SCIENTIFIC REPORTS
natureresearch

OPEN

Antimicrobial Resistance Prediction for Gram-Negative Bacteria via Game Theory-Based Feature Evaluation

Received: 9 May 2019
Accepted: 13 September 2019
Published online: 09 October 2019

Abu Sayed Chowdhury[1], Douglas R. Call[1,2] & Shira L. Broschat[1,2,3]

The increasing prevalence of antimicrobial-resistant bacteria drives the need for advanced methods to identify antimicrobial-resistance (AMR) genes in bacterial pathogens. With the availability of whole genome sequences, best-hit methods can be used to identify AMR genes by differentiating unknown sequences with known AMR sequences in existing online repositories. Nevertheless, these methods may not perform well when identifying resistance genes with sequences having low sequence identity with known sequences. We present a machine learning approach that uses protein sequences, with sequence identity ranging between 10% and 90%, as an alternative to conventional DNA sequence alignment-based approaches to identify putative AMR genes in Gram-negative bacteria. By using game theory to choose which protein characteristics to use in our machine learning model, we can predict AMR protein sequences for Gram-negative bacteria with an accuracy ranging from 93% to 99%. In order to obtain similar classification results, identity thresholds as low as 53% were required when using BLASTp.

Bacteria can cause bloodstream infections, and with the increasing prevalence of antimicrobial resistance (AMR) in bacteria treatment can become complicated[1–7]. AMR results in increased mortality and an increase in the duration of hospitalization. Every year, millions of people in the United States are infected by AMR bacteria, and thousands of people die[8,9]. Hence, accurate identification of AMR in bacteria is essential for the proper administration of appropriate antibacterial agents. To detect AMR in bacteria, *in vitro* cultures are used to monitor the growth of bacteria for different concentrations of drugs and may require several days to obtain accurate antibiotic susceptibility results[10]. In addition, many bacteria cannot be cultured, and a large number of these are becoming available via metagenomic studies[11,12].

With breakthroughs in whole genome sequencing (WGS) method, it is possible to apply sequence alignment approaches such as best-hit methods to identify AMR genes using sequence similarity in public databases[4,13,14]. These methods show good performance in identifying known and highly conserved AMR genes and produce small number of false positives, *i.e*., detecting non-AMR genes as AMR genes[15]. However, they may not be able to to find AMR sequences that have high dissimilarity with known AMR genes, producing unacceptable numbers of false negatives[13,16,17]. A machine learning approach can be used as an alternative solution for identifying putative AMR genes. To train a machine learning algorithm to detect AMR sequences, training data are needed in the form of protein sequences for known AMR genes (positive training data) and protein sequences that are known not to be AMR genes (negative training data). From these training data, we must determine what protein characteristics distinguish AMR genes from non-AMR genes. These characteristics are known as features. Numerous features exist for protein sequences, and a goal of any accurate machine learning model is to determine which features provide the most useful information. Recently, two studies have proposed machine learning approaches for predicting AMR genes. Arango-Argoty *et al*. discuss a deep learning approach—DeepARG[18] to identify novel antimicrobial resistance genes from metagenomic data. DeepARG employs an artificial neural network

[1]School of Electrical Engineering and Computer Science, Washington State University, P.O. Box 642752, Pullman, Washington, USA. [2]Paul G. Allen School for Global Animal Health, Washington State University, P.O. Box 647090, Pullman, Washington, USA. [3]Department of Veterinary Microbiology and Pathology, Washington State University, P.O. Box 647040, Pullman, Washington, USA. Correspondence and requests for materials should be addressed to A.S.C. (email: abu.chowdhury@wsu.edu)

AMR	Oversampling			Undersampling		
	$\delta=1$	$\delta=2$	$\delta=3$	$\delta=1$	$\delta=2$	$\delta=3$
acetyltransferase (*aac*)	6 (0.97)	6 (0.97)	6 (0.97)	5 (0.97)	5 (0.97)	5 (0.97)
β-lactamase (*bla*)	15 (1)	19 (1)	18 (1)	9 (0.97)	9 (0.97)	11 (0.97)
dihydrofolate reductase (*dfr*)	5 (1)	5 (1)	5 (1)	18 (0.96)	28 (1)	25 (1)

Table 1. Classification performance for different δ values (corresponding classification accuracy in parentheses).

based classifier, taking into account the similarity distribution of sequences to all known AMR genes. The other approach is the pairwise comparative modelling (PCM)[19] which leverages protein structure information for AMR sequence identification. PCM builds two structural models for each candidate sequence with respect to AMR and non-AMR sequences, and a machine learning model is applied to find the best structural model for determining whether the sequence belongs to an AMR or non-AMR family. In contrast to these earlier works, we consider all possible candidate features for protein sequences based on the composition, physicochemical, evolutionary, and structural characteristics of protein sequences whose sequence identity ranges between 10% and 90%.

The earlier works discussed above did not apply any feature reduction strategy to find the most relevant, non-redundant and interdependent features. Identifying important features from a set of features to attain high classification accuracy is a challenging problem in machine learning because irrelevant or redundant features can compromise accuracy. Several feature selection techniques can be used to obtain an optimal feature set, and these are broadly classified into three categories: embedded, wrapper, and filter approaches. Both embedded and wrapper methods[20–22] are tightly coupled with a particular learning algorithm, and both achieve good classification accuracy. However, these approaches have a high computational overhead and less generalization of features. Alternatively, filter methods measure feature relevance by considering the intrinsic properties of the data[23–25]. In addition, the filter approach has a lower computational cost than the other methods, and it facilitates comparable accuracy for most classifiers[26]. Thus, we only considered filter method for our feature selection approach.

Most filter methods reject features that are poorly predictive when used alone, even though they may work well when combined with other variables[26,27]. In contrast, in this paper we introduce a game theoretic dynamic weighting based feature evaluation (GTDWFE) approach in which features are selected one at a time based on relevance and redundancy measurements with dynamic re-weighting of candidate features based on their interdependency with the current selected features. Re-weighting is determined using a Banzhaf power index[28], and features are not necessarily rejected because they are poorly predictive as single variables. Instead we consider how features work together as a whole using a game theory approach. In simple terms, game theory is the study of mathematical models for determining how the behavior of one participant depends on the behavior of other participants. We consider features from the protein sequences—both AMR and non-AMR—of the Gram-negative bacterial genera *Acinetobacter*, *Klebsiella*, *Campylobacter*, *Salmonella*, and *Escherichia* for acetyltransferase (*aac*), β-lactamase (*bla*), and dihydrofolate reductase (*dfr*). Next we apply our game theory approach to select a small subset of features from the bacterial protein sequences, and finally we utilize this small feature subset to predict AMR genes using a support vector machine (SVM)[29]. We use protein sequences from different Gram-negative bacteria *Pseudomonas* spp., *Vibrio* spp., and *Enterobacter* spp. to test our classifier. We also make a performance comparison between our classifier and BLASTp.

Results

Interdependent group size based comparative analysis.

We compared the classification performance of our method using an SVM with an interdependent group size of $\delta \in [1, 3]$ where δ is used in the computation of the Banzhaf power index. Using the top k features and 30 different feature subsets ($1 \le k \le 30$) of *Acinetobacter*, *Klebsiella*, *Campylobacter*, *Salmonella*, and *Escherichia* and dividing the dataset into 70%/30% training/test samples, we tuned the SVM based on an equal number of positive and negative samples, to determine the best parameters for the SVM models. We selected the radial basis function kernel for each SVM[30,31] but different C and γ values for each feature subset. C is used to control the cost of misclassification in the SVM, and γ is the kernel parameter. We used the resulting models trained using 70% of the dataset to identify putative AMR genes for the remaining data set. The numbers of the best feature subsets using oversampling and undersampling techniques for different δ values are shown in Table 1 where the classification accuracy achieved for each respective best feature subset is shown in parentheses. Oversampling and undersampling are methods in data analytics for balancing sets of data for which there are inherently more samples of one class than another. For this work, the number of positive samples (AMR) is smaller than the number of negative samples (non-AMR). To compensate, we duplicated the positive training samples (oversampling) and we removed some of the negative training samples (undersampling) to achieve balanced datasets.

The classification accuracies achieved for *aac*, *bla*, and *dfr* vary from 96% to 100%. We also determined $\delta = 3$ to be the overall best interdependent group size, so we set δ to 3 for the remainder of our analyses. The C and γ parameter values for the SVM radial model for selecting each best feature subset with $\delta = 3$ are listed as a supplementary table (Table S1).

Feature selection method comparative analysis.

To assess the performance of our GTDWFE feature evaluation method, we compared it with a popular feature selection algorithm — RReliefF[32]. RReliefF is an updated version of Relief and ReliefF[33,34], a filter-based method that uses distance between instances to find the

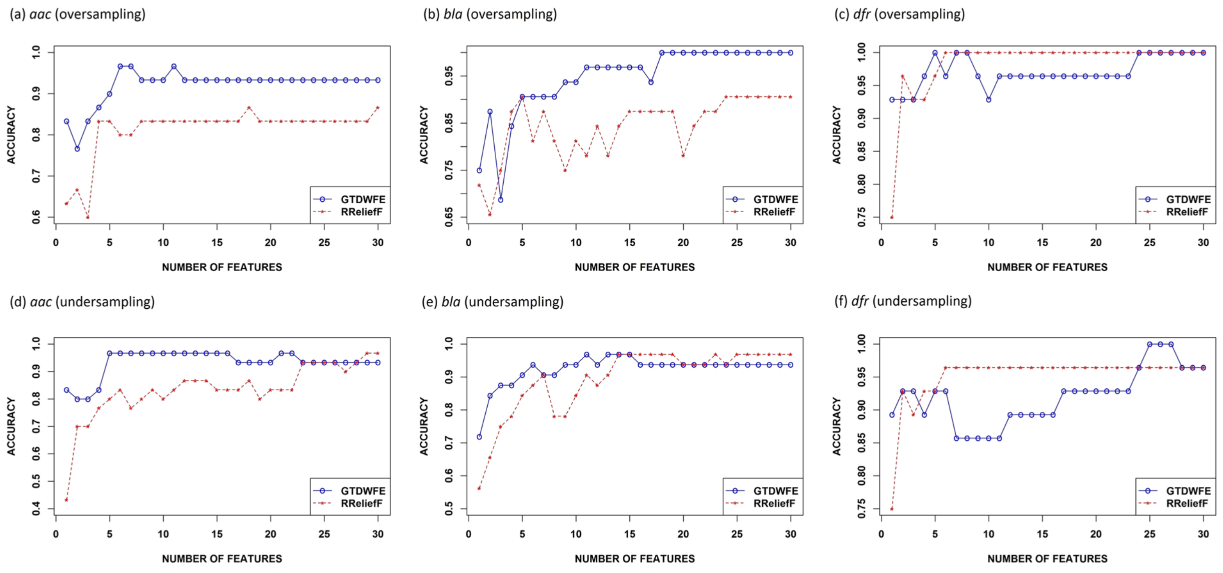

Figure 1. Comparison between GTDWFE and RReliefF accuracies for oversampling and undersampling. Accuracies are given as a function of the number of features used.

relevance weight of each feature to rank the features. For RReliefF, we considered 5 neighbors and 30 instances as suggested in a previous study[35]. We used $\delta = 3$ in our feature selection approach. Comparisons of the results for the two methods are shown in Fig. 1 for oversampling and undersampling. As can be seen in these results, the maximum accuracies our GTDWFE achieved *w.r.t.* the number of features are better than those of RReliefF for all cases. Note that in some cases, RReliefF achieved equal maximum accuracies as GTDWFE method, but the latter required fewer number of features.

Identification of antimicrobial-resistance proteins from *Pseudomonas*, *Vibrio*, and *Enterobacter*. To acquire a better understanding of whether the GTDWFE method can identify putative AMR genes in Gram-negative bacteria, we trained SVM classifiers using 100% of the positive and negative samples for *Acinetobacter*, *Klebsiella*, *Campylobacter*, *Salmonella*, and *Escherichia*. We used the same features obtained previously for $\delta = 3$ to train the SVM model. We then tested the SVM classifiers using the same features for positive and negative samples for *Pseudomonas*, *Vibrio*, and *Enterobacter*. Importantly, for acetyltransferase, we used eight non-AMR samples of acetyltransferase to ascertain whether the SVM classifier was able to distinguish between resistant and non-resistant samples of acetyltransferase. The confusion matrices for both oversampling and undersampling cases are shown in Fig. 2. In a confusion matrix, 'Positive' and 'Negative' indicate AMR (positive) and non-AMR (negative) classes, respectively, and falling diagonal entries indicate correctly identified instances. For oversampling, the GTDWFE method achieved accuracies of 0.93, 0.99, and 0.97 for *aac*, *bla*, and *dfr*, respectively. Moreover, 6 of the 8 non-AMR samples of acetyltransferase were correctly predicted to be negative. For undersampling, the GTDWFE approach had accuracies of 0.91, 0.99, and 0.97 for *aac*, *bla*, and *dfr*, respectively. Of the 8 non-AMR samples, 5 were correctly predicted to be negative samples. Here, as for our test cases in the previous section, the GTDWFE method gives better results when oversampling is used.

Based on the results above, we conclude that our game theory approach for determining protein features for use in a machine learning algorithm can be used with high accuracy to predict AMR in Gram-negative bacteria. We used training data from five different Gram-negative bacterial genera and predicted AMR in three different Gram-negative bacterial genera. The results based on oversampling were the most accurate with 93% accuracy for acetyltransferase resistance, 97% accuracy for dihydrofolate reductase resistance, and 99% accuracy for β-lactamase resistance. The performance of our game theory method was also compared with the BLASTp results considering default parameter settings (https://blast.ncbi.nlm.nih.gov/Blast.cgi?PAGE=Proteins). The outcomes shown in Fig. 3 are the number of matched AMR protein sequences as a function of percent identity for *Pseudomonas*, *Vibrio*, and *Enterobacter* using the AMR genes from *Acinetobacter*, *Klebsiella*, *Campylobacter*, *Salmonella*, and *Escherichia*. For example, considering percent identity $\geq$90 for a sequence from *Pseudomonas*, *Vibrio*, or *Enterobacter* to be matched (true positive) with an AMR sequence from *Acinetobacter*, *Klebsiella*, *Campylobacter*, *Salmonella*, or *Escherichia*, we obtain eight true positives out of ten *aac* sequences, 22 true positives out of 43 *bla* sequences, and eight true positives out of eight *dfr* sequences. The results for *dfr* are much better, but this may in part be due to the limited overall diversity of available *dfr* sequences. Note that the percent identity threshold has to be as low as 41% to obtain accurate results for *aac* and *bla*, but this results in an increased number of false positives. As an example, when the percent identity threshold is set to 41%, six out of eight histone acetyltransferases are incorrectly detected as AMR samples, indicating a large false positive rate. Therefore, using an appropriate threshold setting for BLASTp compromises the accuracy of the results. To obtain the equal true positives as for our oversampling cases (Fig. 2), the percent identity threshold of BLASTp need to be 67% for *aac* and 53% for *bla*; however, these sequence identity thresholds produce three false positives for *aac* (two of

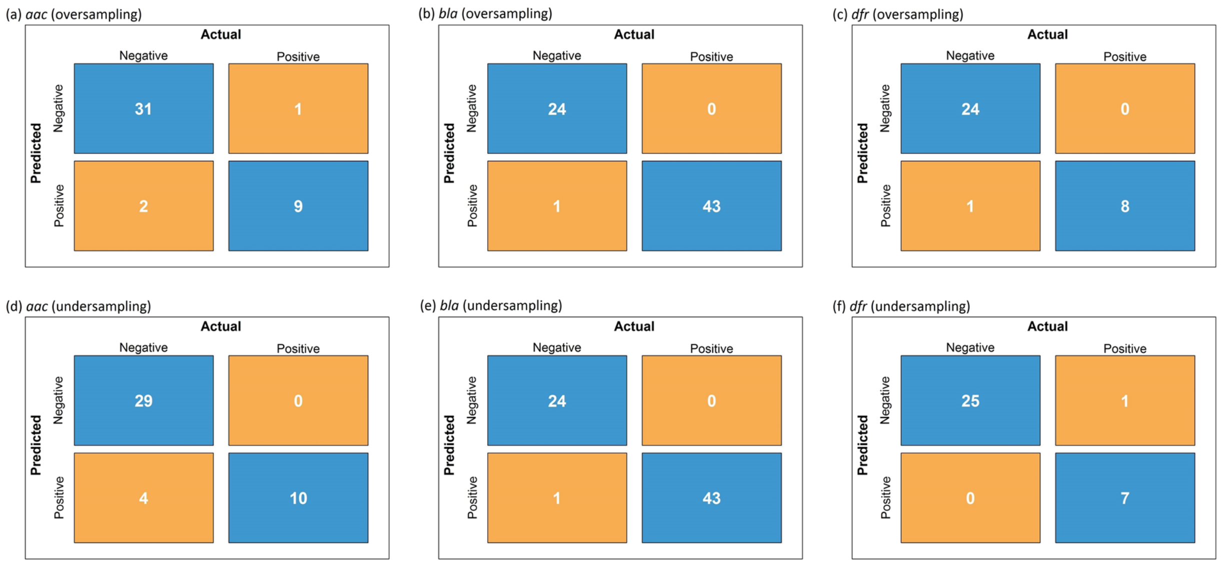

Figure 2. Confusion matrices for oversampling and undersampling.

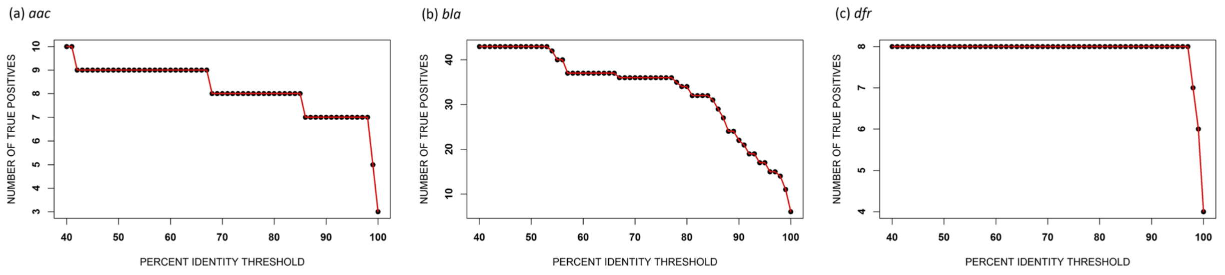

Figure 3. Identification of AMR sequences in *Pseudomonas*, *Vibrio*, or *Enterobacter* using BLASTp as a function of percent identity using AMR sequences from *Acinetobacter*, *Klebsiella*, *Campylobacter*, *Salmonella*, and *Escherichia*.

them are histone acetyltransferases) and six false positives for *bla*. Thus, false positive rates using BLASTp classification for *aac* and *bla* are still high compared to our GTDWFE algorithm. Our machine learning approach provides greater accuracy than conventional methods for highly diverse protein sequences.

Discussion

In this paper we presented a machine learning method for prediction of antimicrobial-resistance genes for three antibiotic classes. The strength of a machine learning algorithm is that it uses features based on the structural, physicochemical, evolutionary, and compositional properties of protein sequences rather than simply their sequence similarity. The novel game theory approach we used to determine protein features for our machine learning algorithm has not been used previously for such a purpose and is especially powerful because features are chosen on the basis of how well they work together as a whole to identify putative antimicrobial-resistance genes by taking into account both the relevance and interdependency of features. As such, we were able to use protein sequences to train the machine learning algorithm using functionally-equivalent amino acid sequences with shared identity that ranged from 10% to 90%. The algorithm was then able to correctly identify genes from an independent data set with 93% to 99% accuracy. The only way this can be achieved by means of a best-hit approach such as BLASTp is by considering sequence matches with as low as 53% similarity. Compared to our approach, this then leads to a greater number of false positives, that is, sequences incorrectly identified as antimicrobial-resistance genes.

Our work included collection of resistance and non-resistance protein sequences, feature extraction, feature evaluation for dimension reduction, handling of imbalanced data sets, and comparison of our method with an existing feature selection approach. The RReliefF algorithm for selecting features is well-known for its accuracy and ability to rank features by their importance, but it does not account for feature interdependence. This was made clear by comparison between results obtained using the game theory algorithm, GTDWFE, and RReliefF. GTDWFE achieved the highest accuracy for all cases using fewer features than RReliefF because of its reduction in irrelevant and/or redundant information. The results of our approach using oversampling were better than those using undersampling.

With growth in both antimicrobial resistance and the number of sequenced genomes available, implementation of machine learning models for accurate prediction of AMR genes represents a significant development toward new and more accurate tools in the field of predictive antimicrobial resistance. In future work, we will create a user-friendly and publicly available program for predicting AMR in bacteria based on the method presented in this paper.

Methods

Data collection. Amino acid sequences for antimicrobial-resistance genes were retrieved from the Antibiotic Resistance Genes Database (ARDB)[36], and non-AMR sequences were obtained from the Pathosystems Resource Integration Center (PATRIC)[37]. A BLASTp search using default parameter settings was performed to find all matching sequences. Initial AMR sequences for the Gram-negative bacteria *Acinetobacter* spp., *Klebsiella* spp., *Campylobacter*spp., *Salmonella* spp., and *Escherichia* spp. numbered 387 for *aac*, 1113 for *bla*, and 804 for *dfr*; there were 159 non-AMR sequences (73 essential genes and 86 histone acetyltransferesases[38]) randomly chosen. Because of the number of duplicate sequences, we used CD-HIT[39,40] to find the unique sequences. Sequences having $\geq$90% similarity were removed for further consideration. The final number of unique sequences obtained were 33 *aac*, 43 *bla*, and 28 *dfr* AMR sequences and 71 non-AMR sequences (64 essential genes and 7 histone acetyltransferases). This data set was used as the training and test set for our model. The histone acetyltransferases together with the essential sequences were used as negative training data only for the *aac* classifier. For *bla* and *dfr*, only essential sequences were used as negative training data. In addition to the training/test data set, 199 *aac*, 588 *bla*, 66 *dfr* AMR sequences and 82 non-AMR sequences (35 essential genes and 47 histone acetyltransferases) for the Gram-negative bacteria *Pseudomonas* spp., *Vibrio* spp., and *Enterobacter* spp. were collected from the data sources indicated above. After application of CD-HIT, 10 *aac*, 43 *bla*, and 8 *dfr* AMR sequences and 33 non-AMR sequences (25 essential genes and 8 histone acetyltransferases) were retained. These were used to test the accuracy of the final classifier. Again, histone acetyltransferases were used only in the *aac* model. Note that sequence similarity for the AMR sequences could be quite low.

Protein features. A literature search was used to identify the composition, physicochemical characteristics, and secondary structure properties of protein sequences[41–46]. As a result of this search, we created 20D feature vectors based on amino acid composition with each of the 20 feature values in a vector representing one of the 20 amino acids. Next the composition, transition, and distribution (CTD) model proposed by Dubchak *et al.*[47,48] was used to retrieve global physicochemical features from protein sequences. The CTD model results in a 3D feature vector for composition, a 3D feature vector for transition, and a 15D feature vector for distribution. As there 8 physicochemical amino acid properties, the CTD paradigm provides a total of $(3 + 3 + 15) \times 8 = 168$ features. We obtained evolutionary-relevant features using a position-specific scoring matrix (PSSM). After producing a PSSM for a protein sequence by applying PSI-BLAST[49], we computed transition scores between adjacent amino acids which resulted in a 400D feature vector for each protein sequence.

Finally, features were obtained for the secondary structure of proteins which provides relevant information in protein fold recognition. PSIPRED[50] was applied to sequences to predict their secondary structures. These were used as described in previous studies[43,44,51–53] to obtain our secondary structure features. Location-oriented features were produced from the spatial arrangements of the α-helix, β-strand, γ-coil states. Normalized maximum spatially consecutive states in the secondary structure sequences were also calculated. Additionally, we retrieved features from segment sequences by disregarding the coil portions in the secondary structure. In such a way, a total of six features were generated from the protein structure information.

Three global information features were generated from the structure probability matrix (SPM) produced by PSIPRED. Local information features were acquired by dividing the SPM into δ submatrices, each with $\lfloor \frac{n}{\delta} \rfloor \times 3$ entries. By selecting $\delta = 8$, we generated 3D features for a particular sub-matrix using the same approach considered for the generation of global information features. Hence, we obtained $3 \times 8 = 24$ local information features. In total, $3 + 24 = 27$ features with global and local information were generated.

By combining all the features, we obtained a 621D high-dimensional feature vector. Detailed descriptions of all the extracted candidate features together with the formulas used to calculate values can be found in our previous work[54]. The objective of this work was to then reduce the dimension of our feature vector in such a way as to produce accurate machine learning predictions for AMR.

Feature evaluation. We adopted a feature evaluation method to select a small number of features based on a relevance, redundancy, and interdependency estimation of all the features. In our approach, we calculated the weight of a feature based on the features selected earlier, and weight readjustment of a feature was performed dynamically when a new selected feature was added to the previously selected feature subset. Here, the weight of a feature actually resembles the interdependence relationship with features previously selected. The details of our feature selection approach called the game theoretic dynamic weighting based feature evaluation (GTDWFE) algorithm are presented in Algorithm 1.

Algorithm 1. Game theoretic dynamic weighting based feature evaluation (GTDWFE) algorithm.

```
Input  : Dataset $D'$, feature set $F$, class $C$, number of features to be selected $T$
Output : Best feature subset $K$
1  $K := \emptyset$;
2  $w(f) := 1$ for all $f \in F$;
3  calculate $R_v(f)$ and $R_d(f)$ for all $f \in F$ using Eqs. 1–3;
4  $R_{sum}(f) := R_v(f) + R_d(f)$ for all $f \in F$;
5  for $j \leftarrow 1$ to $T$ do
6      for each $f \in F$ do
7          $L(f) := R_{sum}(f) \times w(f)$;
8      end
9      select $f_h$ with largest $L(f)$;
10     $K := K \cup \{f_h\}$;
11     $F := F \setminus \{f_h\}$;
12     if $|K| \neq T$ then
13         for each $f \in F$ do
14             compute Banzhaf power index $\phi_B(f)$ over $K$ using Eqs. 4–8;
15             $w(f) := w(f) \times (1 + \phi_B(f))$;
16         end
17     end
18 end
19 output $K$;
```

Algorithm 1 takes data set D', feature set F, binary classes C, and number of features to be selected T and outputs the best feature subset K. To implement this algorithm we first initialize the parameters, setting the same weight to each feature, *i.e.*, weight $w(f)$ of a feature f is set to 1 initially (line 2). Relevance to the target class $R_v(f)$ and similarity value of a feature $R_d(f)$ are calculated for all features (line 3). The greater the relevance of a feature to the target (AMR or non-AMR sequence), the more it can contribute to the prediction by sharing information with the target class. Also, the greater the distance of a feature from all other features, the lower the similarity of the feature with the remaining features, which indicates lower redundancy. We computed Pearson's correlation coefficient between a feature and class using Eq. 1, that is, we estimated the linear correlation between feature f and class C. Pearson's correlation coefficient (denoted by ρ) is computed using

$$R_v(f) = \rho = \frac{\mathbf{E}[(f - \mu_f)(C - \mu_C)]}{\sigma_f \sigma_C} \tag{1}$$

where the expectation is represented by $\mathbf{E}$, μ_f and μ_c are the means, and σ_f and σ_c correspond to the standard deviations for f and c, respectively. We used the absolute value $|\rho|$ as the value of $R_v(f)$ for the feature f. To find the $R_d(f)$ value of a feature f, we measured the average distance of a feature f with all other features using the Tanimoto Coefficient $TC(f, f_j)$ given in Eqs. 2 and 3. Here, d is the total number of features, for our case $d = 621$.

$$R_d(f) = \frac{1}{d-1} \sum TC(f, f_j),\ 1 \le j \le d, f \ne f_j \tag{2}$$

$$TC(f, f_j) = \frac{f \cdot f_j}{\left\|f\right\|^2 + \left\|f_j\right\|^2 - f \cdot f_j} \tag{3}$$

After summing the $R_v(f)$ and $R_d(f)$ values for all features (line 4), the algorithm iterates until the required T features have been selected. For every iteration the value of $L(f)$ is computed (lines 6–8), and the feature with the largest $L(f)$ is selected, added into subset K, and then eliminated from feature set F (lines 9–11).

The weights of the remaining candidate features are recalculated in each iteration to determine the impact of the candidate features on the features selected earlier (lines 13–16). We used the Banzhaf power index[28] to readjust the weight $w(f)$ of a feature f. The Banzhaf power index is widely used in game theory approaches to measure the power of a player to form a coalition with a set of other players S. Winning and losing coalitions in a game are those coalitions with $v(S) = 1$ and $v(S) = 0$, respectively. For every winning coalition of $S \cup \{r\}$ if S would lose without player r, then r is crucial to winning the game. Because player r is a feature, we made a slight modification to the original Banzhaf power index, and the updated definition of the Banzhaf power index for a player r is given in Eq. 4.

$$\phi_B(r) = \frac{1}{|\Pi_\delta|} \sum_{S \subseteq \Pi_\delta} \Delta_r(S) \tag{4}$$

Actual / Predicted	Negative	Positive
Negative	TN	FN
Positive	FP	TP

Table 2. Confusion matrix for classification performance.

where the marginal contribution of the feature r to all coalitions is $\Delta_r(S)$ where $\Delta_r(S) = v(S \cup r) - v(S)$. δ is the upper bound of the cardinality of S, and $|\Pi_\delta|$ gives the total number of subsets of $F\backslash r$ bounded by δ. This means that $\{\Pi_g\} \in S$, and g is the cardinality of a feature subset with $g = 1, 2, \ldots, \delta$.

If we consider two features r and t as two players, we can calculate their interdependence using Eq. 5 where C represents the binary classes 1 (AMR or positive class) and -1 (non-AMR or negative class).

$$\tau(r, t) = \begin{cases} 1 & \text{if } I(f_t; C|f_r) > I(f_t; C) \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

We can formulate $\Delta_r(S)$ as

$$\Delta_r(S) = \begin{cases} 1 & \text{if } I(S; C|f_r) \geq 0, \sum_{f_t \in S} \tau(r, t) \geq \frac{|S|}{2} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

From these equations we see that a feature is important if it increases the relevance of the subset S with the binary classes (*i.e.*, 1 and -1), and it should be interdependent with 50% or more of the members. The I's in these equations are the mutual information and conditional mutual information and are calculated using Eqs. 7 and 8, respectively, where U, V, and Z are random variables.

$$I(U; V) = \sum_{u \in U} \sum_{v \in V} p(u, v) \log \frac{p(u, v)}{p(u)p(v)} \tag{7}$$

$$I(U; V|Z) = \sum_{u \in U} \sum_{v \in V} \sum_{z \in Z} p(u, v, z) \log \frac{p(u, v|z)}{p(u|z)p(v|z)} \tag{8}$$

The algorithm is aborted when T features have been selected from the feature set F. The output feature subset K is the optimal feature subset for providing maximum relevance, minimum redundancy, and informative interdependence relations (line 19).

Imbalanced data. An imbalanced data set has significantly more of one class of training data than the other. Such a data set leads a classifier to predict the majority class more accurately while lowering the accuracy of the minority class predictions. This happens because of over-training of the majority class and under-training of the minority class. To avoid this, a data set can be balanced via sampling techniques[55,56]. There are two major sampling methods, namely oversampling and undersampling. In oversampling, we duplicate data from the minority class to balance the data set. In undersampling, we remove data from the majority class to balance the data set. In this study, we applied both over-sampling and under-sampling techniques to measure the performance of our prediction model.

Support Vector Machine. A Support Vector Machine (SVM)[29] is a supervised machine learning algorithm that represents each data item as a point in p-dimensional space and constructs a hyperplane (decision boundary) to separate data points into two groups. The core set of vectors identifying the hyperplane are known as support vectors. Unlike many classifiers, an SVM avoids overfitting by regularizing its parameters. Overfitting occurs when a classifier models the training data so well that it affects the accuracy of the classifier on new data. The SVM has proven to be a good classifier for protein sequences, classifying them with high accuracy. As predicting AMR is a binary classification problem, we chose an SVM for this work. A radial basis function[30,31] was used as its kernel.

R Scripts and Packages. R (https://cran.r-project.org/mirrors.html), a popular programming language for statistical analysis, provides many built-in packages for data analysis and machine learning. We wrote scripts in R to implement our GTDWFE feature evaluation algorithm and SVM classifier. We utilized the R *stats* (v3.5.0) package to perform the Pearson's correlation measurements, the *proxy* (v0.4–22) package for calculating the Tanimoto coefficients, and *infotheo* (v1.2.0) for finding the mutual information and conditional mutual information. We applied the ROSE (v0.0–3) package to balance the data set and the *tune()* function in the *e1071* (v1.6–8) package to find the best SVM parameters. The *caret* (v4.20) package was used to generate confusion matrices. Finally, we utilized the *FSelector* (v0.31) package to implement RReliefF[32].

Performance measurement. We measured the performance of the SVM classifier with our optimized feature set by generating confusion matrices which were used to calculate classification accuracies. Table 2 shows the structure of a confusion matrix, where *TP*, *TN*, *FP*, and *FN* are true positives (positives accurately classified), true negatives (negatives accurately classified), false positives (negatives classified as positives), and false negatives (positives classified as negatives), respectively. Classification accuracy is calculated from these values as given in Eq. 9.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

Accession codes. NCBI[57] accession numbers for all proteins used in this work are listed in Supplementary Tables S2–S11.

Code Availability

The R scripts written to implement our method are available at https://github.com/abu034004/GTDWFE.

Data Availability

All experimental data are available at https://github.com/abu034004/GTDWFE.

References

1. Hsueh, P.-R., Chen, W.-H. & Luh, K.-T. Relationships between antimicrobial use and antimicrobial resistance in gram-negative bacteria causing nosocomial infections from 1991–2003 at a university hospital in taiwan. *International journal of antimicrobial agents* **26**, 463–472 (2005).
2. Chopra, I. *et al*. Treatment of health-care-associated infections caused by gram-negative bacteria: a consensus statement. *The Lancet infectious diseases* **8**, 133–139 (2008).
3. Slama, T. G. Gram-negative antibiotic resistance: there is a price to pay. *Critical Care* **12**, S4 (2008).
4. Davis, J. J. *et al*. Antimicrobial resistance prediction in patric and rast. *Scientific reports* **6**, 27930 (2016).
5. Kang, C.-I. *et al*. Bloodstream infections caused by antibiotic-resistant gram-negative bacilli: risk factors for mortality and impact of inappropriate initial antimicrobial therapy on outcome. *Antimicrobial agents and chemotherapy* **49**, 760–766 (2005).
6. Davies, J. & Davies, D. Origins and evolution of antibiotic resistance. *Microbiology and molecular biology reviews* **74**, 417–433 (2010).
7. El Chakhtoura, N. G. *et al*. Therapies for multidrug resistant and extensively drug-resistant non-fermenting gram-negative bacteria causing nosocomial infections: a perilous journey toward 'molecularly targeted' therapy. *Expert review of anti-infective therapy* **16**, 89–110 (2018).
8. for Disease Control, C. & (US), P. *Antibiotic resistance threats in the United States, 2013* (Centres for Disease Control and Prevention, US Department of Health and Human Services, 2013).
9. Navon-Venezia, S., Kondratyeva, K. & Carattoli, A. Klebsiella pneumoniae: a major worldwide source and shuttle for antibiotic resistance. *FEMS microbiology reviews* **41**, 252–275 (2017).
10. Didelot, X., Bowden, R., Wilson, D. J., Peto, T. E. & Crook, D. W. Transforming clinical microbiology with bacterial genome sequencing. *Nature Reviews Genetics* **13**, 601 (2012).
11. Thomas, T., Gilbert, J. & Meyer, F. Metagenomics-a guide from sampling to data analysis. *Microbial informatics and experimentation* **2**, 3 (2012).
12. Oulas, A. *et al*. Metagenomics: tools and insights for analyzing next-generation sequencing data derived from biodiversity studies. *Bioinformatics and biology insights* **9**, BBI–S12462 (2015).
13. Yang, Y. *et al*. Args-oap: online analysis pipeline for antibiotic resistance genes detection from metagenomic data using an integrated structured arg-database. *Bioinformatics* **32**, 2346–2351 (2016).
14. Kleinheinz, K. A., Joensen, K. G. & Larsen, M. V. Applying the resfinder and virulencefinder web-services for easy identification of acquired antibiotic resistance and e. coli virulence genes in bacteriophage and prophage nucleotide sequences. *Bacteriophage* **4**, e27943 (2014).
15. Forsberg, K. J. *et al*. Bacterial phylogeny structures soil resistomes across habitats. *Nature* **509**, 612 (2014).
16. McArthur, A. G. & Tsang, K. K. Antimicrobial resistance surveillance in the genomic age. *Annals of the New York Academy of Sciences* **1388**, 78–91 (2017).
17. Xavier, B. B. *et al*. Consolidating and exploring antibiotic resistance gene data resources. *Journal of clinical microbiology* JCM–02717 (2016).
18. Arango-Argoty, G. *et al*. Deeparg: a deep learning approach for predicting antibiotic resistance genes from metagenomic data. *Microbiome* **6**, 23 (2018).
19. Ruppé, E. *et al*. Prediction of the intestinal resistome by a three-dimensional structure-based method. *Nature microbiology* **4**, 112 (2019).
20. Lal, T. N., Chapelle, O., Weston, J. & Elisseeff, A. Embedded methods. In *Feature extraction*, 137–165 (Springer, 2006).
21. Kohavi, R. & John, G. H. Wrappers for feature subset selection. *Artificial intelligence* **97**, 273–324 (1997).
22. Chowdhury, A. S., Alam, M. M. & Zhang, Y. A biomarker ensemble ranking framework for prioritizing depression candidate genes. In *Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), 2015 IEEE Conference on*, 1–6 (IEEE, 2015).
23. He, X., Cai, D. & Niyogi, P. Laplacian score for feature selection. In *Advances in neural information processing systems*, 507–514 (2006).
24. Talavera, L. An evaluation of filter and wrapper methods for feature selection in categorical clustering. In *International Symposium on Intelligent Data Analysis*, 440–451 (Springer, 2005).
25. Dash, M., Choi, K., Scheuermann, P. & Liu, H. Feature selection for clustering-a filter solution. In *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*, 115–122 (IEEE, 2002).
26. Guyon, I. & Elisseeff, A. An introduction to variable and feature selection. *Journal of machine learning research* **3**, 1157–1182 (2003).
27. Kotsiantis, S. Feature selection for machine learning classification problems: a recent overview. *Artificial Intelligence Review* 1–20 (2011).
28. Banzhaf, J. F. III Weighted voting doesn't work: A mathematical analysis. *Rutgers L. Rev.* **19**, 317 (1964).
29. Cortes, C. & Vapnik, V. Support-vector networks. *Machine learning* **20**, 273–297 (1995).
30. Chang, Y.-W., Hsieh, C.-J., Chang, K.-W., Ringgaard, M. & Lin, C.-J. Training and testing low-degree polynomial data mappings via linear svm. *Journal of Machine Learning Research* **11**, 1471–1490 (2010).
31. Vert, J.-P., Tsuda, K. & Schölkopf, B. A primer on kernel methods. *Kernel methods in computational biology* **47**, 35–70 (2004).
32. Robnik-Šikonja, M. & Kononenko, I. An adaptation of relief for attribute estimation in regression. In *Machine Learning: Proceedings of the Fourteenth International Conference (ICML'97)*, 296–304 (1997).
33. Kira, K. & Rendell, L. A. A practical approach to feature selection. In *Machine Learning Proceedings 1992*, 249–256 (Elsevier, 1992).

34. Kononenko, I. Estimating attributes: analysis and extensions of relief. In *European conference on machine learning*, 171–182 (Springer, 1994).
35. Robnik-Šikonja, M. & Kononenko, I. Theoretical and empirical analysis of relieff and rrelieff. *Machine learning* **53**, 23–69 (2003).
36. Liu, B. & Pop, M. Ardb–antibiotic resistance genes database. *Nucleic acids research* **37**, D443–D447 (2008).
37. Wattam, A. R. *et al.* Improvements to patric, the all-bacterial bioinformatics database and analysis resource center. *Nucleic acids research* **45**, D535–D542 (2016).
38. Favrot, L., Blanchard, J. S. & Vergnolle, O. Bacterial gcn5-related n-acetyltransferases: from resistance to regulation. *Biochemistry* **55**, 989–1002 (2016).
39. Li, W. & Godzik, A. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics* **22**, 1658–1659 (2006).
40. Fu, L., Niu, B., Zhu, Z., Wu, S. & Li, W. Cd-hit: accelerated for clustering the next-generation sequencing data. *Bioinformatics* **28**, 3150–3152 (2012).
41. Liu, B. *et al.* Combining evolutionary information extracted from frequency profiles with sequence-based kernels for protein remote homology detection. *Bioinformatics* **30**, 472–479 (2013).
42. Ding, C. H. & Dubchak, I. Multi-class protein fold recognition using support vector machines and neural networks. *Bioinformatics* **17**, 349–358 (2001).
43. Zhang, S., Ding, S. & Wang, T. High-accuracy prediction of protein structural class for low-similarity sequences based on predicted secondary structure. *Biochimie* **93**, 710–714 (2011).
44. Wei, L., Liao, M., Gao, X. & Zou, Q. Enhanced protein fold prediction method through a novel feature extraction technique. *IEEE transactions on nanobioscience* **14**, 649–659 (2015).
45. Cai, C., Han, L., Ji, Z. L., Chen, X. & Chen, Y. Z. Svm-prot: web-based support vector machine software for functional classification of a protein from its primary sequence. *Nucleic acids research* **31**, 3692–3697 (2003).
46. Li, Y. H. *et al.* Svm-prot 2016: a web-server for machine learning prediction of protein functional families from sequence irrespective of similarity. *PloS one* **11**, e0155290 (2016).
47. Dubchak, I., Muchnik, I., Holbrook, S. R. & Kim, S.-H. Prediction of protein folding class using global description of amino acid sequence. *Proceedings of the National Academy of Sciences* **92**, 8700–8704 (1995).
48. Dubchak, I., Muchnik, I., Mayor, C., Dralyuk, I. & Kim, S.-H. Recognition of a protein fold in the context of the scop classification. *Proteins: Structure, Function, and Bioinformatics* **35**, 401–407 (1999).
49. Altschul, S. F. *et al.* Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic acids research* **25**, 3389–3402 (1997).
50. Jones, D. T. Protein secondary structure prediction based on position-specific scoring matrices1. *Journal of molecular biology* **292**, 195–202 (1999).
51. Kurgan, L. A. & Homaeian, L. Prediction of structural classes for protein sequences and domains—impact of prediction algorithms, sequence representation and homology, and test procedures on accuracy. *Pattern Recognition* **39**, 2323–2343 (2006).
52. Kurgan, L., Cios, K. & Chen, K. Scpred: accurate prediction of protein structural class for sequences of twilight-zone similarity with predicting sequences. *BMC bioinformatics* **9**, 226 (2008).
53. Liu, T. & Jia, C. A high-accuracy protein structural class prediction algorithm using predicted secondary structural information. *Journal of theoretical biology* **267**, 272–275 (2010).
54. Chowdhury, A. S., Khaledian, E. & Broschat, S. L. Capreomycin resistance prediction in two species of *Mycobacterium* using a stacked ensemble method. *Journal of Applied Microbiology* (2019).
55. Lin, W.-C., Tsai, C.-F., Hu, Y.-H. & Jhang, J.-S. Clustering-based undersampling in class-imbalanced data. *Information Sciences* **409**, 17–26 (2017).
56. Junsomboon, N. & Phienthrakul, T. Combining over-sampling and under-sampling techniques for imbalance dataset. In *Proceedings of the 9th International Conference on Machine Learning and Computing*, 243–247 (ACM, 2017).
57. for Biotechnology Information, N. C. *NCBI accession number*, https://www.ncbi.nlm.nih.gov/ (Last accessed on August 17, 2018).

Acknowledgements
This work was supported in part by the Carl M. Hansen Foundation.

Author Contributions
A.S.C. collected the data, designed the method, performed the experiments, analyzed the experimental results, and prepared the initial manuscript. D.R.C. and S.L.B. analyzed the collected data, approved the method, guided the experiments, edited the manuscript, and further interpreted the experimental results. All authors reviewed and approved the manuscript.

Additional Information
Supplementary information accompanies this paper at https://doi.org/10.1038/s41598-019-50686-z.

Competing Interests: The authors declare no competing interests.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.