

Table 3 shows the test results of different kinds of oversampling schemes combined with data block constructure. Generally, WGAN-GP outperforms the other 4 compared schemes on most metrics for most classifiers and achieves the highest overall accuracy, recall, F1, and MCC scores. Specifically, when using the SVM classifier, the WGAN-GP oversampling method attained the highest accuracy (0.9232), recall (0.9342), F1 (0.9230), and MCC (0.8487) scores. When using the GB classifier, WGAN-GP achieved the highest accuracy (0.9269), F1 (0.9266), and MCC (0.8562) scores. WGAN-GP achieved the highest accuracy (0.9315), recall (0.9321), F1 (0.9322), and MCC (0.8650) scores when the RF classifier was used. When using the LR classifier, WGAN-GP achieved the highest accuracy (0.9320), precision (0.9327), F1 (0.9328), and MCC (0.8665) scores. When using the GNB classifier, WGAN-GP achieved the highest Recall (0.9785), F1 (0.9178), and MCC (0.8324) scores. Among them, the combination scheme of WGAN-GP with LR attained the overall highest accuracy, F1, and MCC scores, and the combination with GNB attained the overall highest recall score. The highest overall precision score was achieved by the combination of SMOTE and GB.

It is worth noting that the WGAN-GP oversampling method, when combined with different classifiers, can achieve the highest MCC and F1 scores for each respective classifier. Given that the MCC metric is commonly used to evaluate binary classification models, particularly for unbalanced datasets, this suggests that WGAN-GP generates samples with better diversity, thereby reducing the classifier's bias towards the majority class.

Table 3: Comparison of model performance with data block constructure

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Oversample Method | Classifier | Accuracy | Precision | Recall | F1 | MCC |
| WGAN-GP | SVM | **0.9232** | **0.9342** | 0.9145 | **0.9230** | **0.8487** |
| GB | **0.9269** | 0.9379 | 0.9181 | **0.9266** | **0.8562** |
| RF | **0.9315** | 0.9345 | **0.9321** | **0.9322** | **0.8650** |
| LR | **0.9320** | **0.9327** | 0.9354 | **0.9328** | **0.8665** |
| GNB | 0.9108 | 0.8673 | **0.9785** | **0.9178** | **0.8324** |
| BorderlineSMOTE | SVM | 0.9171 | 0.9277 | 0.9087 | 0.9168 | 0.8367 |
| GB | 0.9252 | 0.9365 | 0.9159 | 0.9249 | 0.8529 |
| RF | 0.9293 | 0.9328 | 0.9291 | 0.9299 | 0.8605 |
| LR | 0.9244 | 0.9217 | 0.9321 | 0.9255 | 0.8514 |
| GNB | 0.9046 | 0.8658 | 0.9662 | 0.9114 | 0.8189 |
| SMOTE | SVM | 0.9129 | 0.9229 | 0.9055 | 0.9121 | 0.8291 |
| GB | 0.9237 | **0.9389** | 0.9099 | 0.9229 | 0.8501 |
| RF | 0.9301 | **0.9352** | 0.9282 | 0.9307 | 0.8622 |
| LR | 0.9248 | 0.9255 | 0.9282 | 0.9257 | 0.8519 |
| GNB | 0.9042 | **0.8736** | 0.9518 | 0.9095 | 0.8151 |
| RandomOversample | SVM | 0.9193 | 0.9308 | 0.9099 | 0.9190 | 0.8411 |
| GB | 0.9181 | 0.9289 | 0.9096 | 0.9178 | 0.8388 |
| RF | 0.9235 | 0.9210 | 0.9312 | 0.9249 | 0.8493 |
| LR | 0.9286 | 0.9259 | **0.9360** | 0.9297 | 0.8597 |
| GNB | **0.9112** | 0.8695 | 0.9751 | 0.9177 | 0.8324 |
| None | SVM | 0.8909 | 0.8763 | **0.9178** | 0.8947 | 0.7863 |
| GB | 0.8957 | 0.8834 | **0.9189** | 0.8989 | 0.7957 |
| RF | 0.8795 | 0.8645 | 0.9085 | 0.8840 | 0.7639 |
| LR | 0.8983 | 0.8767 | 0.9342 | 0.9029 | 0.8016 |
| GNB | 0.8719 | 0.8279 | 0.9506 | 0.8829 | 0.7570 |

**3.5 GO terms enrichment analyses.**

To comprehend the model's predictions for genes related to the methylation pathway, we employed trained classifiers to assign confidence scores to all genes. With α set to 0.6, each classifier retained 176 models. The average score of these 176 models for each gene was then calculated as the classifier's scoring result. Subsequently, high-confidence genes assessed by each classifier were utilized for gene enrichment analysis. Here, we define high-confidence genes as the top 1% of genes ranked by confidence. Based on the gene quantity in our dataset, each classifier typically has 270 high-confidence genes. Since GNB tends to blindly assign high confidence to many genes, for GNB, these genes should be considered in the top 1500 of confidence rankings (when rankings exceed 1500, other classifiers almost always give confidence levels below 0.5). Figure 7 compares the top 20 GO terms ranked by each classifier. Overall, the predictions of the five classifiers exhibit high consistency. GO terms enrichment analysis indicates that high-confidence genes are strongly enriched for RNA methylation functions. The terms with the highest enrichment include methylation, RNA modification, RNA splicing, and RNA localization. Other terms related to rRNA, mRNA, tRNA, and ncRNA also appear in the enrichment analysis report, with the gene quantity enriched on these GO TERMS reaching over 200. This confirms that the predicted genes are closely associated with the RNA methylation process.

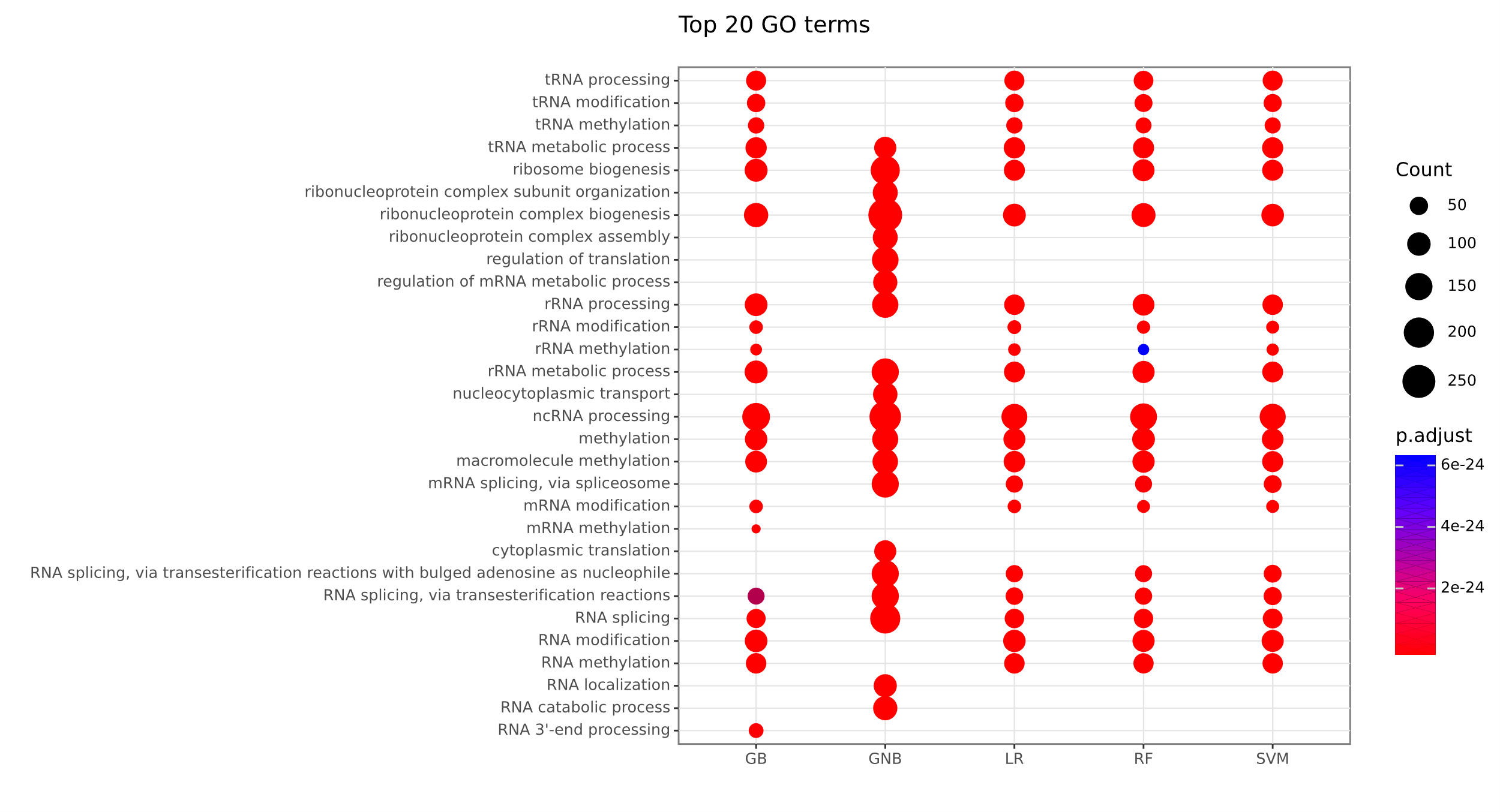


Figure 7: GO terms enrichment analyses of high-confidence genes

**Discussion**

**Conclusion**

In this study, we illustrate the importance of RNA methylation and the imbalance learning problem in the prediction task of genes associated with RNA methylation pathways. We propose a deep learning-based Wasserstein generative adversarial net with gradient penalty (WGAN-GP) approach to alleviate the imbalance problem in gene prediction tasks. We use five machine learning classifiers (SVM, GB, GNB, RF, LR) for model training and evaluate the model performance using various metrics. Experimental results show that the improved WGAN-GP effectively captures the features of gene expression data and accurately models the distribution of positive samples in the feature space. Besides, it generates synthetic samples that are difficult to distinguish from real positive samples but with better diversity than other oversample method. Compared to other conventional schemes, our study is effective and performance leading. Our research will be considered for future extension and applicability to other issues: 1) multi-classification problems with datasets of high imbalance ratios; 2) the use of deep neural networks as an alternative to machine-learning classifiers.

**References:**

[1]. Boccaletto, Pietro, et al. "MODOMICS: a database of RNA modification pathways. 2017 update." Nucleic acids research 46.D1 (2018): D303-D307.

[2] Xie, S., Chen, W., Chen, K., Chang, Y., Yang, F., Lin, A., Shu, Q., Zhou, T., & Yan, X. (2020). Emerging roles of RNA methylation in gastrointestinal cancers. In Cancer Cell International (Vol. 20, Issue 1). https://doi.org/10.1186/s12935-020-01679-w

[3] Zhang, M., Song, J., Yuan, W., Zhang, W., & Sun, Z. (2021). Roles of RNA Methylation on Tumor Immunity and Clinical Implications. In Frontiers in Immunology (Vol. 12). https://doi.org/10.3389/fimmu.2021.641507

[4] Qin, Y., Li, L., Luo, E., Hou, J., Yan, G., Wang, d. O., Qiao, Y., & Tang, c. Heng. H. (2020). Role of m6A RNA methylation in cardiovascular disease (Review). In International Journal of Molecular Medicine (Vol. 46, Issue 6). https://doi.org/10.3892/ijmm.2020.4746

[5] Mendel, M., Chen, K. M., Homolka, D., Gos, P., Pandey, R. R., McCarthy, A. A., & Pillai, R. S. (2018). Methylation of Structured RNA by the m6A Writer METTL16 Is Essential for Mouse Embryonic Development. Molecular Cell, 71(6). https://doi.org/10.1016/j.molcel.2018.08.004

[6] Flores, J. v., Cordero-Espinoza, L., Oeztuerk-Winder, F., Andersson-Rolf, A., Selmi, T., Blanco, S., Tailor, J., Dietmann, S., & Frye, M. (2017). Cytosine-5 RNA Methylation Regulates Neural Stem Cell Differentiation and Motility. Stem Cell Reports, 8(1). https://doi.org/10.1016/j.stemcr.2016.11.014

[7] He, P. C., & He, C. (2021). m 6 A RNA methylation: from mechanisms to therapeutic potential. The EMBO Journal, 40(3). https://doi.org/10.15252/embj.2020105977

[8]. Tsagkogeorga, G., Santos-Rosa, H., Alendar, A., Leggate, D., Rausch, O., Kouzarides, T., Weisser, H., & Han, N. (2022). Predicting genes associated with RNA methylation pathways using machine learning. Communications Biology, 5(1).

[9]. Fernández, A., García, S., Herrera, F., & Chawla, N. v. (2018). SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary. In Journal of Artificial Intelligence Research (Vol. 61).

[10]. Chawla, N. v, Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. In Journal of Artificial Intelligence Research (Vol. 16).

[11].Blagus, R., & Lusa, L. (2013). SMOTE for high-dimensional class-imbalanced data. BMC

Bioinformatics, 14, 106.

[12]. Han, H., Wang, W.-Y., & Mao, B.-H. (2005). Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. In LNCS (Vol. 3644).

[13]. Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (n.d.). Generative Adversarial Nets.

[14]. Ma, L., Shuai, R., Ran, X., Liu, W., & Ye, C. (2020). Combining DC-GAN with ResNet for blood cell image classification. Medical and Biological Engineering and Computing, 58(6). https://doi.org/10.1007/s11517-020-02163-3[15]. Gadermayr M, Li K, Müller M, et al. Domain‐specific data augmentation for segmenting MR images of fatty infiltrated human thighs with neural networks[J]. Journal of Magnetic Resonance Imaging, 2019, 49(6): 1676-1683.

[16]. Frid-Adar, M., Diamant, I., Klang, E., Amitai, M., Goldberger, J., & Greenspan, H. (2018). GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification. Neurocomputing, 321. https://doi.org/10.1016/j.neucom.2018.09.013

[17]. Xiao, Y., Wu, J., & Lin, Z. (2021). Cancer diagnosis using generative adversarial networks based on deep learning from imbalanced data. Computers in Biology and Medicine, 135. https://doi.org/10.1016/j.compbiomed.2021.104540

[18] Gulrajani I, Ahmed F, Arjovsky M, et al. Improved training of wasserstein gans[J]. Advances in neural information processing systems, 2017, 30.

[19] Lopez-Paz, D., & Oquab, M. (2017). Revisiting classifier two-sample tests. 5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings.

[20]. Rouillard, A. D., Gundersen, G. W., Fernandez, N. F., Wang, Z., Monteiro, C. D., McDermott, M. G., & Ma’ayan, A. (2016). The harmonizome: a collection of processed datasets gathered to serve and mine knowledge about genes and proteins. Database : The Journal of Biological Databases and Curation, 2016.https://doi.org/10.1093/database/baw100

[21] Wan, C., & Jones, D. T. (2020). Protein function prediction is improved by creating synthetic feature samples with generative adversarial networks. Nature Machine Intelligence, 2(9), 540–550. https://doi.org/10.1038/s42256-020-0222-1

[22] L. Qu, Y. Wang, T. Yang, L. Zhang and Y. Sun, "WGAN-GP-Based Synthetic Radar Spectrogram Augmentation in Human Activity Recognition," 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS, Brussels, Belgium, 2021, pp. 2532-2535, doi: 10.1109/IGARSS47720.2021.9554556.

[23] Panwar, S., Rad, P., Jung, T. P., & Huang, Y. (2020). Modeling EEG Data Distribution with a Wasserstein Generative Adversarial Network to Predict RSVP Events. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 28(8), 1720–1730. https://doi.org/10.1109/TNSRE.2020.3006180