Hasan and Lonardi *BMC Bioinformatics* 2020, **21**(Suppl 14):367
https://doi.org/10.1186/s12859-020-03688-y

BMC Bioinformatics

RESEARCH Open Access

DeeplyEssential: a deep neural network for predicting essential genes in microbes

Md Abid Hasan* and Stefano Lonardi

From The Sixth International Workshop on Computational Network Biology: Modeling, Analysis, and Control (CNB-MAC 2019)
Niagara Falls, NY, USA. 07 September 2019

*Correspondence:
mhasa006@ucr.edu
Department of Computer Science and Engineering, University of California Riverside, 900 University Ave, 92507 Riverside, CA, USA

Abstract

Background: Essential genes are those genes that are critical for the survival of an organism. The prediction of essential genes in bacteria can provide targets for the design of novel antibiotic compounds or antimicrobial strategies.

Results: We propose a deep neural network for predicting essential genes in microbes. Our architecture called DEEPLYESSENTIAL makes minimal assumptions about the input data (i.e., it only uses gene primary sequence and the corresponding protein sequence) to carry out the prediction thus maximizing its practical application compared to existing predictors that require structural or topological features which might not be readily available. We also expose and study a hidden performance bias that effected previous classifiers. Extensive results show that DEEPLYESSENTIAL outperform existing classifiers that either employ down-sampling to balance the training set or use clustering to exclude multiple copies of orthologous genes.

Conclusion: Deep neural network architectures can efficiently predict whether a microbial gene is essential (or not) using only its sequence information.

Keywords: Essential genes, Deep neural network, Microbes, Data leak

Background

Essential genes are those genes that are critical for the survival and reproduction of an organism [1]. Since the disruption of essential genes induces the death of an organism, the identification of essential genes can provide targets for new antimicrobial/antibiotic drugs [2, 3]. Essential genes are also critical for the creation of artificial self-sustainable living cells with a minimal genome [4]. Finally, essential genes have been a cornerstone in the study of the origin and evolution of organisms [5].

The identification of essential genes via wet-lab experiments is labor intensive, expensive and time-consuming. Such experimental procedures include single gene knock-out

[6, 7], RNA interference, and transposon mutagenesis [8, 9]. Moreover, these experimental approaches can produce contradicting results [10]. With the recent advances in high-throughput sequencing technology, computational methods for predicting essential genes has become a reality. Some of the early prediction methods used comparative approaches by homology mapping, see, e.g., [11, 12]. With the introduction of large gene database such as DEG, CEG, and OGEE [13–15], researchers designed more complex prediction models using a wider set of features. These features can be broadly categorized into (i) sequence features, i.e., codon frequency, GC content, gene length [16–18], (ii) topological features, i.e., degree centrality, cluster coefficient [19–22], and (iii) functional features, i.e., homology, gene expression cellular localization, functional domain and other molecular properties [10, 23–26]. More recent studies about the 3D structure of proteins can also be incorporated in topological features set [27, 28].

Sequence-based features can be directly obtained from the primary DNA sequence of a gene and its corresponding protein sequence. Functional features such as network topology require knowledge of protein-protein interaction network, e.g., STRING and HumanNET [29, 30]. Gene expression and functional domain information can be obtained from databases like PROSITE and PFAM [31, 32]. Some of the less studied bacterial species, however, lack these functional and topological features, which would prevent the use of prediction tools that rely on them. Sequence-based classifiers are the most practical methods because they use the minimal amount of features.

Several studies have been published on the problem of predicting essential genes from their sequences. In [17], the authors developed a tool called ZUPLS that uses (i) a Z-curve derived from the sequence, (ii) homology mapping and (iii) domain enrichment score as features to predict essential genes in twelve prokaryotes after training the model on two bacteria. Although ZUPLS worked well on cross-organism prediction, the limited number of bacterial species used in the training set cast doubts on the ability of ZUPLS to generalize to more diverse bacterial species. In [33], the authors proposed a method that employs PCA on features derived from the gene sequence, protein domains, homologous and topological information. Among the studies that predict essential genes across multiple bacterial species, [26] employed several genomic, physio-chemical and subcellular localization features to predict gene essentiality across fourteen bacterial species. In their work, the authors dealt with the redundancy in the dataset (i.e., homologous genes shared by multiple bacterial genomes) by clustering genes based on their sequence similarities. In [16], nucleotide, di-nucleotide, codon, amino acid frequencies, and codon usage analysis were used for predicting essentiality in sixteen bacterial species. The authors used CD-HIT [34] for homology detection in both essential and non-essential genes. In [35], the authors identified essential genes in fifteen bacterial species using information theoretical features, e.g., Kullback-Leibler divergence between the distribution of k-mers (for $k = 1, 2, 3$), conditional mutual information, and entropy. Although their work showed promising results for intra-organism and cross-organism predictions, the model performed rather poorly when trained on the complete bacterial dataset. Recently, [10] showed the most extensive prediction analysis of thirty-one bacterial species. The authors employed the features proposed in [26], with additional features such as transmembrane helices and Hurst exponent. Their algorithm used a regularized feature selection method called least absolute shrinkage and selection operator (Lasso) and used a support vector machine (SVM) as a classifier.

The most recent work on gene essentiality prediction [36] uses network-based features, Lasso for feature selection, and a Random Forest as the classifier. The authors used a recursive feature extraction technique to compute 267 features in three different categories i.e. *local features* such as degree distribution, *egonet features* which refers to the node and the induced subgraph formed by all of its neighbors, and *regional features* which are a combination of local and egonet features. They also used fourteen network centrality measures as a separate feature set for the essentiality prediction. Finally, they combined their network-based features with the sequence based features in [10] and [17] for their prediction model. For the models in [10, 36], and [17], the authors down-sampled non-essential genes to balance the training set but did not realize that their dataset contained multiple copies of homologous genes which created a "data leak" issue which biased their results (see below).

In this work we propose a feed-forward deep neural network (DNN) called DEEPLYESSENTIAL that uses features derived solely from the primary gene sequence, thus maximizing its practical application compared to other predictors that require structural or topological features which might not be readily available. To the best of our knowledge, this is the first time a deep neural network has been used for gene essentiality prediction.

Materials and methods

Dataset

Genomic data for thirty bacterial species were obtained from the database DEG, which is a curated and comprehensive repository of experimentally-determined bacterial and archaeal essential genes. Among the thirty bacterial species, nine are *Gram-positive* (GP) and twenty-one are *Gram-negative* (GN). DEG provides the primary DNA sequence and the corresponding protein sequence for both essential and non-essential genes, as well as gene functional annotations. We only considered protein-coding genes, i.e., we excluded RNA genes, pseudo-genes, and other non-coding genes. At the time of writing, DEG contained 28,876 essential protein-coding genes (of which 8746 belonged to a GP species and 20,130 belonged to a GN species) and 209,026 non-essential protein-coding genes (of which 45,002 were GP and 164,024 were GN). Table 1 shows the basic statistics of the dataset. Observe that the dataset is highly unbalanced. While species NC_000907 and NC_002771 have approximately the same number of essential and non-essential genes and bacteria NC_000908 has more essential genes than non-essential genes, for ten bacterial species less than 10% of their genes are essential. In order to improve the performance of our classifier, we balanced the dataset by down-sampling non-essential genes.

Feature selection

As said, various intrinsic gene features, such as protein domains, protein-interaction network data, etc. have been used for predicting gene essentiality [33, 35]. DEEPLYESSENTIAL utilizes codon frequency, maximum relative synonymous codon usage (RSCU), codon adaptation index (CAI), gene length and GC content. In addition to DNA-derived features, DEEPLYESSENTIAL uses amino acid frequencies and the length of the protein sequence.

Table 1 The thirty bacterial species used for our experiments (GP is Gram-positive, GN is Gram-negative)

Accession	GP/GN	# Essential genes	# Non-essential genes
NC_000907	GN	1284	1024
NC_000908	GP	762	188
NC_000913	GN	1810	14000
NC_000915	GN	646	2270
NC_000962	GP	4144	17586
NC_000964	GP	542	7808
NC_002163	GN	788	5602
NC_002505/002506	GN	1558	5886
NC_002516	GN	906	21266
NC_002745	GP	604	4562
NC_002771	GP	620	644
NC_003197	GN	460	8456
NC_004347	GN	804	2206
NC_004631	GN	1422	15822
NC_004663	GN	650	8906
NC_005966	GN	998	5188
NC_006351/006350	GN	1010	10444
NC_007297	GP	454	2674
NC_007795	GP	702	5082
NC_008463	GN	670	1920
NC_008601	GN	784	2658
NC_009009	GP	436	4104
NC_009511	GN	1070	8630
NC_010729	GN	1488	6870
NC_011375	GP	482	2354
NC_011916	GN	960	6448
NC_016776	GN	1094	7486
NC_016810	GN	706	8070
NC_016856	GN	210	10420
NC_007650/007651	GN	812	10452

Codon frequency

Codon frequency has been recognized as an important feature for gene essentiality prediction [10, 26]. Given the primary DNA sequence of a gene, its codon frequency is computed by sliding a window of three nucleotides along the gene. The raw count of $4^3 = 64$ codons is then normalized by the total number of genes. Observe in Fig. 1 that the codon frequency is significantly different in the two classes. For instance, codon AAA, GAA, TGA, GAT, AAG, ATT and AGA have at least 30% difference in their normalized codon frequency between essential and non-essential genes.

Gene length and GC content

Other distinguishing features for gene essentiality are gene length and GC content. Figure 2 shows the distribution of gene length in GP, GN and the combined dataset (GP+GN). Observe that genes in the GP+GN dataset and the GN dataset have a similar average length in the two classes, while essential genes in the GP dataset are on average longer than non-essential genes. As said, the GC content is another informative feature of essentiality prediction. Figure 3 shows the difference in distribution in GC content

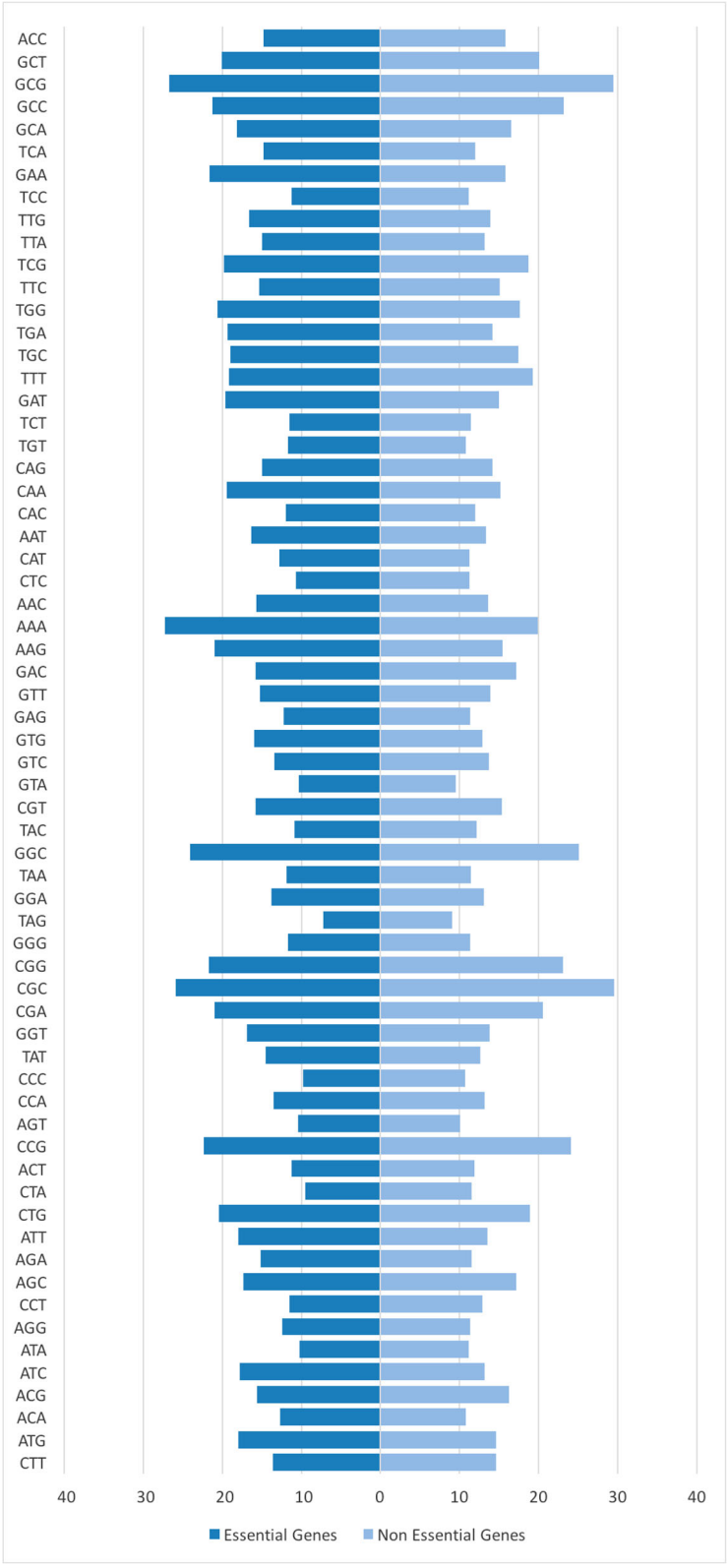

Fig. 1 Normalized codon frequency of genes in GP + GN dataset

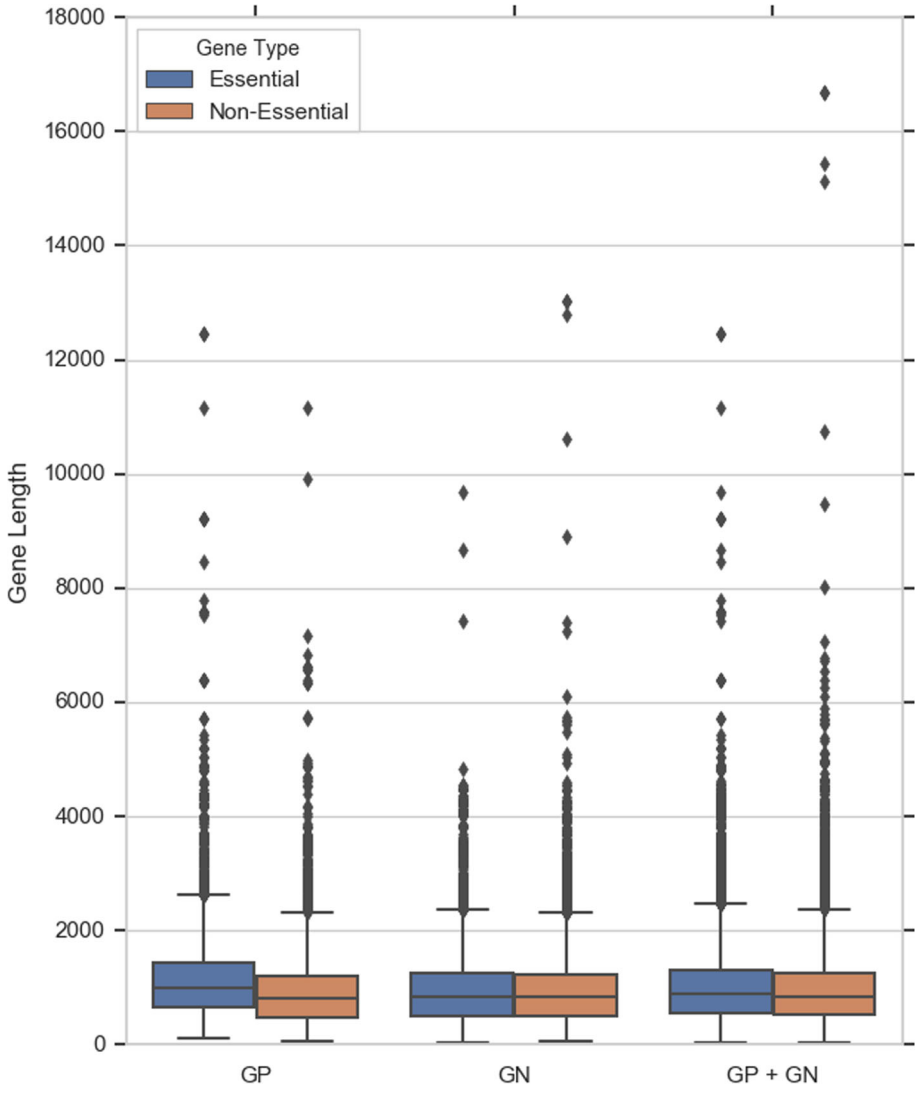

Fig. 2 Distribution of gene lengths in datasets GP+GN, GN, GN

between the two classes. Observe that non-essential genes have higher GC content than essential genes.

Relative synonymous codon usage

Unbalanced synonymous codon usage is prevalent both in prokaryotes and eukaryotes [37]. The degree of bias varies among genes not only in different species but also among genes in the same species. Differences in codon usage in one gene compared to its surrounding genes may imply its foreign origin, different functional constraints or a different regional mutation. As a result, examining codon usage helps to detect changes in evolutionary forces between genomes. Essential genes are critical for the survival of an organism thus codon usage acts as a strong distinguishing feature. To calculate the relative synonymous codon usage we compare the observed number of occurrence of each codon to the expected number of occurrences (assuming that all synonymous codons have equal

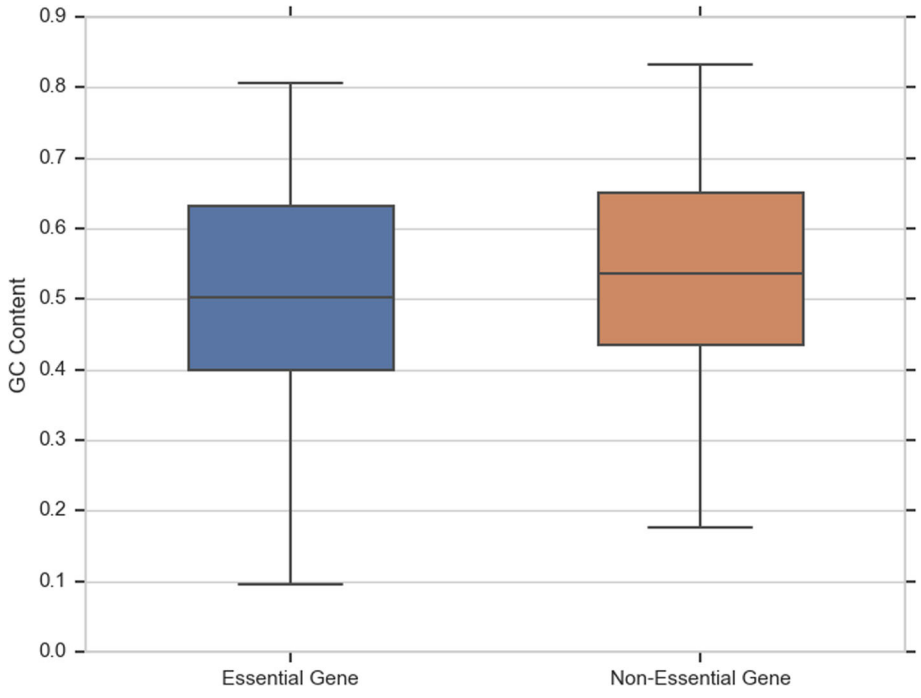

Fig. 3 GC content distribution in essential and non-essential gene sets in the GP+GN dataset

probability). Given a synonymous codon i that has an n-fold degenerate amino acid, we compute the *relative synonymous codon usage* (RSCU) as follows

$$\text{RSCU}_i = \frac{X_i}{(1/n)\sum_{i=1}^{n} X_i}$$

where X_i is the number of occurrence of codon i, and n is 1, 2, 3, 4, or 6 (according to the genetic code).

Codon adaptation index

The *codon adaptation index* (CAI) estimates the bias towards certain codons that are more common in highly expressed genes [37]. The CAI is defined as the geometric mean of the relative adaptedness statistics. The *relative adaptedness* for codon i is defined on the relative frequency of the codon in a species-specific reference set of highly expressed genes. Formally, the relative adaptedness is defined by

$$r_i = \frac{\text{RSCU}_i}{\text{RSCU}_{max}} = \frac{X_i}{X_{max}}$$

where RSCU_{max} and X_{max} are corresponding RSCU and X value of the most frequently used codon. The CAI for a gene is defined by

$$\text{CAI} = \left(\prod_{i=1}^{L} r_i\right)^{\frac{1}{L}}$$

where L is the number of codons in the gene excluding methionine, tryptophan, and stop codon. The range of CAI is $(0, 1]$ where higher values indicating a higher proportion of the most abundant codons.

Protein sequence features

Another informative set of features used for the prediction of gene essentiality are those derived from the corresponding protein sequences. Previous studies have used frequencies of rare amino acids, and the number of codons that are one-third base mutations removed from the stop codons [10]. DEEPLYESSENTIAL only uses amino acids frequencies and the lengths of the protein sequences.

Combining all the features

Given the primary DNA sequence of a gene, we generate $4^3 = 64$ values for the codon frequency, and single values for the GC content, gene length, CAI and RSCU_{max}. From the protein sequence, we compute the amino acid frequency vector (20 values), and one value for the protein length. The total number of features used by DEEPLYESSENTIAL is 89.

Multi-layer perceptron

A multi-layer perceptron (MLP) consists of multiple layers of computational units where the information flows in the forward direction, from input nodes through hidden nodes to the output nodes without any cycles [38]. MLP networks have been used successfully for several molecular biology problems, see, e.g. [39–41]. The architecture of DEEPLYESSENTIAL is composed of an input layer, multiple hidden layers, and an output layer. The output layer encodes the probability of a gene to be essential. The addition of a dropout layer makes the network less sensitive to noise during training and increase its ability to generalize. This layer randomly assigns zero weights to a fraction of the neurons in the network [42].

Let $\overrightarrow{x} = (x_1, \cdots, x_n)^T$ be the input to the MLP. Let vector y denote the output of the i^{th} hidden layer. The output y^i depends on the input in the previous layer as follows

$$y^i = a\left(W^i x^{(i-1)} + b^{(i-1)}\right)$$

where a is the activation function, b is the bias and W is the weight matrix for the edges in the network. During the training phase, the network learns the weights W and the bias b. DEEPLYESSENTIAL uses a rectified linear unit (ReLU) in each neuron in the hidden layers. ReLU is an element-wise operation that clamps all negative values to zero.

In the output layer DEEPLYESSENTIAL uses a sigmoid as the activation function to perform discrete classification

$$y = \frac{1}{1 + e^{-x}}$$

The loss function is the binary cross-entropy defined by

$$\sum_{c=1}^{M} \hat{y}_{o,c} \log(p_{o,c})$$

where M is the number of classes (two in our case), $\hat{y}$ is the binary indicator if class label c is the correct classification for observation o, and p is the predicted probability that observation o belongs to class c. Figure 4 illustrates the architecture of the neural network used in DEEPLYESSENTIAL.

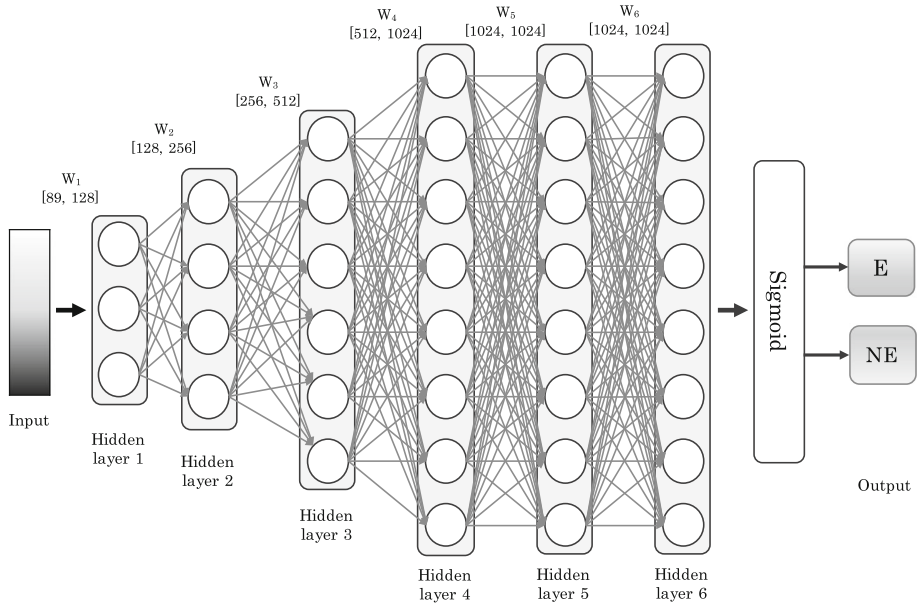

Fig. 4 The architecture of the neural network used in DEEPLYESSENTIAL

Results and discussion

Classifier design and evaluation

As mentioned above, the number of non-essential genes is significantly larger than the number of essential genes. To address this imbalance in the training set and allow for unbiased learning, we randomly down-sample non-essential genes. In [18], the authors showed that balancing the dataset did not negatively influence the prediction of gene essentiality.

Model hyper-parameters

Recall that each gene (and its corresponding protein) is represented by 89 features in the input layer. The architecture of DEEPLYESSENTIAL was determined by running extensive experiments on the training data over a wide range of hyper-parameters. The number of hidden layers, the number of nodes in each of the hidden layers, the batch size, the dropout rate and the type of optimizer were selected by optimizing the performance of the classifier during cross-validation. Table 2 lists the range of hyper-parameters considered and the values of the hyper-parameters selected for the final architecture of DEEPLYESSENTIAL.

Observe in Fig. 4 that the final fully-connected layer reduces the 1024 dimensional vector to a two-dimensional vector corresponding to the two prediction classes

Table 2 Hyperparameters for DEEPLYESSENTIAL

Parameters	Range	Selected parameter
# hidden layers	[2 - 8]	6
# nodes	[32, 64, 128, 512, 1024, 2048]	128, 256, 512, 1024, 1024, 1024
Dropout rate	[0.1 - 0.5]	0.3
Epochs	–	100 (early stopping)
Optimizer	sgd, adam, adadelta, RMSProp	adadelta

(essential/non-essential). The sigmoid activation function forces the output of the two neurons in the output layer to sum to one. Thus the output value represents the probability of each class. Among the available optimizer in Table 2, we chose adadelta because of its superior performance. Adadelta is parameter-free, thus we did not need to define the learning rate. The training was run for 100 epochs with early stopping criteria.

We trained DEEPLYESSENTIAL on three datasets, namely GP, GN, and GP+GN. For each dataset, 80% data is used for training, 10% data for validation and 10% data for testing. The random selection was repeated ten times, i.e., ten-fold cross-validation was performed to complete the inference.

Evaluation metrics

The tools described in [10, 16, 26] and [35] are currently unavailable. We ran DEEPLYESSENTIAL on the datasets used in the corresponding papers, and compared DEEPLYESSENTIAL's classification metrics to the published metrics.

We evaluated the performance of DEEPLYESSENTIAL using the Area Under the Curve (AUC) of the Receiver Operating characteristic Curve (ROC). ROC plot represents the trade-off between sensitivity and specificity for all possible thresholds. Although our primary evaluation measure is the AUC score, we also report the following additional performance measures

$$\begin{aligned} \text{Sensitivity(Sn)} &= \frac{TP}{(TP+FN)} \\ \text{Specificity(Sp)} &= \frac{TN}{(FP+TN)} \\ \text{PPV} &= \frac{TP}{(TP+FP)} \\ \text{Accuracy} &= \frac{(TP+TN)}{(TP+FN+TN+FP)} \end{aligned}$$

where TP, TN, FP and FN represent the number of true positives, true negatives, false positives, and false negatives, respectively.

All experiments were carried out a Titan GTX 1080 Ti GPU, running Keras v2.1.5.

Gene essentiality prediction

We collected essential and non-essential gene for thirty bacterial species described above into three datasets, namely GP, GN, and GP+GN. After re-balancing the dataset by down-sampling non-essential genes, we extracted the features for each gene as explained above. Table 3 shows the basic statistics for each dataset.

Table 4 shows the training classification performance of DEEPLYESSENTIAL, averaged over ten repetitions. The violin plot in Fig. 5 shows the distribution of AUCs across the ten repetitions of the experiment, which appears very stable. The receiver operator curves (ROC) are shown in Fig. 6. DEEPLYESSENTIAL yielded an area under the curve of 0.838,

Table 3 Basic statistics for GP, GN, and GP+GN (balanced and unbalanced)

Dataset	# Training samples	# Validation samples	# Test samples
GP	7,065	883	884
GN	14,364	1,795	1,797
GP+GN (bal)	21,432	2,678	2,680
GP+GN (unbal)	90,571	11,321	11,322

Table 4 Training classification performance of DEEPLYESSENTIAL on GP, GN, GP+GN

Metric	GP	GN	GP+GN
AUC	0.838	0.823	0.842
Sensitivity	0.741	0.784	0.801
Specificity	0.758	0.708	0.721
PPV	0.774	0.722	0.749
Accuracy	0.749	0.745	0.762

0.829 and 0.842 for GP, GN, and GP+GN on average, respectively. The ROC curve also indicates the relation between the number of training samples and the stability of the prediction performance. Observe that DEEPLYESSENTIAL's performance was more stable on the GP+GN dataset than the GP dataset (which contains the smallest number of samples). Tthe precision-recall curves shows the ability of DEEPLYESSENTIAL to yield low false positive rate and low false negative rate consistently across all datasets.

Comparison with down-sampling methods

As mentioned in the previous section, the gene essentiality dataset is highly unbalanced. It is well-known that class imbalance can negatively affect the performance of a classifier [43]. To quantify how class imbalance affects the performance of our classifier we trained DEEPLYESSENTIAL on the full (unbalanced) dataset that has 322.6% more non-essential genes than essential genes. Figure 7 shows that the sensitivity and Positive Predictive Value (PPV) of the classifier trained on unbalanced data are much worse than the balanced dataset. As said, some of the existing methods use down-sampling to address this problem. Both Liu et al. [10] and Azhagesan et al. [36] randomly down-sampled the majority class data to match the size of the minority class. DEEPLYESSENTIAL also uses this

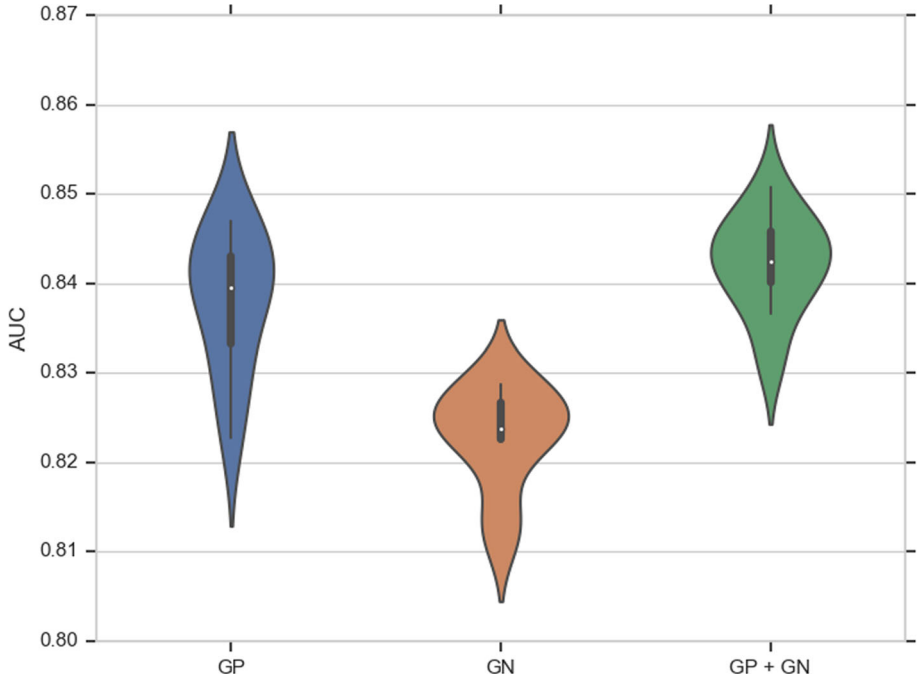

Fig. 5 Violin plot of DEEPLYESSENTIAL's AUC across ten experiments on the GP, GN, GP+GN datasets

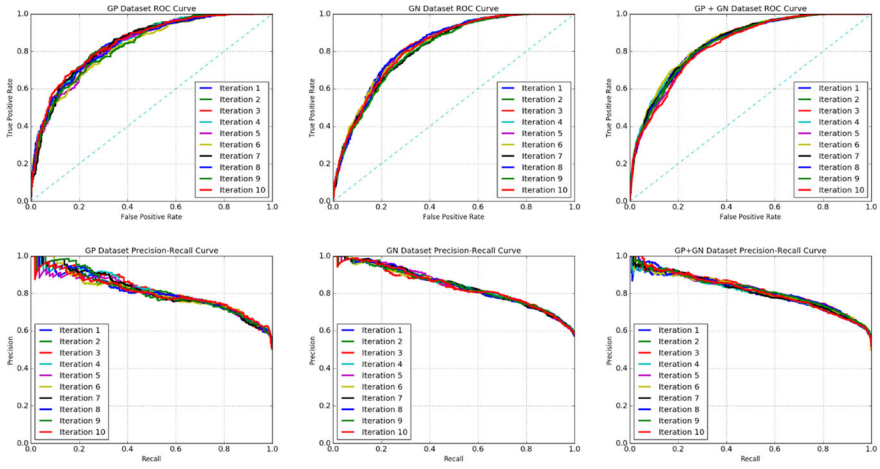

Fig. 6 DEEPLYESSENTIAL's ROC and AUPR curves on GP, GN, GP+GN

approach. Table 5 shows the performance DEEPLYESSENTIAL compared to the two published methods that use down-sampling. Observe that DEEPLYESSENTIAL achieves the best AUC, sensitivity, and PPV.

Identification of "data leak" in the gene essentiality prediction

Bacteria are unicellular organisms with a relatively small set of genes. Across bacterial species, a significant fraction of the genes is conserved because they perform similar

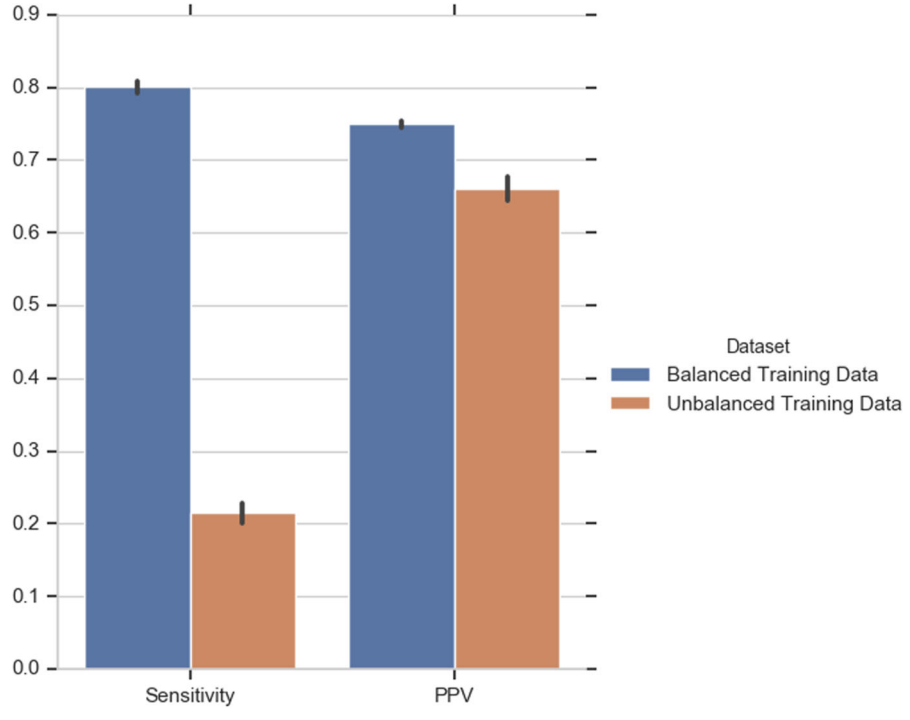

Fig. 7 Comparing the prediction performance of DEEPLYESSENTIAL when trained on balanced or unbalanced GP+GN dataset

Table 5 Comparing the performance of DEEPLYESSENTIAL on down-sampled dataset against methods that solely use sequence features; numbers in boldface indicate the best performance

Method	# features	AUC	Sensitivity	PPV
Liu et al. 2017	40	0.794	0.715	0.243
Azhagesan et al. 2018	267	0.838	0.754	0.321
ZUPLS	274	0.705	0.663	0.255
DeeplyEssential	89	**0.842**	**0.801**	**0.749**

fundamental biological functions. These conserved genes are quite similar at the sequence level. All published methods rely on a dataset containing multiple bacteria on which genes have been labeled essential or non-essential. Let us call x and y two homologous genes, i.e., two genes that have a very similar primary DNA sequence. If x is used on the training and y if used for testing, this introduces a bias, or a "data leak". Training examples and testing examples are supposed to be distinct, and in this hypothetical scenario, they are not.

To quantify the effect of the data leak issue, we clustered the set of all genes across the thirty bacterial species using OrthoMCL [44]. OrthoMCL is a popular method for clustering orthologous, homologous and paralog proteins which use reciprocal best hit alignment to detect potential in-paralog/recent paralog pair, and reciprocal best hit alignments between two genomes to identify potential ortholog pairs. A similarity graph is then generated based on the proteins that are interlinked. To split large clusters, a Markov Clustering algorithm (MCL) is then invoked [45]. Inside MCL clusters, weights between each pair of proteins are normalized to correct for evolutionary differences.

As said, OrthoMCL produces a list of clusters where each cluster consists of genes that have been determined to be orthologous. To quantify the effect of gene sequence similarity on the prediction performance, we created a dataset where no gene from a single cluster can be assigned to both training set and testing set. The modified dataset contains 11,168 training samples, 2,798 validation samples, and 4,270 testing samples. The prediction was repeated ten times. Table 6 shows how the clustering step heavily influences DEEPLYESSENTIAL's prediction performance. AUC decreased by more than 7% (on average), while the accuracy decreased by 6.9% (along with a significant decrease in all performance measures). Figure 8 shows the difference in performance before and after clustering. While the AUCs were stable across experiments, sensitivity, specificity, and PPV varied largely across experiments on the clustered dataset.

Comparison with methods that cluster orthologous genes

Some published studies have addressed the data leak issue by identifying homologous genes using sequence similarity metrics. In [35], the authors used the Kullback-Leibler

Table 6 Comparing the effect of clustering on the prediction performance of DEEPLYESSENTIAL on the GP+GN dataset

Metric	Non-clustered	Clustered	Difference (%)
AUC	0.842	0.786	7.12%
Sensitivity	0.801	0.780	2.69%
Specificity	0.721	0.646	11.60%
PPV	0.749	0.688	8.86%
Accuracy	0.762	0.713	6.87%

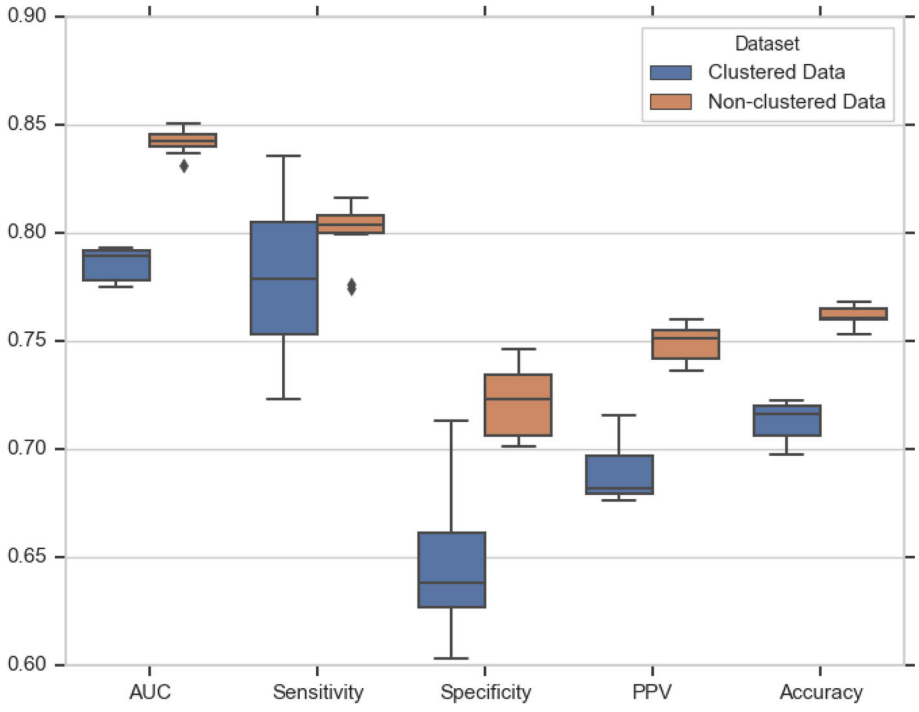

Fig. 8 Effect of "data leak" on DEEPLYESSENTIAL's prediction performance on GP+GN dataset

divergence to measure the distance between k-mer distribution (for $k = 1, 2, 3$) obtained from sequences. In [16], the authors used CD-HIT to remove redundancy in the training data and improve the generalization ability of their model. As explained in the previous section, DEEPLYESSENTIAL uses OrthoMCL to cluster homologous genes to prevent similar genes to appear in both training and testing dataset. Table 7 shows the performance comparison of DEEPLYESSENTIAL with [16] and [35] on their respective datasets.

Observe that in both cases DEEPLYESSENTIAL achieves the best predictive performance. As said, although each of these two approaches uses a distinct method to determine orthologous genes, the use of the same dataset for the experiments ensures a fair comparison.

Feature importance

DEEPLYESSENTIAL uses exclusively sequence-based features and yet produces higher prediction performance. Unlike other machine learning classifiers, the DNN architecture does not readily provide any insight about the feature set that contributed maximally

Table 7 Comparing the performance of DEEPLYESSENTIAL and Ning et al. and Nigatu et al. on their respective datasets [16, 35]; numbers in boldface indicate the best performance

Method	Clustering method	AUC
Ning et al. 2014	CD-HIT	0.758
DEEPLYESSENTIAL	OrthoMCL	**0.818**
Nigatu et al. 2017	*Kullback-Leibler divergence*	0.650
DEEPLYESSENTIAL	OrthoMCL	**0.840**

towards the prediction performance. To understand the impact of a feature on the predictive performance, we carried out an ablation study which removes one or more features from the input and determines the performance difference. In order for the ablation study to be informative, features cannot be highly correlated. In this latter case, the removal of a feature is immediately compensated by its highly correlated feature. To address this issue, we first computed the pairwise Pearson correlation among all input features. Figure 9 illustrate the heatmap of the pairwise correlation. Each axis shows the indices of the features: indexes 0–65 contains DNA specific feature, index 68–89 contains protein specific features. GC content, CAI and RSCU_{max} have a negative correlation with all other features. There were nineteen pair of features showing a correlation higher than 0.9 (in absolute value). For the ablation study, we either removed one feature at a time (if uncorrelated) or one of the 19 feature pairs. We tested the performance changes on the GP + GN dataset using 5-fold cross-validation. Specifically, we measured the difference in AUC and ordered the features based on their impact in decreasing the predictive performance (Fig. 10). Observe that codon TTT caused the highest AUC decrease (3.5%) while AGA, TTC, CGT, CGA, gene length, protein length, GC content, CAI, amino acids K, L, R, W, Y, C, G, E, F, and pairs of correlated features CCG+CGC, TAA+TTA, gene length+L, D, and protein length+T induced 2.5% –3% AUC decrease. Our finding that gene and protein length are highly informative features for essentiality prediction recapitulate their prominence, as illustrated in other sequence-based methods, e.g., [10] and [17]. Moreover, it is well-known that in essential genes within the functional category related to information

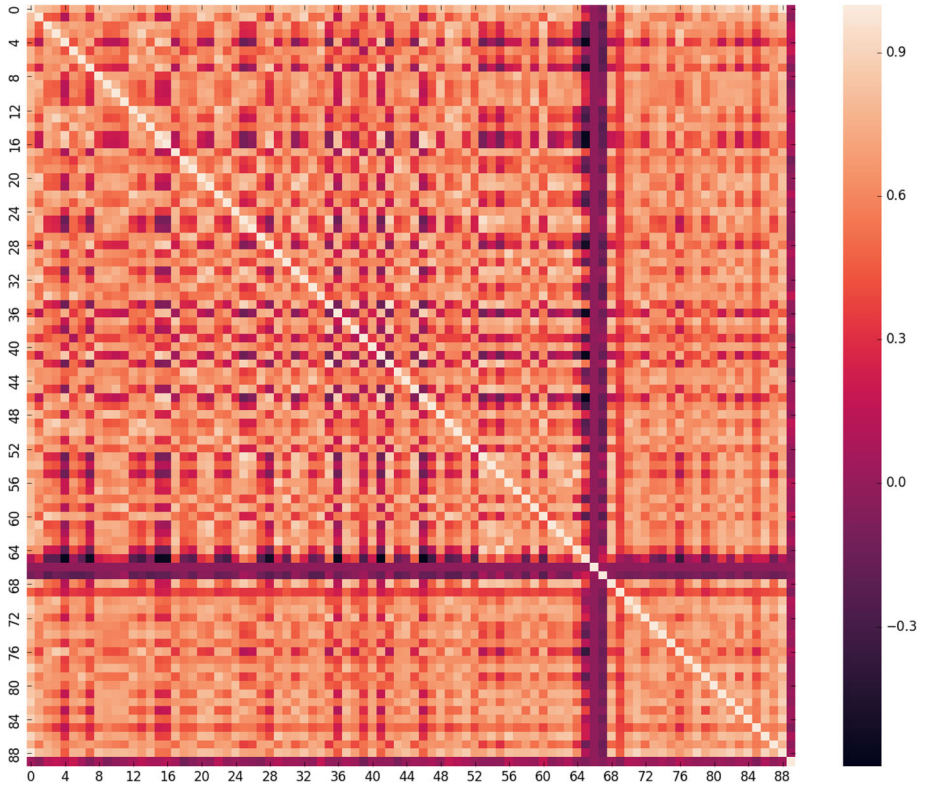

Fig. 9 Pairwise correlation among all features; features 0–65 are DNA specific feature; features 68–89 are protein specific features

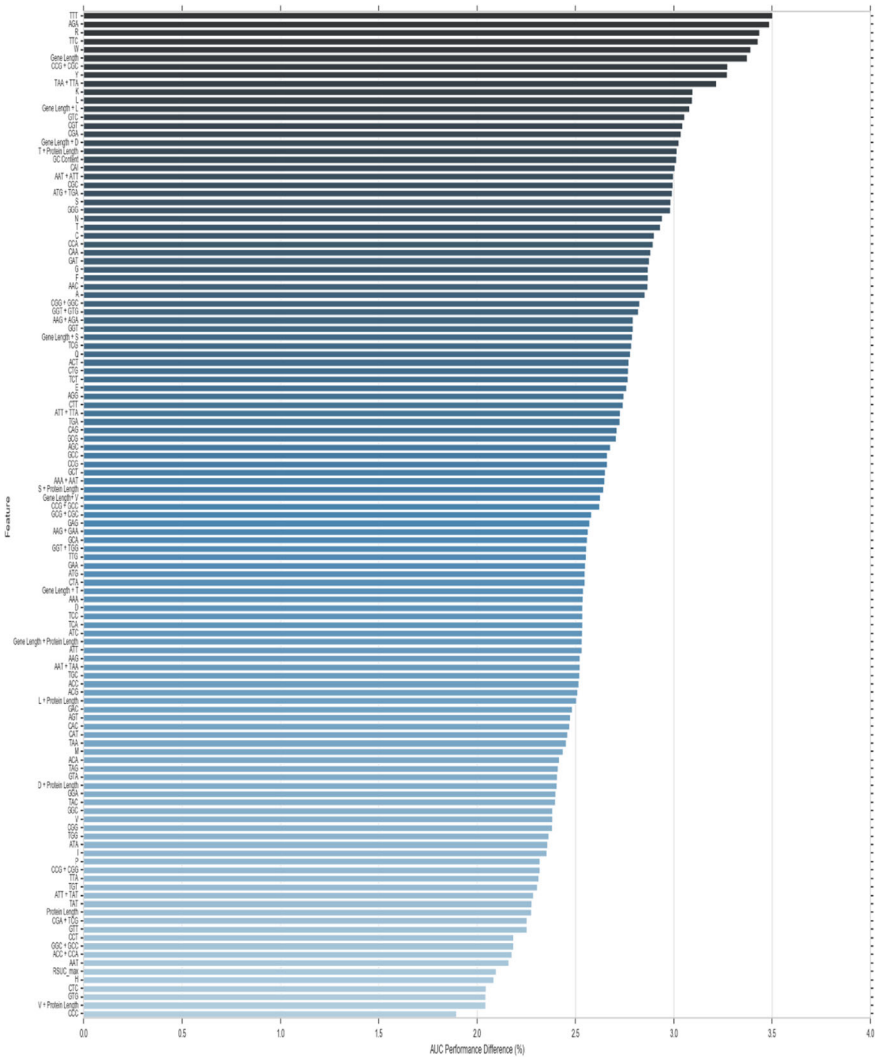

Fig. 10 Changes in AUC predictive performance due to the removal of a feature or pairs of correlated features

storage and process, encoded amino acids K, L and subcategories of encoded amino acids C, G, E, F are preferentially suited at the leading strand where these are responsible for energy production and conversion, carbohydrate transport and other essential metabolic processes [46].

Discussion

A large number of structural and functional features have been used for gene essentiality prediction, i.e. producibility, choke points, load scores, damages, degree of centrality, clustering coefficient, closeness centrality, betweenness centrality, gene expression, phyletic retention, among others. These features cannot be obtained from the gene sequences and are often not available for many bacterial species. To maximize its practical utility, DEEPLYESSENTIAL uses exclusively features derived directly from the sequence.

Previous works have addressed the high imbalance of the training dataset by either down-sampling non-essential genes or by clustering orthologous genes across species. In

order to make a meaningful and fair comparison, we compared DEEPLYESSENTIAL's performance to both approaches. In fact, our experiments showed that DEEPLYESSENTIAL has better predictive performance both on down-sampled and clustered datasets. On the down-sampled dataset used in [10], DEEPLYESSENTIAL demonstrated an improvement of 12.8% in AUC compared to [10]. In addition, DEEPLYESSENTIAL produced significantly better sensitivity and precision than the three methods in Table 5, achieving 6.2% improved sensitivity and 137.4% improved precision compared to [36]. If one uses all the 597 features in [36], then this latter method achieves 1.7% improved AUC compared to DEEPLYESSENTIAL. We believe that collecting this very large amount of features from multiple databases does not warrant the additional (minor) benefit in predictive performance. DEEPLYESSENTIAL also achieved better performance on clustered datasets. Table 7 shows 7.9% and 29.2% improved AUC compared to [16] and [35], respectively.

As an alternative to the proposed approach that uses a carefully selected set of features as input, one could consider training a convolutional neural network (CNN) that uses exclusively one hot encoding of the DNA and protein sequence as input. One hot encoding is a process that converts categorical variables into a numerical vector that is convenient for the prediction by machine learning models. We expect that the limited size of the available training data would be insufficient to allow for the CNN to extract relevant features. As a consequence, we expect CNN-based classifiers not to be as accurate compared to the architecture proposed here.

Conclusion

We proposed a deep neural network architecture called DEEPLYESSENTIAL to predict gene essentiality in microbes. DEEPLYESSENTIAL makes a minimal assumption about the input data (i.e, it only uses the gene sequence), thus maximizing its practical application compared to other predictors that require structural or topological features which might not be readily available. Extensive experiments show that DEEPLYESSENTIAL has better predictive performance than existing prediction tools. We believe that DEEPLYESSENTIAL could be further improved if more annotated bacterial data was available, making it an essential tool for drug discovery and synthetic biology experiments in microbes.

Abbreviations

DEG: Database of essential genes; PCA: Principal component analysis; GP: Gram positive; GN: Gram negative; DNN: Deep neural network; RSCU: Relative synonymous codon usage; CAI: Codon adaptation index; AUC: Area under the curve; ROC: Receiver operator curve; PPV: Positive predictive value; CNN: Convolutional neural network

Acknowledgements

Not applicable.

About this supplement

This article has been published as part of *BMC Bioinformatics Volume 21 Supplement 14, 2020: Selected original articles from the Sixth International Workshop on Computational Network Biology: Modeling, Analysis, and Control (CNB-MAC 2019): bioinformatics*. The full contents of the supplement are available online at https://bmcbioinformatics.biomedcentral.com/articles/supplements/volume-21-supplement-14.

Authors' contributions

MAH designed and optimized the model, and carried out the experiments. MAH and SL wrote the manuscript. MAH and SL read and approved the final manuscript.

Funding

This work was supported, in part, by the US National Science Foundation [IIS-1814359]. Salaries and publication costs were funded by the same NSF grant. The funding body did not play any roles in the design of the study and collection, analysis, and interpretation of data and in writing the manuscript.

Availability of data and materials
The code of DEEPLYESSENTIAL is freely available at https://github.com/ucrbioinfo/DeeplyEssential.

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Published: 30 September 2020

References

1. Juhas M, Eberl L, Glass JI. Essence of life: essential genes of minimal genomes. Trends Cell Biol. 2011;21(10):562–8.
2. Hu W, Sillaots S, Lemieux S, Davison J, Kauffman S, Breton A, Linteau A, Xin C, Bowman J, Becker J, Jiang B, Roemer T. Essential gene identification and drug target prioritization in aspergillus fumigatus. PLoS Pathog. 2007;3(3):24.
3. Clatworthy AE, Pierson E, Hung DT. Targeting virulence: a new paradigm for antimicrobial therapy. Nat Chem Biol. 2007;3(9):541–8.
4. Juhas M, Eberl L, Church GM. Essential genes as antimicrobial targets and cornerstones of synthetic biology. Trends Biotechnol. 2012;30(11):601–7.
5. Koonin EV. Comparative genomics, minimal gene-sets and the last universal common ancestor. Nat Rev Microbiol. 2003;1(2):127–36.
6. Chen L, Ge X, Xu P. Identifying essential streptococcus sanguinis genes using genome-wide deletion mutation. Methods Mol Biol. 2015;1279:15–23.
7. Giaever G, Chu AM, Ni L, Connelly C, Riles L, Véronneau S, Dow S, Lucau-Danila A, Anderson K, André B, Arkin AP, Astromoff A, El-Bakkoury M, Bangham R, Benito R, Brachat S, Campanaro S, Curtiss M, Davis K, Deutschbauer A, Entian K-D, Flaherty P, Foury F, Garfinkel DJ, Gerstein M, Gotte D, Güldener U, Hegemann JH, Hempel S, Herman Z, Jaramillo DF, Kelly DE, Kelly SL, Kötter P, LaBonte D, Lamb DC, Lan N, Liang H, Liao H, Liu L, Luo C, Lussier M, Mao R, Menard P, Ooi SL, Revuelta JL, Roberts CJ, Rose M, Ross-Macdonald P, Scherens B, Schimmack G, Shafer B, Shoemaker DD, Sookhai-Mahadeo S, Storms RK, Strathern JN, Valle G, Voet M, Volckaert G, Wang C-Y, Ward TR, Wilhelmy J, Winzeler EA, Yang Y, Yen G, Youngman E, Yu K, Bussey H, Boeke JD, Snyder M, Philippsen P, Davis RW, Johnston M. Functional profiling of the saccharomyces cerevisiae genome. Nature. 2002;418(6896):387–91.
8. Salama NR, Shepherd B, Falkow S. Global transposon mutagenesis and essential gene analysis of helicobacter pylori. J Bacteriol. 2004;186(23):7926–35.
9. Cullen LM, Arndt GM. Genome-wide screening for gene function using RNAi in mammalian cells. Immunol Cell Biol. 2005;83(3):217–23.
10. Liu X, Wang B-J, Xu L, Tang H-L, Xu G-Q. Selection of key sequence-based features for prediction of essential genes in 31 diverse bacterial species. PLoS ONE. 2017;12(3):0174638.
11. Mushegian AR, Koonin EV. A minimal gene set for cellular life derived by comparison of complete bacterial genomes. Proc Natl Acad Sci USA. 1996;93(19):10268–73.
12. Zhang X, Acencio ML, Lemke N. Corrigendum: Predicting essential genes and proteins based on machine learning and network topological features: A comprehensive review. Front Physiol. 2016;7:617.
13. Luo H, Lin Y, Gao F, Zhang C-T, Zhang R. DEG 10, an update of the database of essential genes that includes both protein-coding genes and noncoding genomic elements. Nucleic Acids Res. 2014;42(Database issue):574–80.
14. Ye Y-N, Hua Z-G, Huang J, Rao N, Guo F-B. CEG: a database of essential gene clusters. BMC Genomics. 2013;14:769.
15. Chen W-H, Minguez P, Lercher MJ, Bork P. OGEE: an online gene essentiality database. Nucleic Acids Res. 2012;40(Database issue):901–6.
16. Ning LW, Lin H, Ding H, Huang J, Rao N, Guo FB. Predicting bacterial essential genes using only sequence composition information. Genet Mol Res. 2014;13(2):4564–72.
17. Song K, Tong T, Wu F. Predicting essential genes in prokaryotic genomes using a linear method: ZUPLS. Integr Biol. 2014;6(4):460–9.
18. Yu Y, Yang L, Liu Z, Zhu C. Gene essentiality prediction based on fractal features and machine learning. Mol Biosyst. 2017;13(3):577–84.
19. Plaimas K, Eils R, König R. Identifying essential genes in bacterial metabolic networks with machine learning methods. BMC Syst Biol. 2010;4:56.
20. Acencio ML, Lemke N. Towards the prediction of essential genes by integration of network topology, cellular localization and biological process information. BMC Bioinformatics. 2009;10:290.
21. Lu Y, Deng J, Rhodes JC, Lu H, Lu LJ. Predicting essential genes for identifying potential drug targets in aspergillus fumigatus. Comput Biol Chem. 2014;50:29–40.
22. Cheng J, Xu Z, Wu W, Zhao L, Li X, Liu Y, Tao S. Training set selection for the prediction of essential genes. PLoS ONE. 2014;9(1):86805.
23. Wei W, Ning L-W, Ye Y-N, Guo F-B. Geptop: a gene essentiality prediction tool for sequenced bacterial genomes based on orthology and phylogeny. PLoS ONE. 2013;8(8):72343.
24. Cheng J, Wu W, Zhang Y, Li X, Jiang X, Wei G, Tao S. A new computational strategy for predicting essential genes. BMC Genomics. 2013;14:910.
25. Deng J, Deng L, Su S, Zhang M, Lin X, Wei L, Minai AA, Hassett DJ, Lu LJ. Investigating the predictability of essential genes across distantly related organisms using an integrative approach. Nucleic Acids Res. 2011;39(3):795–807.

26. Palaniappan K, Mukherjee S. Predicting "essential" genes across microbial genomes: A machine learning approach. In: 2011 10th International Conference on Machine Learning and Applications and Workshops, vol. 2. ieeexplore.ieee.org; 2011. p. 189–94. https://doi.org/10.1109/icmla.2011.114.
27. Malod-Dognin N, Pržulj N. GR-Align: fast and flexible alignment of protein 3D structures using graphlet degree similarity. Bioinformatics. 2014;30(9):1259–65.
28. Faisal FE, Newaz K, Chaney JL, Li J, Emrich SJ, Clark PL, Milenković T. GRAFENE: Graphlet-based alignment-free network approach integrates 3D structural and sequence (residue order) data to improve protein structural comparison. Sci Rep. 2017;7(1):14890.
29. Szklarczyk D, Morris JH, Cook H, Kuhn M, Wyder S, Simonovic M, Santos A, Doncheva NT, Roth A, Bork P, Jensen LJ, von Mering C. The STRING database in 2017: quality-controlled protein–protein association networks, made broadly accessible. Nucleic Acids Res. 2017;45(D1):362–8.
30. Hwang S, Kim CY, Yang S, Kim E, Hart T, Marcotte EM, Lee I. HumanNet v2: human gene networks for disease research. Nucleic Acids Res. 2019;47(D1):573–80.
31. Hulo N, Bairoch A, Bulliard V, Cerutti L, De Castro E, Langendijk-Genevaux PS, Pagni M, Sigrist CJA. The PROSITE database. Nucleic Acids Res. 2006;34(Database issue):227–30.
32. Finn RD, Bateman A, Clements J, Coggill P, Eberhardt RY, Eddy SR, Heger A, Hetherington K, Holm L, Mistry J, Sonnhammer ELL, Tate J, Punta M. Pfam: the protein families database. Nucleic Acids Res. 2014;42(Database issue): 222–30.
33. Lin Y, Zhang F-Z, Xue K, Gao Y-Z, Guo F-B. Identifying bacterial essential genes based on a feature-integrated method. IEEE/ACM Trans Comput Biol Bioinform. 2017. https://doi.org/10.1109/tcbb.2017.2669968.
34. Li W, Godzik A. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. Bioinformatics. 2006;22(13):1658–9.
35. Nigatu D, Sobetzko P, Yousef M, Henkel W. Sequence-based information-theoretic features for gene essentiality prediction. BMC Bioinformatics. 2017;18(1):473.
36. Azhagesan K, Ravindran B, Raman K. Network-based features enable prediction of essential genes across diverse organisms. PLoS ONE. 2018;13(12):0208722.
37. Moriyama EN. Codon Usage | Papers in Genetics | Papers in the Biological Sciences | University of Nebraska - Lincoln. https://digitalcommons.unl.edu/bioscigenetics. Accessed: 9 May 2018.
38. Schmidhuber J. Deep learning in neural networks: an overview. Neural Netw. 2015;61:85–117.
39. Li Y, Chen C-Y, Wasserman WW. Deep feature selection: Theory and application to identify enhancers and promoters. J Comput Biol. 2016;23(5):322–36.
40. Shen D, Wu G, Suk H-I. Deep learning in medical image analysis. Annu Rev Biomed Eng. 2017;19:221–48.
41. Finnegan A, Song JS. Maximum entropy methods for extracting the learned features of deep neural networks. PLoS Comput Biol. 2017;13(10):1005836.
42. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res. 2014;15(1):1929–58.
43. Yin H, Gai K. An empirical study on preprocessing High-Dimensional Class-Imbalanced data for classification. In: 2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems. ieeexplore.ieee.org; 2015. p. 1314–9. https://doi.org/10.1109/hpcc-css-icess.2015.205.
44. Li L, Stoeckert Jr CJ, Roos DS. OrthoMCL: identification of ortholog groups for eukaryotic genomes. Genome Res. 2003;13(9):2178–89.
45. Van Dongen SM. Graph clustering by flow simulation. PhD thesis. 2000.
46. Lin Y, Zhang RR. Putative essential and core-essential genes in mycoplasma genomes. Sci Rep. 2011;1:53.

Publisher's Note