Article

Not peer-reviewed version

Enhancing the Classification of Biosynthetic Gene Clusters through Comprehensive NLP-Based Approach

Dwijesh Chandra Mishra *, Sharanbasappa D Madival *, Anu Sharma, Neeraj Budhlakoti, Krishna Kumar Chaturvedi, Ulavappa Basavanneppa Angadi, Mohammad Samir Farooqi, Sudhir Srivastava, Pavana Basavaraja, Alka Arora, Girish Kumar Jha, Shesh Rai

Posted Date: 25 October 2023

doi: 10.20944/preprints202310.1564.v1

Keywords: Biosynthetic Gene Clusters; Natural Language Processing; Machine Learning; Hybrids PKS-NRPS; SMOTE

Article

Enhancing the Classification of Biosynthetic Gene Clusters through Comprehensive NLP-Based Approach

Sharanbasappa D Madival [1], Dwijesh Chandra Mishra [2,*], Anu Sharma [2], Neeraj Budhlakoti [2], Krishna Kumar Chaturvedi [2], Ulavappa B Angadi [2], Mohammad Samir Farooqi [2], Sudhir Srivastava [2], Pavana B [2], Alka Arora [2], Girish K. Jha [2] and Shesh N. Rai [3,4,5]

[1] Graduate School, ICAR-Indian Agricultural Research Institute, New Delhi 110012, India
[2] ICAR-Indian Agricultural Statistics Research Institute, New Delhi 110012, India
[3] Cancer Data Science Center, University of Cincinnati, College of Medicine, Cincinnati, Ohio, USA
[4] Biostatistics and Informatics Shared Resources, University of Cincinnati Cancer Center, Cincinnati, Ohio, USA
[5] Division of Biostatistics and Bioinformatics, Department of Environmental Health Sciences, University of Cincinnati, College of Medicine, Cincinnati, Ohio, USA
* Correspondence: dwij.mishra@gmail.com

Abstract: Biosynthetic gene clusters are specific genomic regions in microorganisms, like bacteria and fungi, responsible for producing bioactive compounds. Identifying these clusters is complex due to their diverse nature. This research presents a comprehensive approach to effective BGC identification. The study focuses on five classes of Natural Products: PKS , NRPS, RiPP, Terpenes, and Hybrid PKS-NRPS. Data was gathered from the MiBIG database in GBK format. Protein sequences from each file were extracted, and sequences under the same BGC ID were combined. Physicochemical properties were calculated, and sequence embeddings were generated using NLP techniques like CountVec, TFIDF, and Word2Vec specific to each NP class. An integrated feature matrix was created by merging physicochemical properties and generated embeddings. This matrix was used for training and testing of nine ML models such SVM, RF and many more. The study explored data balancing techniques with and without SMOTE and employed Grid Search for parameter optimization. This led to six datasets and 54 models. The LR model, using TFIDF with SMOTE, emerged as the most effective, achieving an accuracy of 0.96, AUC of 0.9912, and other strong metrics. This method enhances BGC identification for drug development, offering a broader understanding of their applications in medicine and biotechnology.

Keywords: Biosynthetic Gene Clusters; Natural Language Processing; Machine Learning; Hybrids PKS-NRPS; SMOTE

1. Introduction

Natural Products refer to a diverse group of chemical compounds produced by living organisms, including plants, bacteria, fungi, and marine organisms [1]. These compounds have been extensively explored for their pharmaceutical, agricultural, and industrial applications due to their remarkable biological activities. Many well-known drugs, such as Penicillin, Taxol, and Erythromycin, are Natural Products or derivatives thereof, highlighting their significance in medicine [2].

The biosynthesis of Natural Products is governed by specific genomic regions known as Biosynthetic Gene Clusters (BGCs) [3]. These are specific genomic regions responsible for encoding enzymes and regulatory proteins involved in the synthesis of Natural Products (Figure **1**). These clusters consist of contiguous genes and exhibit conserved genetic architectures, making them identifiable through sequence motifs and codon usage bias [4]. Natural Products' chemical diversity arises from the use of building blocks, biosynthetic chemistry, and tailoring enzymes. Environmental signals play a crucial role in redirecting primary metabolites towards secondary metabolism, where core biosynthetic enzymes combine these metabolites to form intricate structures. Tailoring enzymes then modify these structures to produce the final Natural Products.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 October 2023 doi:10.20944/preprints202310.1564.v1

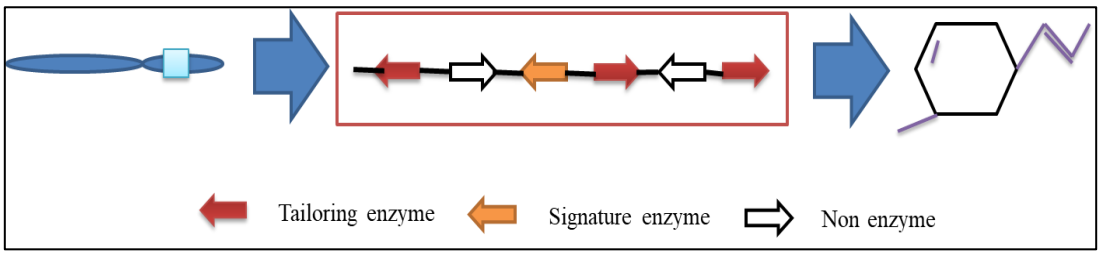

Figure 1. Overview of Biosynthetic gene clusters.

BGCs are associated with the synthesis of various Natural Products, each originating from specific precursors. For example, Polyketide Synthases (PKSs) use acyl-CoA or malonyl-CoA as precursors to form polyketides [5], while Non-Ribosomal Peptide Synthetases (NRPSs) utilize aminoacyl-tRNA to produce nonribosomal peptides [6]. Ribosomally synthesized and post-translationally modified Peptides (RiPPs) have precursor peptides that undergo various modifications to form diverse RiPPs, including antimicrobial peptides and lantibiotics [7]. Terpene synthases use isopentenyl pyrophosphate (IPP) and dimethylallyl pyrophosphate (DMAPP) as precursors to synthesize terpenes [8]. Some BGCs are hybrids of PKS and NRPS gene clusters, using both acyl-CoA/malonyl-CoA and aminoacyl-tRNA precursors, combining the biosynthetic pathways of PKS and NRPS.

Understanding BGCs is crucial for elucidating the biosynthetic pathways of Natural Products, optimizing their production, and discovering novel bioactive compounds. However, BGC classification remains challenging due to its complex and diverse nature. The diverse range of bioactive compounds produced through these pathways presents exciting opportunities for novel therapeutics, antimicrobial agents, and other applications with the potential to address various health challenges.

Traditional methods for BGC identification have predominantly relied on laborious and time-consuming experimental approaches. Gene knockout experiments, for instance, involve the inactivation of specific genes within an organism's genome to observe the impact on the production of Natural Product [9]. While informative, these experiments are resource-intensive and limited in scope, as they require prior knowledge of the target gene. Chemical assays have also been employed to detect the presence of specific Natural Products. Bioactive compounds are isolated from the organism of interest, and their chemical structures are elucidated through spectroscopic techniques [10]. Despite the fact that traditional approaches can directly detect the target molecule, they are not appropriate for identifying novel BGCs or compounds with low abundance. Traditional methods are useful, but they come with many difficulties because of the complexity and variety of BGCs.

In recent years, the advent of genomics and sequencing technologies has revolutionized the field of BGC identification. Genome mining approaches involve the systematic analysis of genomic data to predict and annotate BGCs. This approach relies on the identification of genes associated with particular biosynthetic pathways, which are then linked to the production of Natural Products. Despite these advancements, traditional methods have certain limitations. Many microbial species are challenging to culture in the laboratory, limiting access to their biosynthetic potential. Additionally, the huge amount of unexplored microbial diversity presents a challenge in discovering novel Natural Products and their associated BGCs.

In recent years, the Artificial Intelligence (AI) field with Natural Language Processing (NLP) and ML techniques has shown its huge potential in biological research. Automating and accelerating the analysis and identification of BGC, AI, and NLP may transform this field. NLP-based techniques extract valuable information from scientific literature and databases, facilitating more accurate BGC prediction by transforming textual data into structured datasets. Meanwhile, ML approaches, particularly supervised and unsupervised learning models, have been crucial in deciphering complex relationships within biological data. They enable precise classification of gene clusters using curated datasets of known BGCs and can even discover novel BGCs without prior annotated data. Studies

 doi:10.20944/preprints202310.1564.v1

like DeepBGC [12] and PRISM have showcased the potential of AI-driven tools in BGC identification from genomic and metagenomic datasets, respectively [13,14].

The rationale behind the current study is to overcome the limitations and challenges faced by previous efforts for BGC classification. Existing tools like antiSMASH, DeepBGC, and ML-Miner while valuable, had their shortcomings. The antiSMASH relied on Hidden Markov Models (HMMs) and struggled with rediscovery problems and identifying certain BGC classes like RiPPs. While, DeepBGC depended on existing Pfam families, leading to less robust results and increased false positives, on the other hand, ML-Miner [15] faces problems in effectively identifying and distinguishing a challenging class like hybrids of PKS-NRPS.

To address these shortcomings, we aimed to develop a novel approach that would be more exhaustive, accurate, and comprehensive. Our strategy involved integrating both sequence features and physicochemical properties, recognizing the importance of combining diverse data sources for better insights.

2. Materials and Methods

2.1. Data Collection and Preparation

To gather comprehensive and relevant information about known Biosynthetic Gene Clusters (BGCs), we accessed the publicly available MiBIG database(Medema et al., 2015) (Minimum Information about a Biosynthetic Gene cluster, https://mibig.secondarymetabolites.org/). For our study, we specifically focused on specific class of BGCs i.e. PKS, NRPS, RiPP, Terpenes, and Hybrids of PKS-NRPS. By utilizing the search criteria available in the MiBIG database, we obtained GenBank (gbk) files containing multiple genes that encode specific Natural Products. These GenBank files offered crucial information, including the BGC ID, species name, products encoded, and protein-coding sequences of each gene.

We proceeded to download a substantial number of BGCs for each class, amounting to 605 PKS, 456 NRPS, 344 Hybrid PKS-NRPS, 328 RiPP, and 159 Terpene. Extracting individual genes from the respective GenBank files resulted in a rich set of genetic information for each BGC class, with 12177 genes for PKS, 7068 genes for NRPS, 3096 genes for RiPP, 803 genes for Terpenes, and 5962 genes for Hybrid PKS-NRPS BGCs. To facilitate further analysis, we thoughtfully concatenated all the genes within a single BGC cluster into a single entity and stored the compiled results in a CSV file. This dataset, comprising consolidated genes for each BGC class, served as the input for vectorization and the calculation of physicochemical properties.

2.2. Hardware and Software Environment

In this study, we utilized Python 3.10.10 as the primary programming language for both training and testing procedures. Keras and TensorFlow were employed as essential libraries for building and running the machine-learning models. The coding platform used was Jupyter Notebook. To support our analyses and computations, we incorporated several important Python libraries, such as Numpy, pandas, scikit-learn, genism, and matplotlib.

Windows 8.1 Pro operating system (64-bit) platform was used which was equipped with an Intel (R) core TM i7 4770, running at a clock speed of 3.4 GHz. The system was equipped with a total of 12.00 GB of RAM, out of which 11.9 GB was available for our computations. This hardware setup provided the computational resources necessary to carry out our machine learning and classification tasks effectively.

2.3. Calculation of Physicochemical Properties

The physicochemical properties of proteins play pivotal roles in their structure, function, and interactions within biological systems. We calculated some important properties including molecular weight, Isoelectric point (pI), Charge distribution, Hydrophobicity, Secondary structure predictions, Aromaticity, Instability index, and the molecular extinction coefficient. These properties are calculated using Python libraries such as the BioPython and ProtParam modules.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 October 2023 doi:10.20944/preprints202310.1564.v1

2.4. Sequence to Vector Conversion

In this study, we used NLP techniques to convert the protein sequences to numerical vectors. Specifically, we tried 3 different NLP techniques, such as Count Vector [31], TF-IDF (Term Frequency-Inverse Document Frequency) [32], and Word2Vec to obtain the sequence embeddings. Details of the implementation of each method are provided below in subsequent sub-sections.

2.4.1. Count Vectorizer Method

The Count Vectorizer is a technique commonly used in NLP to convert protein sequences into numerical vectors by counting the occurrence of each amino acid. The principle behind Count Vectorizer is to create a bag-of-words model, where each word in the text is treated as a separate token, and its frequency is counted.

In this implementation, we utilize the scikit-learn library's `CountVectorizer` to transform protein sequences into trigrams. To begin, we import the necessary module, specifically importing the 'CountVectorizer' class from `sklearn.feature_extraction.text`. Next, we initialize the `CountVectorizer` with specific parameters. We set the `analyzer` parameter to ``'char'`, indicating that we want to analyze the protein sequences at the character level. Additionally, we set the `ngram_range` parameter to (3, 3), which instructs the `CountVectorizer` to generate trigrams from the protein sequences and maximum features to 2500. Then the core transformation is performed using the `fit_transform` method of the `CountVectorizer` instance. We pass the list of protein sequences, `protein_sequences`, as input to this method. The `fit_transform` method operates in two steps: "fit" and "transform". During the "fit" step, the `CountVectorizer` learns the vocabulary of the protein sequences. It maps each unique character to a distinct index, building a dictionary that will be used for the subsequent transformation process. The "transform" step then converts the original protein sequences into their trigram representations based on the learned vocabulary. The result of this transformation, containing the trigram feature vectors for the protein sequences, is saved in the variable. These trigram representations are numerical, making them suitable inputs for various machine learning models and enabling further analysis and classification tasks.

2.4.2. TF-IDF (Term Frequency-Inverse Document Frequency) method

TF-IDF is a widely used weighting scheme that assigns weights to words based on their frequency in a document and their rarity in the overall corpus. It represents words as vectors based on their importance in the protein sequence. TF-IDF is effective in capturing the discriminative power of words within a sequence. TFIDF consists of two main components: Term Frequency (TF) and Inverse Document Frequency (IDF).

TF measures the frequency of a word in a document, and is calculated by mathematical formula

$$TF = \frac{number\ of\ times\ a\ trigram\ appears\ in\ a\ sequence}{Total\ number\ of\ trigrams\ in\ a\ sequence}$$

IDF quantifies the rarity of the word across the entire corpus and is given by mathematical equation.

$$IDF = log\ (\frac{total\ number\ of\ sequences}{the\ number\ of\ sequences\ containing\ the\ trigram})$$

The product of TF and IDF yields the TFIDF score for each word, resulting in a document-specific numerical representation. Words that appear frequently in a document but are rare across the corpus receive higher TFIDF scores, making them more discriminative and relevant to the document's content.

In the implementation part, Similar to the CountVec method but here TfidfVectorizer is used to convert protein sequences into trigrams using the TFIDF approach. The `analyzer` parameter is set to ``'char'`, the `ngram_range` parameter is set to ` (3, 3) `, and maximum features to 2500. Then `fit_transform` method of the TfidfVectorizer is applied to convert the protein sequences into their trigram representations, simultaneously learning the vocabulary and transforming the data. The

 doi:10.20944/preprints202310.1564.v1

resulting trigram representations are stored as numerical representations and are utilized for further ML analysis.

2.4.3. Word2Vec

Word2Vec is a word embedding technique that represents words in a continuous vector space, capturing semantic relationships and context. It employs two models: Skip-gram, which predicts context words given a target word, and Continuous Bag of Words (CBOW), which predicts the target word given context words. Both models use neural networks with an embedding layer to convert words into dense vector representations. During training, the models adjust embeddings to maximize prediction accuracy. The resulting word embeddings encode semantic and syntactic similarities between words, making them valuable for various natural language processing tasks such as classification and many more.

In the implementation, the protein sequences are converted into trigrams using a sliding window approach. The resulting trigrams are stored in a list called 'trigrams.' Next, the Word2Vec model is initialized with the 'trigrams' list. The 'sg' parameter is set to 1, indicating the use of the Skip-gram algorithm for training. The 'vector_size' parameter is set to 300, specifying the dimensionality of the resulting trigram embeddings. The 'window' parameter is set to 5, indicating the maximum distance between the current and predicted trigrams in a sentence. The 'min_count' parameter is set to 1, ensuring that even trigrams with a single occurrence are considered during training. Lastly, the 'workers' parameter is set to 5, determining the number of CPU cores used for training. For each sequence in 'trigrams,' the code retrieves the corresponding embeddings from the trained Word2Vec model, ensuring that each trigram has a corresponding embedding. It then calculates the average of these trigram embeddings along each dimension to obtain a single continuous vector, representing the embedding of the entire sequence. These sequence embeddings are stored in the 'embeddings' list. The resulting 'embeddings' list contains the continuous vector representations of the protein sequences, where each vector captures the semantic relationships between trigrams within the respective sequence. These embeddings can be further utilized in classification, to gain insights and make predictions based on the underlying patterns within the protein sequences.

2.5. Comprehensive Feature Matrix

After obtaining sequence embeddings for proteins, we integrated them with the calculated physicochemical properties of the corresponding BGC sequences. The concatenation step effectively combined both the structural and chemical properties of proteins, enhancing the information available for subsequent analysis. This integrated feature representation serves as input for our ML classification task. This approach allows for a more holistic understanding of the proteins within the BGCs and facilitates a more in-depth exploration of their functional roles and potential applications.

2.6. Data Balancing and Scaling

The imbalanced datasets can lead to biased models, as the classifier may prioritize the majority class and neglect the minority classes. To address the challenge of imbalanced datasets, we employed the Synthetic Minority Over-sampling Technique (SMOTE). SMOTE is a popular technique that generates synthetic samples for the minority classes, thus balancing the class distribution [33]. By creating synthetic instances through interpolation between existing samples, SMOTE effectively increases the representation of the minority classes, making the dataset more balanced. Additionally, we performed feature scaling using MinMax Scaler to bring all features to a similar scale, ensuring that no particular feature dominated others during model training. This preprocessing step was essential in achieving reliable and accurate protein sequence classification results.

2.7. Model building and Classification using Machine Learning

This integrated feature matrix serves as the input for our classification task, where we aim to predict and classify the Biosynthetic Gene Clusters (BGCs). In this study, we conducted a

 doi:10.20944/preprints202310.1564.v1

comprehensive assessment of various machine-learning models for the classification of biosynthetic gene clusters (BGCs). To effectively address the class imbalance in our dataset, we performed two sets of experiments: one with SMOTE and the other one without SMOTE. For each of the three NLP (Natural Language Processing) methods employed, namely Word2Vec, TFIDF, and Count Vector, we generated two sets of datasets: one with SMOTE and one without SMOTE. This resulted in a total of six datasets, each incorporating embeddings from the respective NLP method, combined with calculated physicochemical properties. Within each of the six datasets, we trained and evaluated nine different machine-learning models. **Logistic Regression** (LR) is a linear classification algorithm that models the probability of a binary outcome. **Gaussian Naive Bayes** (GNB) is a probabilistic classification method based on Bayes' theorem, assuming conditional independence of features given the class. **Decision Tree** (DT) [34] is a tree-like model useful for classification and regression, though prone to overfitting and benefiting from pruning. **Random Forest (RF)** [35], is an ensemble method that combines multiple decision trees, particularly effective for classification and regression while mitigating overfitting. **K-Nearest Neighbours (KNN)** [36] is an intuitive algorithm that predicts based on the majority class of its nearest neighbours, suitable for various data types. **Support Vector Machine (SVM)** [37] is a powerful algorithm for classification and regression tasks, finding hyperplanes that best separate data classes. **Extreme Gradient Boosting (XGBoost)** [38] is a high-performance gradient-boosting framework used for structured data. **Categorical Boosting (CatBoost)** [39] is a gradient-boosting library adept at handling categorical features without extensive preprocessing. **Artificial Neural Network (ANN)** [40] mimics the human brain's interconnected neurons, employed in tasks like image recognition and natural language processing. Each ML technique has its own strengths and unique approach to pattern recognition and classification. To optimize these models' hyperparameters and achieve the best possible results, we applied a **grid** search method. This systematic approach allowed us to explore various hyperparameter combinations for each model, enabling us to identify the optimal configuration that yielded the highest classification performance.

2.8. Cosine Similarity Calculation

To assess whether three different NLP methods (CountVec, TFIDF, and Word2Vec) effectively capture biological information from the protein sequences or not, we calculated the cosine similarity scores, which is a widely used metric in NLP to assess the similarity between two text vectors, such as protein sequences. Given two protein sequence embeddings, A and B, the similarity between them is calculated using the following formula: Cosine Similarity (A, B) = (A · B) / (||A|| * ||B||). Here, (A · B) represents the dot product of the two embeddings A and B, and (||A|| * ||B||) represents the product of their Euclidean norms (magnitude). The resulting cosine similarity score ranges from -1 to 1, where -1 indicates complete dissimilarity, 0 indicates no similarity, and 1 indicates identical sequences.

For this purpose, we downloaded the top 100 protein sequences from **yeast**, **mouse**, and **E. coli** obtained from the UniProt protein sequence database. Further, we applied each NLP method to convert the protein sequences into numerical representations (vectors). For TFIDF and CountVec, we transformed the sequences into trigrams, while for Word2Vec, we used pre-trained embeddings. After obtaining the vector representations, we calculated the cosine similarity between all pairs of sequences within and across the organisms.

2.9. Performance Evaluation

In order to evaluate the performance of various classifiers used in the current study, we employed various metrics, each offering valuable insights into its effectiveness for the multiclass classification task.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 October 2023 doi:10.20944/preprints202310.1564.v1

1. Stratified k-fold Cross-validation: We utilized stratified k-fold cross-validation, a modification of standard k-fold cross-validation, to ensure balanced class distributions in each fold. This technique allows us to assess the classifier's performance robustly on different subsets of the dataset.

2. Confusion Matrix: The confusion matrix visually represents the performance of the classifier by showing the true and predicted class labels. In contrast to binary classification, a multiclass confusion matrix has neither positive nor negative classifications. For each class individually, the True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) values are determined. A confusion matrix for the multiclass classification problem is shown in Table 3.1. These values allow us to measure the accuracy and errors in the classification process.

Table 1. Sample Confusion Matrix.

Predicate class		Class 1	Class 1	……	Class n
	Class 1	R11	R12	……	R1n
	Class 2	R21	R22	……	R2n
	….	….	….	……	….
	Class n	Rn1	Rn2	……	Rnn

TP: The true value and predicted value should be the same. All diagonal elements will become TP for respective classes, **FP:** The sum of values of corresponding rows except for the TP value, **FN:** The sum of values of corresponding columns except for the TP value, **TN:** The sum of values of all rows and columns except the values of that class that we are calculating for.

3. Recall (Sensitivity): Recall calculates the percentage of truly positive samples correctly predicted by the classifier. It is useful when it is necessary to locate all positive samples or minimize false negatives.

$$\textbf{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \times 100$$

4. Precision: Precision measures the percentage of correctly predicted positive samples out of all samples predicted as positive. It helps to evaluate the classifier's ability to minimize false positives.

$$\textbf{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \times 100$$

5. F1 Score: The F1 score combines precision and recall, providing a harmonic measure of their balance. It indicates the classifier's ability to achieve both high precision and high recall.

$$\textbf{F1 Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

6. Accuracy: The accuracy metric quantifies the fraction of correctly predicted samples among all samples in the dataset. It gives an overall measure of the classifier's performance.

$$\textbf{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \times 100$$

7. Balanced Accuracy: The balanced accuracy metric calculates the arithmetic mean of sensitivity and specificity, providing a balanced measure for imbalanced datasets.

$$\textbf{Balanced accuracy} = \frac{\text{Sensitivity+ Specificity}}{2}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 October 2023 doi:10.20944/preprints202310.1564.v1

8. Matthews Correlation Coefficient (MCC): MCC is a statistical test that evaluates the agreement between actual and predicted values, ranging from -1 to 1.

$$\mathbf{MCC} = \frac{(TP * TN)\text{-}(FN * FP)}{\sqrt{(TP + FP) * (TP + FN) * (TN + TP) * (TN + FN)}}$$

9. Kappa Statistic: The Kappa statistic compares the observed accuracy with the expected accuracy due to random chance, ranging from 0 to 1.

$$\mathbf{Cohen's\ Kappa} = \frac{2 * (TP * TN - FN * FP)}{(TP + FP) * (FP + TN) * (TP + FN) * (FN+ TN)}$$

10. Micro, Macro, and Weighted Precision, Recall, and F1 Score: In multiclass classification, evaluating the performance of a model requires more nuanced metrics beyond simple accuracy. Micro, macro, and weighted precision, recall, and F1 scores are commonly used metrics to provide a comprehensive assessment of the model's performance across multiple classes. Micro-averaging calculates by aggregating the counts of true positives, false positives, and false negatives across all classes before computing the metrics. Micro-averaging is useful when the classes have significantly varying sizes, as it prevents the dominance of larger classes in the evaluation. Macro-averaging calculates for each individual class and then takes the average across all classes. It treats each class equally and provides insight into the model's performance for each class separately. Macro-averaging is useful when all classes are considered equally important and you want to identify classes with lower performance. Weighted averaging is similar to macro-averaging but takes into account the class distribution by assigning different weights to each class based on their prevalence.

11. AUC-ROC Curve: The Receiver Operating Characteristic (ROC) curve and its Area Under the Curve (AUC) measure the classifier's ability to distinguish between classes in binary classification tasks. It summarizes the model's performance across various thresholds.

By employing these performance metrices, we can comprehensively evaluate our classifier's performance and make informed decisions about its effectiveness for the multiclass classification task.

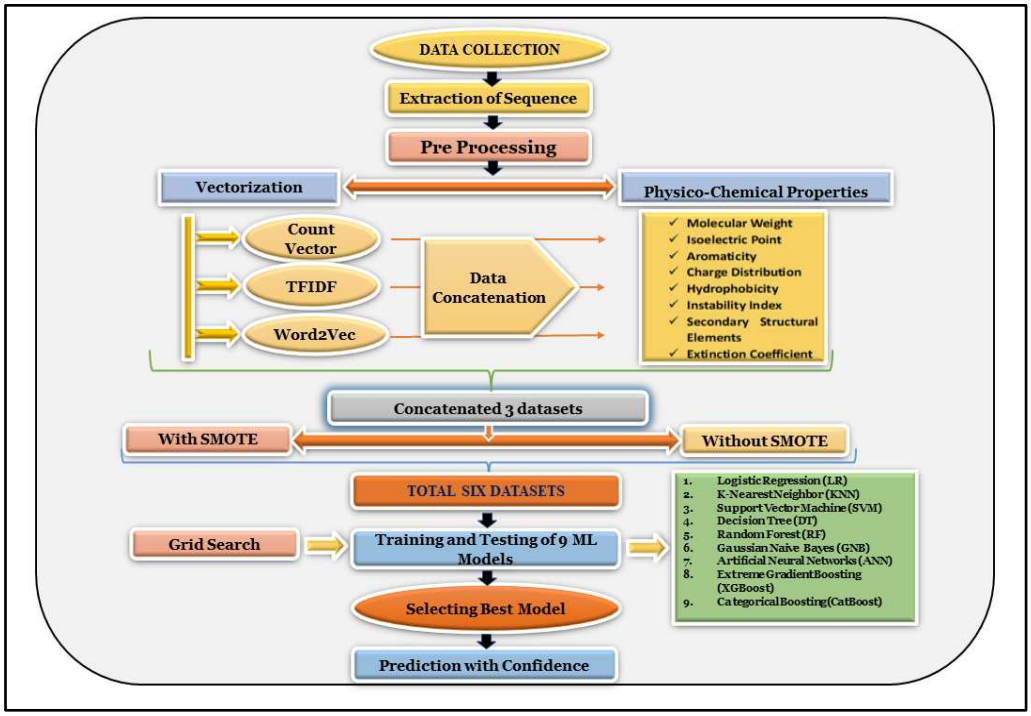

Figure 2. Flowchart of methodology for the identification of BGC in the current study.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 25 October 2023 doi:10.20944/preprints202310.1564.v1

3. Results and Discussion

In this section, we will discuss, the detailed results obtained and discuss their combined results for the proposed methodology.

3.1. Multiclass Classification and Evaluation:

In this study, we conducted an extensive experimental analysis on six different datasets using three distinct Natural Language Processing (NLP) techniques coupled with nine ML models. Specifically, we investigated the impact of the SMOTE on model performance in comparison to scenarios without SMOTE. The models were evaluated based on various performance metrics including Accuracy, Kappa Score, Mathews Score, Balanced Accuracy, Precision, Recall, F1-Score, and Area Under the Curve (AUC) Score. Detailed results are presented in respective tables (**Supplementary Tables S1–S6)**) respectively for easy reference and analysis.

- For Count Vector with SMOTE (Supplementary **Table S1**), it is observed that Logistic Regression (LR), Extreme Gradient Boosting (XGBoost), and Artificial Neural Network (ANN) achieved notable performance with Accuracy scores of 0.93, 0.91, and 0.89 respectively. These models also exhibited high Precision, Recall, and F1-Score values, indicating their effectiveness in classification tasks.
- Comparatively, when Count Vector was used without SMOTE (Supplementary **Table S2**), the performance of models slightly decreased across all metrics. Notably, LR, XGBoost, and ANN still demonstrated commendable results with Accuracy scores of 0.82, 0.80, and 0.76 respectively.
- The results obtained for TFIDF with SMOTE (Supplementary **Table S3**), indicate that LR, SVM, and Random Forest (RF) models exhibited strong performance with Accuracy scores of 0.94, 0.90, and 0.88 respectively. These models also demonstrated high Precision, Recall, and F1-Score values, suggesting their suitability for the given classification tasks.
- When TFIDF was employed without SMOTE (Supplementary **Table S4**), LR, XGBoost, and SVM models showed promising results with Accuracy scores of 0.77, 0.80, and 0.77 respectively. These models maintained competitive performance across various evaluation metrics.
- Additionally, results obtained for Word2Vec with and without SMOTE (Supplementary **Tables S5 and S6**), demonstrated that LR, Random Forest, and Extreme Gradient Boosting consistently yielded favorable outcomes across the datasets, emphasizing their robustness and applicability in the context of this study.

Further, results are visualized through various graphs for different evaluation measures i.e., confusion metrics (**Supplementary Figures S1, S3, S5, S7, S9, and S11**) and ROC curve (**Supplementary Figures S2, S4, S6, S8, S10, and S12**) for each dataset.

3.2. Cosine similarity calculation

From **Table 2** it has been concluded that TFIDF and CountVec demonstrated high similarity, indicating their effectiveness in capturing biological information within organisms. On the other hand, the low similarity of Word2Vec suggests that it may not effectively capture specific biological features of protein sequences.

Table 2. Shows the average cosine similarity of 3 different NLP techniques.

Sl. No.	NLP Methods	Average Cosine Score
1	TFIDF	0.93
2.	Count Vector	0.90
3.	Word2Vec	0.33

Based on the presented results, both tables and figures, it is evident that the utilization of TFIDF with SMOTE in conjunction with Logistic Regression (Refer **Supplementary Table S3**) yields superior performance. This conclusion is substantiated by the calculated Cosine Similarity Scores,

 doi:10.20944/preprints202310.1564.v1

which indicates a high similarity of approximately 0.93 for TFIDF. Hence, we can say that TFIDF is capturing better biological information and rightly obtained results are also pointing the same.

4. Summary and Conclusion

We have developed a comprehensive approach to tackle the complex task of identifying biosynthetic gene clusters (BGCs) responsible for producing bioactive compounds. This method combines NLP-derived sequence features with physicochemical properties of proteins. By using advanced machine learning (ML) techniques, we have proposed a robust methodology for classifying BGCs. With the incorporation of diverse data types, including sequence information and physicochemical attributes, along with the implementation of data balancing, has proven to be a substantial asset. These measures enable the creation of a more dependable and extensive dataset for BGC classification. Notably, the integration of these elements, coupled with SMOTE, has not only boosted the accuracy of our model but also furnished valuable insights into the intricate nature of BGCs.

Identifying biosynthetic gene clusters (BGCs) poses challenges, particularly due to class imbalances and distinguishing hybrid PKS-NRPS clusters from the PKS and NRPS classes. To overcome this, we implemented the Synthetic Minority Over-sampling Technique (SMOTE). This technique notably bolstered our model's capability to handle the minority classes, particularly the intricate PKS-NRPS hybrids. This improvement ensures the effectiveness and reliability of our approach, even when certain BGC classes are underrepresented in the dataset.

Our research outcomes underscore the effectiveness of our TFIDF approach in conjunction with SMOTE-enhanced Logistic Regression (LR) machine learning models for accurately categorizing biosynthetic gene clusters (BGCs) into their respective classes. The machine learning model's contribution was substantial, elevating the overall robustness and accuracy of BGC classification to an impressive level. The model achieved outstanding accuracy, precision, recall, and F1 score metrics, all registering at an impressive 96%. Moreover, the model exhibited a high Area Under the Curve (AUC), affirming its remarkable ability to distinguish between various BGC classes with exceptional accuracy. The amalgamation of physicochemical properties alongside sequence data has proven to be a pivotal advantage, enabling the creation of a more dependable and comprehensive dataset for BGC classification, with an AUC of 0.9912.

In comparison with existing tools BGC identification tools, our model does not rely on HMMs or rule-based algorithms, avoiding the rediscovery problem of antiSMASH and enabling comprehensive classification, including RiPP BGCs. Unlike DeepBGC, our method does not depend on existing Pfam families, providing a more robust and inclusive BGC identification. Moreover, our model efficiently handles high-dimensional by selecting maximum features during the vectorization of protein sequences only, which significantly reduced the code complexity and improved computational efficiency compared to ML Miner's approach. Moreover, our model achieved a higher accuracy of 97% for four classes, showcasing its superior performance compared to ML Miner. It accurately identifies hybrids and achieves higher accuracy (97%) compared to ML Miner's 92%.

In conclusion, our research offers several advantages over the ML Miner strategy, including hybrid identification, improved computational efficiency through dimensionality reduction, the integration of physicochemical properties, and higher classification accuracy. These advancements enhance the potential of our approach to bioactive compound discovery and drug development research. Our model achieved impressive results demonstrating its effectiveness in accurately classifying BGCs into five different classes, including challenging hybrids (PKS-NRPS hybrids).

In conclusion, our proposed approach introduces a novel and comprehensive method for BGC classification, effectively addressing the challenges posed by their diverse and complex nature. The TFIDF with SMOTEs Logistic Regression ML model, in conjunction with the integration of physicochemical properties demonstrated superior performance over existing tools such as ML Miner. The high accuracy, Kappa, Balanced Accuracy, F1 Score, Precision, Recall, and AUC achieved by our model highlight its potential for bioactive compound discovery and drug development research, contributing to the advancement of Natural Product biosynthesis and facilitating the

 doi:10.20944/preprints202310.1564.v1

exploration of novel bioactive compounds with therapeutic applications. Additionally, the significance of Cosine similarity in reinforcing the reliability of our classification model indicates a high degree of sequence similarity within specific BGC classes, further endorsing the strength of our approach.

5. Future Scope

For future work, one can strengthen the BGCs classification through more robust ensemble models and incorporating diverse datasets, by exploring dimensionality reduction techniques and advanced NLP methods to improve processing efficiency, and accuracy at the same time by reducing the false positives. Exploring novel drug targets within identified Natural Products presents an innovative dimension in the research of Natural Products, offering potential breakthroughs in therapeutics. To gain a comprehensive ecosystem understanding, researchers can conduct functional studies, metabolomics analyses, and targeted Natural Product characterizations, contributing to a deeper comprehension of microbial communities and their intricate dynamics. Integrating multi-omics data, including metagenomics, metatranscriptomics, metaproteomics, and metabolomics, provides a nuanced portrayal of microbial communities and hence their Natural Product dynamics. Further, the prediction of chemical structures from sequence information is a promising avenue for unveiling the vast chemical diversity of Natural Products. Interdisciplinary collaboration with experts from microbiology, chemistry, and environmental science will enrich insights and ensure practical relevance. These explorations synergize to advance our knowledge of microbial ecosystems, Natural Product diversity, and their potential applications in fields like medicine and biotechnology.

List of Abbreviations

AUC	Area Under ROC Curve
BGC	Biosynthetic Gene Clusters
NLP	Natural Language Processing
TFIDF	Term Frequency and Inverse Document Frequency
ML	Machine Learning
PKS	Polyketide Synthases
NRPS	Non-Ribosomal Peptide Synthetases
RiPP	Ribosomally synthesized and Post translationally modified Peptides
SMOTE	Synthetic Minority Over-sampling Technique

References

1. Katz, L., & Baltz, R. H. (2016). Natural Product discovery: past, present, and future. *Journal of Industrial Microbiology and Biotechnology*, *43*(2-3), 155-176.
2. Zhang, M. M., Qiao, Y., Ang, E. L., & Zhao, H. (2017). Using Natural Products for drug discovery: the impact of the genomics era. *Expert opinion on drug discovery*, *12*(5), 475-487.
3. Zarins-Tutt, J. S., Barberi, T. T., Gao, H., Mearns-Spragg, A., Zhang, L., Newman, D. J., & Goss, R. J. M. (2016). Prospecting for new bacterial metabolites: a glossary of approaches for inducing, activating and upregulating the biosynthesis of bacterial cryptic or silent Natural Products. *Natural Product reports*, *33*(1), 54-72.
4. Cane, D. E., Walsh, C. T., & Khosla, C. (2000). Harnessing the Biosynthetic Code: Combinations, Permutations, and Mutations. Science, 282(5386), 63–68.
5. Wambo, Paul A. ML-Miner: A Machine Learning Tool Used for Identification of Novel Biosynthetic Gene Clusters. Diss. Université d'Ottawa/University of Ottawa, 2022.
6. Winn, M., Fyans, J. K., Zhuo, Y., & Micklefield, J. (2016). Recent advances in engineering nonribosomal peptide assembly lines. *Natural Product reports*, *33*(2), 317-347.
7. Rudolf, J. D., & Chang, C. Y. (2020). Terpene synthases in disguise: enzymology, structure, and opportunities of non-canonical terpene synthases. *Natural Product reports*, *37*(3), 425-463.
8. Hetrick, K. J., & van der Donk, W. A. (2017). Ribosomally synthesized and post-translationally modified peptide Natural Product discovery in the genomic era. *Current opinion in chemical biology*, *38*, 36-44.

9. Zazopoulos, E., Huang, K., Staffa, A., Liu, W., Bachmann, B. O., Nonaka, K., ... & Farnet, C. M. (2003). A genomics-guided approach for discovering and expressing cryptic metabolic pathways. *Nature biotechnology*, *21*(2), 187-190.
10. Dewick, P. M. (2002). Medicinal Natural Products: a biosynthetic approach. John Wiley & Sons.
11. Medema, M. H., Blin, K., Cimermancic, P., de Jager, V., Zakrzewski, P., Fischbach, M. A., ... & Takano, E. (2011). antiSMASH: rapid identification, annotation, and analysis of secondary metabolite biosynthesis gene clusters in bacterial and fungal genome sequences. Nucleic Acids Research, 39(Supplement_2), W339-W346.
12. Hannigan, G. D., Prihoda, D., Palicka, A., Soukup, J., Klempir, O., Rampula, L., ... & Bitton, D. A. (2019). A deep learning genome-mining strategy for biosynthetic gene cluster prediction. *Nucleic acids research*, *47*(18), e110-e110.
13. Skinnider MA, Dejong CA, Rees PN, Johnston CW, Li H, Webster ALH, et al. (2015). Genomes to Natural Products PRediction Informatics for Secondary Metabolomes (PRISM). Nucleic Acids Res. 43(20):9645-62.
14. Skinnider MA, Merwin NJ, Johnston CW, Magarvey NA. (2017). PRISM 3: expanded prediction of Natural Product chemical structures from microbial genomes. Nucleic Acids Res. 45(W1): W49-W54.
15. Wambo, P. A. (2022). ML-Miner: A Machine Learning Tool Used for Identification of Novel Biosynthetic Gene Clusters (Doctoral dissertation, Université d'Ottawa/University of Ottawa).
16. Kautsar, S. A., Blin, K., Shaw, S., Navarro-Muñoz, J. C., Terlouw, B. R., Van Der Hooft, J. J., ... & Medema, M. H. (2020). MIBiG 2.0: a repository for biosynthetic gene clusters of known function. *Nucleic acids research*, *48*(D1), D454-D458.
17. Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. Information processing & management, 24(5), 513-523.
18. Reference: K. Sparck Jones. (1972). "A statistical interpretation of term specificity and its application in retrieval." Journal of Documentation, 28(1), 11-21.
19. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, *26*.
20. Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, *16*, 321-357.
21. Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2003). KNN model-based approach in classification. In On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings (pp. 986-996). Springer Berlin Heidelberg.
22. Hosmer Jr, David W., Stanley Lemeshow, and Rodney X. Sturdivant. *Applied logistic regression*. Vol. 398. John Wiley & Sons, 2013.
23. Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2003). KNN model-based approach in classification. In On the Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings (pp. 986-996). Springer Berlin Heidelberg.
24. Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*, 273-297.
25. Lewis, D. D. (1998, April). Naive (Bayes) at forty: The independence assumption in information retrieval. In European conference on machine learning (pp. 4-15). Berlin, Heidelberg: Springer Berlin Heidelberg.
26. Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). *Classification and regression trees*. CRC press.
27. Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5-32.
28. Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).
29. Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: unbiased boosting with categorical features. *Advances in neural information processing systems*, *31*.
30. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.
31. Leslie, C., Eskin, E., & Noble, W. S. (2001). The spectrum kernel: A string kernel for SVM protein classification. In *Biocomputing 2002* (pp. 564-575).
32. Dunning, T. (1994). Accurate methods for the statistics of surprise and coincidence. *Computational linguistics*, *19*(1), 61-74.
33. Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, *16*, 321-357.
34. Loh, W. Y. (2011). Classification and regression trees. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, *1*(1), 14-23.
35. Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5-32.
36. Omohundro, S. M. (1989). *Five balltree construction algorithms* (pp. 1-22). Berkeley: International Computer Science Institute.
37. Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*, 273-297.

 doi:10.20944/preprints202310.1564.v1

38. Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785-794).
39. Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: unbiased boosting with categorical features. *Advances in neural information processing systems*, *31*.
40. Rosenblatt, F. (1962). Principles of neurodynamics: Perceptrons and the theory of brain mechanisms (Vol. 55). Washington, DC: Spartan books.