eMicrobe
https://www.sciltp.com/journals/eMicrobe

Article

Prediction of Synthetic Lethality in *Escherichia coli* Based on Feature Engineering through Graph Embedding

Qian Xu [1,2,†], Yimiao Feng [3,†], Haixia Guo [1], Yawei Su [1], Xiaoru Chen [4], Haoran Sun [5], Jing Feng [4] and Fengbiao Guo [1,2,*]

[1] School of Pharmaceutical Sciences, Wuhan University, Wuhan 430072, China
[2] Key Laboratory of Combinatorial Biosynthesis and Drug Discovery, Ministry of Education, Wuhan University, Wuhan 430072, China
[3] School of Information Science and Technology, ShanghaiTech University, Shanghai 201210, China
[4] Institute of Artificial Intelligence, School of Computer Science, Wuhan University, Wuhan 430072, China
[5] School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China
* Correspondence: fbguoy@whu.edu.cn
† These authors contributed equally to this work.

How To Cite: Xu, Q.; Feng, Y.; Guo, H.; et al. Prediction of Synthetic Lethality in *Escherichia coli* Based on Feature Engineering through Graph Embedding. *eMicrobe* **2026**, *2*(1), 6. https://doi.org/10.53941/emicrobe.2026.100006

Received: 19 October 2025
Revised: 8 January 2026
Accepted: 12 January 2026
Published: 22 January 2026

Abstract: Synthetic lethality (SL) is a genetic interaction that refers to the phenomenon of cell death caused by the simultaneous inactivation of two non-lethal genes. Due to high-cost constraints and time consumption of experimental screening, computational prediction methods have become the main research tool. Currently, methods based on machine learning have been widely used in SL research, and discovering effective features to enhance the accuracy of predictions remains the key challenge to overcome in current research. We propose an SL prediction method based on graph embedding. First, we transformed five types of raw omics data into graph structures to capture the complex associations among genes. Then, using the graph embedding technique, we extracted feature information for each gene and constructed the feature representation of SL pairs by mathematical operations. Finally, different from GNN, which infers a single graph, we used the machine learning classifiers to discriminate positive and negative samples. Our method achieved better AUC than GNN-based baseline methods. Overall, this study firstly proposed a prediction model for *Escherichia coli* (*E. coli*) SLs that integrates the advantages of graph embedding techniques and classifier ensembles, which significantly improves the accuracy and reliability of prediction, and also provides new perspectives and methods for this field.

Keywords: sythetic lethality; *Escherichia coli*; machine learning; graph embedding

1. Introduction

The concept of synthetic lethality (SL) can be traced back to the early 20th century, when geneticist Calvin B. Bridges, in his study of mutants in the *Drosophila melanogaster*, discovered that some mutations did not cause death when left alone, but when they were inactive with other specific mutations in combination, they caused death [1]. This phenomenon suggests that the deletion of a single gene may not have a significant effect on normal cell growth and division, but when both genes are deleted at the same time, it can lead to organismal or cell death. Studying SL interactions between genes provides a more important perspective to our understanding of the fundamentals of cellular life activities, revealing the interactions and dependencies between genes [2–4]. This contributes to our understanding of disease mechanisms and provides a rationale for developing personalized targeted therapies [5–7]. Currently, databases based on SL have been designed to support the discovery of anticancer drug targets. For instance, the comprehensive knowledge database SynLethDB collects SL gene pairs

across species, and its upgraded version, SynLethDB 2.0, has included additional human SL data such as SLs identified through CRISPR screening, further enhancing the functionality of the database [8,9]. Zhu et al. designed a synthetic lethal and rescue interaction database for microbial genetics called Mslar [10]. In addition, the concept of SL is of great importance in microbial research. Studying SL in microorganisms can help to streamline the genome to obtain a minimal set of genes [11], and then design and construct microorganisms with specific functions and properties for use in bioenergy production [12].

The identification of SL gene pairs is mainly divided into experimental identification and computational predictions. The methods of experimental identification include high-throughput hybridization, RNA interference, gene editing, and other techniques [13,14]. However, experimental identification suffers from limited sample size, high cost, time consumption, and potential off-target effects. In recent years, with the wide application of technologies such as machine learning, there has been an increasing number of methods and tools for computational prediction of SL pairs [15]. Existing computational methods for SL prediction can be further grouped based on their underlying algorithms. Some methods leverage network or graph information; for instance, Li et al. proposed a graphical information centrality metric-based approach to identify SL pairs [16]. Kranthi et al. used functional networks to predict the SL of coding genes [17]. Another group utilizes matrix decomposition techniques; Liany et al. predicted SL interactions by integrating multiple heterogeneous data sources and applying matrix decomposition techniques [18]. SL2MF proposed by Liu et al. is based on logistic matrix factorization (Logistic MF), which combines protein-protein interaction (PPI) data and gene ontology (GO) for SL prediction [19]. More recently, methods incorporating graph neural networks (GNNs) and knowledge graphs (KGs) have emerged. KG4SL is a method for SL prediction that incorporates the knowledge graph (KG) into the graph neural network (GNN) model [20]. The predictive SL method in GCATSL uses a graph-contextualized attention network [21]. KR4SL is an interpretable deep learning model that utilizes knowledge graph reasoning and dynamic programming to identify the SL partner genes for primary genes [22]. Zhu et al. proposed a method of factor-aware knowledge GNN to predict SL in human cancers [23]. MPASL combines attention mechanisms, multi-view learning, and a knowledge graph to predict SL [24]. However, the current identification methods of SL focus on humans, with very few applications reported in microorganisms.

Graph embedding is an effective means to transform the structural information of nodes in a graph into low-dimensional, dense feature vectors suitable for machine learning models. These vectors are capable of capturing nonlinear relationships and higher-order interactions within complex biological systems, thereby enabling the identification of previously overlooked synthetic lethal (SL) pairs. Graph embedding mainly includes matrix factorization-based, random walk-based, and deep learning-based methods [25]. We have employed random walk-based methods in our research because they are more computationally efficient than the other two types of methods, as they do not require complex matrix operations or a large number of parameter training operations. Meanwhile, it is more flexible in capturing diverse features in graph data. The DeepWalk algorithm proposed by Perozzi et al. generates a series of wandering paths by randomly walking through neighboring nodes with equal probability and then forms a node representation [26]. The biased random walk-based Node2vec algorithm takes into account the weight relationship between nodes on this basis [27], which allows for more flexible control of the wandering strategy and provides more comprehensive feature information for the node representation.

Notably, existing graph-based multi-feature fusion methods typically integrate heterogeneous data into a unified graph structure, potentially compromising the structural specificity inherent to individual data modalities [28,29]. To address this limitation, we implement a modality-specific modeling strategy that constructs distinct graph structures for nucleotide sequence information, protein sequence similarity, gene expression profiles, protein-protein interaction networks, and genetic fitness features, thereby preserving the intrinsic topological characteristics of each data type. Regarding feature fusion, conventional deep learning approaches employing simple multilayer perceptions (MLPs) may inadvertently propagate raw data noise into subsequent predictions. Our proposed framework addresses this challenge through a machine learning-based hierarchical feature processing architecture (Figure 1), implementing phased feature selection and optimized fusion to eliminate redundant information and significantly enhance predictive performance effectively.

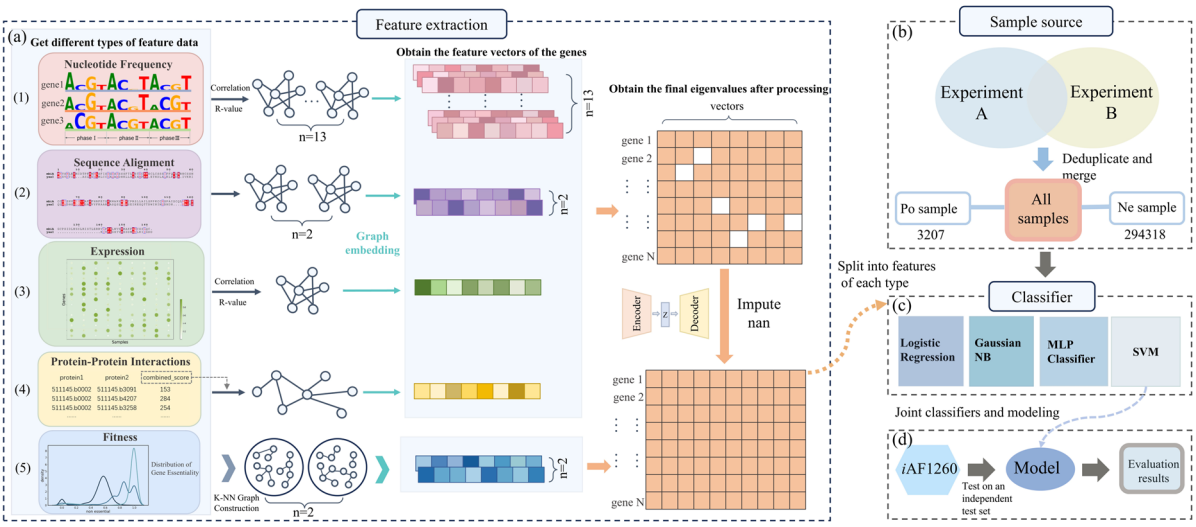

Figure 1. An overview of the methods used for SL prediction of *E. coli*. (**a**) The aim is to convert the five classes of data into a graph form using appropriate methods and obtain a feature representation of each gene based on graph embedding. Missing values are imputed using VAE. **(b)** Positive and negative sample data obtained from two experiments are combined and deduplicated. (**c**) Different types of classifiers are used to compare prediction performance. (**d**) The model is validated using independent sets obtained from the *i*AF1260 metabolic mode (Genome-scale metabolic network model of *Escherichia coli*).

2. Materials and Methods

2.1. Data Collection

We collected data on SL gene pairs from two studies by Côté et al. and French et al. [30,31]. The two studies respectively included 1881 [30] positive samples and 1373 [31] positive samples. Both studies identified SL gene pairs by creating double deletion mutants. By merging and deduplicating these two data sets, we obtained 3207 positive samples and 294,318 negative samples. In preliminary experiments, we observed that the non-augmented dataset demonstrated marginally better performance than SMOTE-augmented data, showing slightly lower AUC but higher F1 scores. Given the significant difference in the number of positive and negative samples, to achieve a relatively balanced sample, we adopted all 3207 positive samples and randomly selected 20,000 pairs from the negative samples. Ultimately, we used 3207 positive samples and 20,000 negative samples for our analysis, involving a total of 3694 genes.

2.2. Different Types of Raw Features

We adopted five types of omics data and utilized them to extract raw features.

2.2.1. Nucleotide Sequence Composition

The nucleotide information of the sequence is the basis for determining the function of genes, and the nucleotide sequence can reflect the association information between genes [32]. A codon is composed of three adjacent bases. According to the position of the bases in the codon, it can be divided into three phases, phase I, II, and III. Taking a single nucleotide as an example, each phase of the base has the possibility of A, C, G, and T, four types of nucleotides. We aim to extract the frequency of these four types of nucleotide characters at each phase, totaling $3 \times 4 = 12$ variables. The formula is as follows:

$$\begin{cases} z_1 = a_1, z_2 = c_1, z_3 = g_1, z_4 = t_1 \\ z_5 = a_2, z_6 = c_2, z_7 = g_2, z_8 = t_2 \\ z_9 = a_3, z_{10} = c_3, z = g_3, z_{12} = t_3 \end{cases} \quad (1)$$

where *a, c, g,* and *t* denote the frequencies of the four nucleotides and 1, 2, and 3 denote the three phases.

For single nucleotides, intervals cannot be formed. For dinucleotides and trinucleotides, we introduce the nucleotide interval *l*, which ranges from 0 to 5, and each set of intervals forms a set of data. For dinucleotides, two nucleotides can only be divided into a single nucleotide character before and after if an interval is formed between them. The interval *l* ranges from 0 to 5, and the number of variables corresponding to each interval is 3×4^2, i.e., 48, and each interval forms a set of independent variables.

For trinucleotides, if there are intervals within the nucleotides, according to the arrangement of the sequences, they can be divided into two basic patterns: one is the "pre-single nucleotide-post-dinucleotide" pattern, and the other is the "pre-dinucleotide-post-single nucleotide" pattern. The number of variables corresponding to interval l of 0 is 3×4^3, i.e., 192 variables, and the number of variables corresponding to each interval is $2 \times 3 \times 4^3$, i.e., 384 variables when the interval l ranges from 1 to 5. A total of 13 sets of variables were formed for our involved oligonucleotides: 1 for single nucleotide, 6 for both dinucleotide and trinucleotide.

2.2.2. Protein Alignment Similarity

Similarly, protein sequences reflect the functional properties of genes, and the similarity between protein sequences can not only reveal the intrinsic linkage of gene functions but also provide clues for understanding potential interactions between genes [33]. Sequence alignment of *E. coli*'s proteome can be used to obtain similarity metrics between different genes. E-value measures the statistical significance of the alignment, while identity value shows the similarity between sequences. First, build a dedicated protein database for all protein sequences of *E. coli* itself. Then, using these two metrics obtained from the BLAST tool, we can quantify the similarities and differences between genes within the genome and thus establish links between gene pairs.

2.2.3. Gene Expression Level

The expression data were downloaded and collected from the Gene Expression Omnibus (GEO) database. After acquiring the gene expression data for *E. coli* from the database, the raw count data were normalized using the Transcripts Per Million (TPM) method to adjust for differences in sequencing depth and library size. Subsequently, a collective aggregation of all samples was conducted, followed by $\log_2$ transformation of the expression values to stabilize variance. The processed dataset comprises 2889 features and has been uploaded to the site (https://github.com/Christal6/ECSL-Predict/ (accessed on 3 March 2025)).

2.2.4. PPI Interaction Strength

Protein-protein interactions (PPIs) not only reflect direct associations between proteins but also provide important molecular-level information for exploring functional associations between gene pairs and identifying SL pairs [34]. The PPI data were downloaded from the STRING database [35].

2.2.5. Gene Fitness Value

The fitness and necessity of a gene are both important indicators to measure the viability of a gene in a specific environment. The lower the fitness and essentiality of a gene, the more critical it is to the growth and development of an organism, and to a certain extent, it reflects the function of the gene [36]. Genes with lower fitness are more likely to form complex interactions with other genes in the genome. Therefore, the fitness of a gene can have an impact on the discovery of synthetic lethal pairs. Therefore, we collected experimental data on gene fitness and used geptop2 [37], a high-precision tool to predict the necessity of the *E. coli* genome [38], to obtain a set of datasets.

Among these five types of feature data, the sequence composition and fitness are, for the first time, converted into a graph structure and extracted as discriminant features for SL prediction.

2.3. Converting Raw Features to Graph Structures

We intend to convert all the relationships between gene pairs of the five types of features into the form of a graph, the graph $G = (V, E)$, where V is the set of n genes involved in each graph, E is the set of linked gene pairs, and the magnitude of the weight of the edges is the scores between the gene pairs. The forms of the features aforementioned, except interaction features and sequence comparison features that are directly represented as gene pairs and their scores, are converted for the data as follows:

2.3.1. Conversion Based on Correlation Coefficients

The Pearson correlation calculation between gene pairs was performed after the expression data were taken as rank values, and finally, the gene pair data with a Pearson correlation coefficient r value greater than 0.7 were retained to construct the data [39].

Nucleotide sequence features were also calculated based on Pearson correlation for the similarity r value of nucleotide frequencies between genes for each of the 13 groups of data, and the top two million gene pair samples

with the highest correlation coefficients *r* were selected for each group by combining the computational volume and correlation considerations.

The formula for calculating the Pearson correlation coefficient *r* is as follows:

$$r = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2 \sum_{i=1}^{n} (y_i - \bar{y})^2}} \quad (2)$$

where two of the genes are represented by variables *X* and *Y*. The corresponding expression or nucleotide frequencies of the genes are: $x_1, x_2, \cdots, x_n$ and $y_1, y_2, \cdots, y_n$.

2.3.2. K-Nearest Neighbors Graph Construction

To transform the gene fitness features, we employed the *K*-nearest neighbor method, commonly used for clustering and classification in machine learning, to uncover patterns and structures in the data [40]. We applied three gene fitness-related features and calculated the inter-gene distances using two metric measures. For each gene, the *K* nearest connections were identified, with the distance magnitude serving as the weight for the gene's connected edges, thereby creating a gene correlation network.

2.4. Node2vec and Producing Topological Features

The random walk selects the next node at each node with the same selection probability for each neighboring node, while the biased random walk, Node2vec, introduces two hyperparameters, p and q, and takes into account the weights of the edges to compute the probability of the node, aiming to find the mapping $f: V \rightarrow \mathbb{R}^d$ mapping node $v \in V$ to a d-dimensional vector of real numbers, and the concrete idea of the implementation is illustrated in Figure 2. Using Node2vec, we can transform the structural-functional information of a gene in a graph into a d-dimensional feature vector representation. Node2vec is a second-order stochastic walk, where the walk of the current node is related to the previous node.

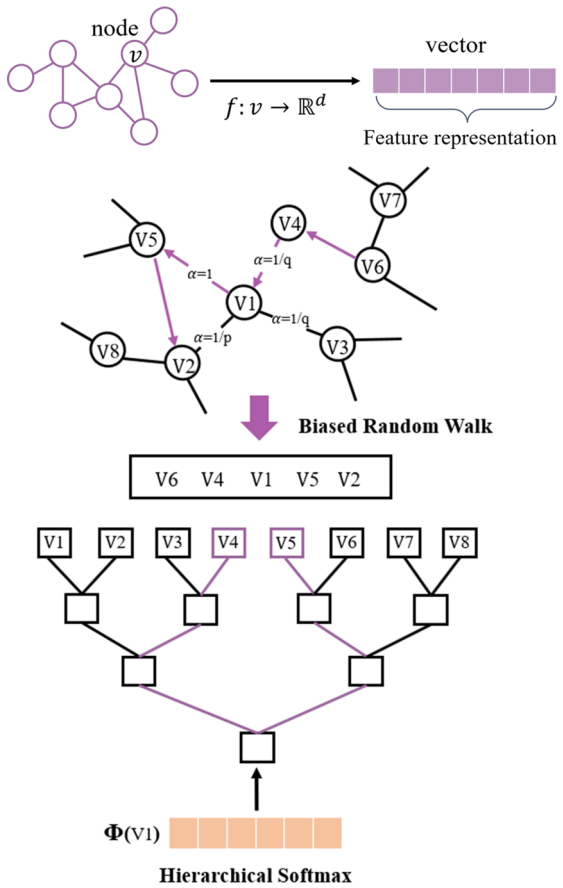

Figure 2. The transformation of embedded data types and the schematic diagram of the core idea of a biased random walk. Node2Vec serializes the graph structure through biased random walks, then learns the rules of nodes in the sequence through the Skip-gram model, uses hierarchical softmax to efficiently optimize model parameters, and finally outputs node embedding vectors that retain network structure features.

Given a graph $G = (V, E)$, assume that a random walk sequence travels from node *u* to node *v*, from node *v* to the next node *x* via edge (v, x), the transfer probability π_{vx} on edge (v, x) is:

$$\pi_{vx} = \frac{w_{vx} \cdot \alpha_{pq}(u, x)}{\sum_{y \in V} w_{vy} \cdot \alpha_{pq}(u, y)} \tag{3}$$

where w_{vx} is the weight on edge (v, x) and the bias term α_{pq} is defined as follows:

$$\alpha_{pq}(u, \mathrm{x}) = \begin{cases} \frac{1}{p} \cdots \text{if } x = u \\ 1 \cdots \text{if } x \neq u \cdot \text{and}(x, u) \in E \\ \frac{1}{q} \cdots \text{if } x \neq u \cdot \text{and}(x, u) \notin E \end{cases} \tag{4}$$

where both p and q are parameters controlling the random walk strategy, p is the return parameterand q is the in-out parameter.

Each graph produced a certain number of embedding features for each gene (Table 1). For each embedding feature, we needed to acquire transformed features for gene pairs according to Equation (5). Finally, gene pairs' features from each data type are input into a classifier. We have five types of raw data and constructed 19 graphs. For *k*-mer, there are 128 × 13 = 1664 features from 13 graphs that are input to the first classifier of machine learning, and so on.

Table 1. The number of features for each of the five categories.

Categories	Number
k-mer	128 × 13 × 2 [a,b]
blast	96 × 2 × 2
express	192 × 1 × 2
ppi	192 × 1 × 2
fitness	128 × 2 × 2

[a] There are 13 graphs corresponding to different pairs of *k* and *l* values; It is similar to the other data types. [b] For each graph embedding feature of single genes, each gene pair has two feature values transformed according to Equation (5); It is the same as the other data types.

For SL pairs and negative samples, the feature vectors of the two genes are involved. Assume that the embedding dimension for genes extracted from a graph is *d*. For gene n and gene m, assume that the feature vector of gene a is $A = (a_1, a_2, \ldots, a_d)$, the eigenvector of gene b is $B = (b_1, b_2, \ldots, b_d)$, the encoding gene pair is characterized as follows.

$$\left(\frac{A + B}{2}, abs(A - B)\right) = \left[\left(\frac{a_1 + b_1}{2}\right), \ldots, \left(\frac{a_d + b_d}{2}\right), |a_1 - b_1|, \ldots, |a_d - b_d|\right] \tag{5}$$

To construct each classifier, we should combine all the transformed features (each graph corresponds to d × 2 dimensions) from all the graphs of one data type as the input. For example, there are 128 × 2 × 13 = 3328 input vectors for the classifier of *k*-mer.

2.5. Data Imputation

For imputation of missing values, common imputation methods include simple imputation methods (such as mean imputation and median imputation), interpolation imputation methods (such as linear interpolation, spline interpolation), and fitting imputation methods [41,42], while VAE imputation has achieved remarkable results in the imputation of DNA methylation missing data and genomic data, etc. [43]. Moreover, compared to the former, VAE can learn the underlying distribution of the data and keep the features as consistent and reasonable as possible globally after filling. Therefore, we choose VAE as the filling method for this study. The principle of VAE involves probabilistic modeling and variational inference [44].

Assuming we have input data x, the VAE aims to learn the latent representation of the data z. The encoder will define a normal distribution of the latent variable z based on the mean μ and standard deviation σ of the input x, $z \sim N(\mu, \sigma^2)$. The decoder then maps z from the distribution sampling back to the reconstructed input $\hat{x}$.

The goal of VAE is to maximize a lower bound on the log-likelihood of the data, called the Evidence Lower Bound (*ELBO*):

$$\mathcal{L}(\theta, \phi, x) = \mathbb{E}_{q_\phi(z|x)}[\log\ p_\theta(x|z)] - D_{KL}(q_\phi(z|x)||p(z)) \tag{6}$$

where θ is a parameter of the decoder and ϕ is a parameter of the encoder. $\mathrm{Log}\, p_\theta(x|z)$ is the probability of generating the data x under the latent variable z. $D_{KL}(q_\phi(z|x)||p(z))$ is the KL scatter between the encoder-defined distribution $q_\phi(z|x)$ and the prior distribution $p(z)$.

2.6. Evaluation Metrics

To evaluate the predictive performance, we used evaluation metrics, including Precision, Sensitivity (Recall), F1-score, the area under a Receiver Operating Characteristic (ROC) curve (AUC), and the area under the precision-recall curve (AUPR). These metrics were defined as follows:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{7}$$

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{8}$$

$$F1 - score = \frac{2 * \text{precision} * \text{sensitivity}}{\text{precision} + \text{sensitivity}} \tag{9}$$

where *TP* denotes the number of correctly predicted positive samples; *TN* denotes the number of correctly predicted negative samples; *FN* denotes the number of incorrectly predicted positive samples; and *FP* denotes the number of incorrectly predicted negative samples. (ROC) curves were obtained by plotting the True Positive Rate (TPR) and False Positive Rate (FPR) at different threshold settings.

3. Results

3.1. Feature Engineering by Graph Embedding of Five Types of Raw Omics Data

We extracted five types of raw data: nucleotide sequence information, protein sequence similarity, gene expression profiles, protein-protein interaction networks, and genetic fitness. Then we transformed them into 19 graphs, respectively. Subsequently, Node2vec was applied to each graph to generate low-dimensional embeddings for gene representations. The embedding dimension *d* will significantly influence the performance of the classifying model [25]. For the selection of embedding dimension *d* when transforming feature vectors using Node2vec, the AUC scores for the embedding dimension *d* ranging from 64 to 224 for features performed from expression, interactions, and sequence comparison are shown in Figure 3a. With the increase of dimension *d*, the AUC score increases significantly at first and then improves slowly. Therefore, we finally adopted the 192 graph embedding features for the omics data of expression level, protein-protein interaction, and 96 for blast alignment. Due to the large amount of data in sequence composition, and considering computational limitations, we set the embedding dimension *d* of the walk features derived from nucleotide frequency to 128, based on trials with the initial three features. The specific number of features used by each category is shown in Table 1.

For the feature extraction part of sequence nucleotide frequency, we take the number of bases *k* composed of oligonucleotides from 1 to 3 and the interval *l* between nucleotides from 0 to 5. For each *k* and *l*, we got the oligonucleotide frequencies and transformed them into one graph based on the correlations of composition frequencies of gene pairs. Combining the graph embedding features of each parameter pair in turn, the change of the AUC scores for the same dataset during the training stage is shown in Figure 3b. As can be seen from the Figure, the prediction results of the fusion of the 13 network features are optimal when *k* is 3, *l* is 5, and the AUC reaches 0.925, which reflects most of the information contained in the sequence. Therefore, for the feature extraction of the nucleotide frequency of the sequence, we finally chose to take the *k* maximum value of 3, and *l* maximum value of 5 for combining sequence network features.

With the fitness features, we employed two ways of measuring distance, Euclidean distance and cosine similarity, to construct the graph, respectively, which comprehensively consider the similarity between different aspects of the data, and improve the accuracy and robustness of the *K* nearest neighbor graph. At the same time, for the selection of the number of neighbors *n* in *K* nearest neighbors, the range of 35~100 with an interval of 5 was tried, and different values of *n* were used to compose the map, and then Node2vec was applied to get the embedding vectors of the genes. The AUC change curve of the five-fold cross-validation is shown in Figure 3c. It can be seen that the AUC scores of cosine similarity and Euclidean distance measures are the highest when *n*-values are 80 and 95, respectively. Then the results decrease with the increase of the *n*-value, which may be due

to the graph structure better capturing the distribution information of the data at these two *n*-values. Furthermore, we also trained models using graph embedding features combined with different values of *n*, and the AUC score trend for each combination of 1, 3, 4, 5, and 10 *n*-values with the two distance measures is shown in Figure 3d and Table S1. It can be observed that the best AUC (0.927) results are obtained when combining only one *n*-value, which may be attributed to the reduced noise and interference, or the diminished influence of important features after merging network features corresponding to different *n*-values.

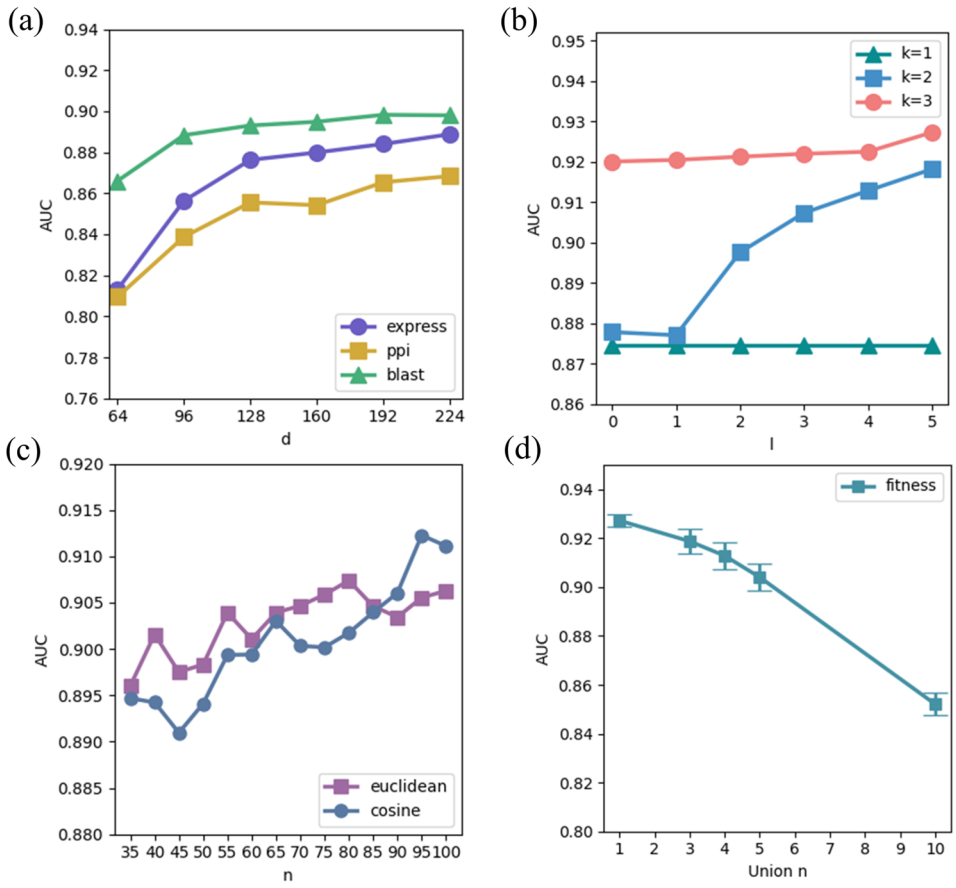

Figure 3. The selection of some parameters. (**a**) The influence curve of the embedding dimension *d* on the model performance. (**b**) The influence curve of the number of nucleotides *k* and interval *l* in the nucleotide sequence on the model. (**c**) The influence of the number of neighbors *n* of the fitness feature. (**d**) The influence of the number of groups of fitness composition, combined with different numbers of neighbors *n*, on the model.

After extracting gene feature vectors for each data category, we noted that the number of genes included in each category was relatively limited compared to the total number of positive and negative samples. For different feature types, we separately trained Support Vector Machine (SVM) classifiers using available positive and negative samples with data through five-fold cross-validation. The resultant classification performance metrics are presented in Table 2. As can be seen, all types of embedding features illustrate similarly good prediction performance.

Table 2. Comparison of results before and after imputing missing values.

	Categories	AUC	AUPR	Precision	Sensitivity	F1
Before Imputation	*k*-mer	0.9252 ± 0.0092	0.7882 ± 0.0169	0.8261 ± 0.0099	0.6049 ± 0.022	0.6982 ± 0.0135
	blast	0.8883 ± 0.0080	0.7220 ± 0.0116	0.8519 ± 0.0142	0.4779 ± 0.0085	0.6122 ± 0.0061
	express	0.8840 ± 0.0152	0.7069 ± 0.0230	0.8499 ± 0.0126	0.4524 ± 0.0161	0.5903 ± 0.0115
	ppi	0.8653 ± 0.0079	0.6564 ± 0.0180	0.7987 ± 0.0326	0.3919 ± 0.0301	0.5254 ± 0.0315
	fitness [a]	–	–	–	–	–
After Imputation	*k*-mer	0.9252 ± 0.0067	0.7896 ± 0.0104	0.8300 ± 0.0152	0.8300 ± 0.0152	0.7021 ± 0.0106
	blast	0.8848 ± 0.0090	0.7207 ± 0.0121	0.8517 ± 0.0103	0.4874 ± 0.0236	0.6197 ± 0.0201
	express	0.8847 ± 0.0034	0.7076 ± 0.0111	0.8407 ± 0.0144	0.4590 ± 0.0218	0.5934 ± 0.0157
	ppi	0.8629 ± 0.0087	0.6657 ± 0.0205	0.8333 ± 0.0199	0.4047 ± 0.0225	0.5445 ± 0.0215
	fitness	0.9272 ± 0.0026	0.7566 ± 0.0077	0.7972 ± 0.0126	0.5778 ± 0.0093	0.6699 ± 0.0066

[a] For fitness, we did not perform the imputation because this feature is complete for all samples.

3.2. Compare the Results Before and After Feature Data Imputation

By utilizing VAE to impute missing values for the four types of features, with the exception of fitness, which had no missing data, and the original feature vectors were complete. This is depicted in Figure 4a, which presents the number of samples utilized for training the genes of the SL pairs before and after the completion of missing values. With five categories of re-representing features (Table 2), a SVM model could be trained and tested. Before and after imputing the missing values, the AUC scores of the SVM were assessed through a five-fold cross-validation process. The findings demonstrate that the AUC scores of each category of feature remain nearly unchanged post-imputation, indicating the efficacy of this imputation method: through the imputation, more samples could be utilized, and the result would be more robust. Among them, sequence composition has the highest AUC, around 0.9252, whereas PPI's AUC is the lowest (around 0.864). After the imputation, the training and test sets contained more samples, making the gene pair set complete. This enhanced the robustness of the model by reducing the bias introduced by missing data.

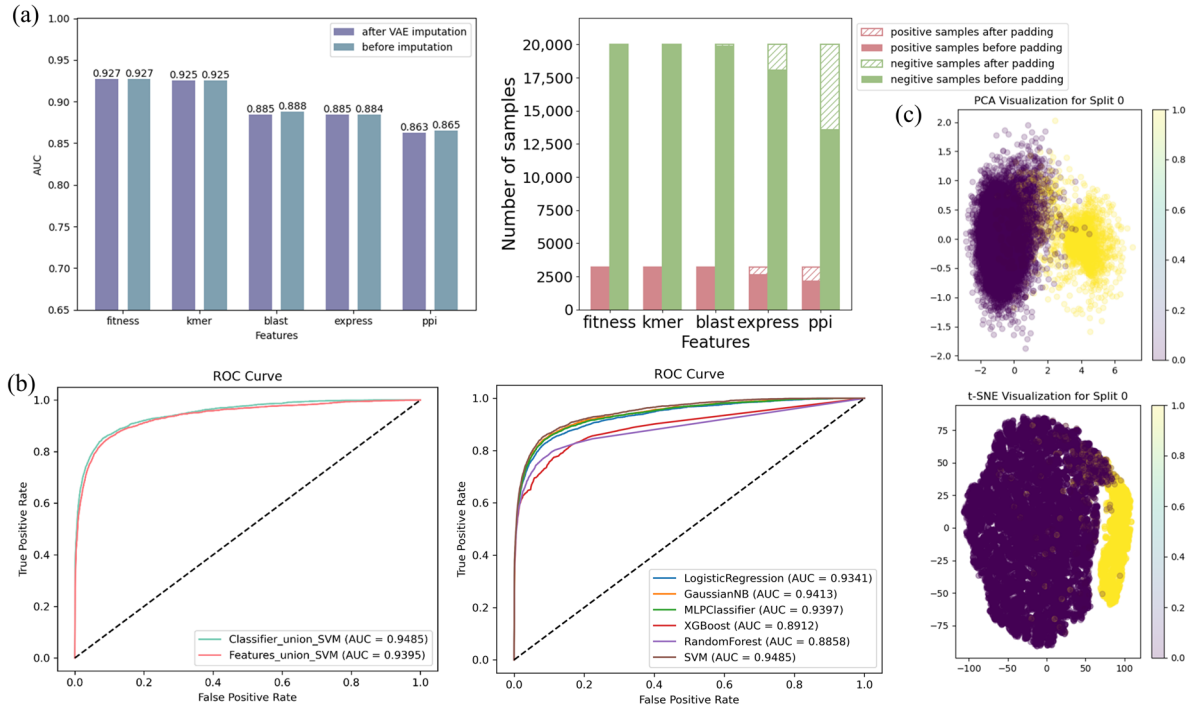

Figure 4. Performance of data imputation and model classification. (**a**) Schematic representation of AUC score changes and sample size changes before and after VAE imputation. (**b**) ROC curves comparing ensemble classifier scores and direct feature fusion, and ROC curves adopting different classifiers when ensemble scores are used. (**c**) Schematic representation of sample scores from five classifiers.

3.3. Performance Evaluation of the Fused Models

After obtaining the complete features of the five categories, all methods (both different feature categories and classifiers) were evaluated using thorough randomized five-fold cross-validation to accurately partition the data into training and testing sets. For each method, the same training sets and test sets are involved.

3.3.1. Comparing the Results of Different Classifiers and Methods

Next, we implemented a feature fusion to establish the final prediction model. We tried the method of classifier score union, which integrates the predicted scores of the classifier output to form a new feature representation. To identify the optimal modeling strategy, we systematically compared different classification algorithms, including Logistic Regression, Gaussian Naive Bayes, Multilayer Perceptron, XGboost, RandomForest, and SVM under consistent experimental conditions. After rigorous five-fold cross-validation with parameter tuning, the SVM classifier demonstrated superior performance with an AUC of 0.9485, establishing itself as the most effective algorithm for processing these fused score features (Table 3). This outcome suggests that SVM better captures the intrinsic relationships within the combined classifier scores than the other methods. In addition, to visualize the performance of these classifiers, Figure 4b provides an intuitive view of the comparison of ROC curves. The results of visualizing the principal features formed by five fused scores for positive and

negative samples, as exemplified by the first division of the five-fold cross-validation with yellow representing positive samples and purple denoting negative samples, show a clear separation of two different categories, with visualization performed using both PCA and t-SNE methods (Figure 4c).

Subsequently, we conducted a controlled comparison with the traditional feature fusion method that directly concatenates features from different sources. To ensure methodological fairness, this baseline approach was evaluated using the same optimized SVM model identified in the previous stage. After comparative analysis, as shown in Figure 4b and Table 3, the method of classifier score union has about 1% improvement (0.9395 to 0.9485) in AUC metrics compared to the traditional feature fusion method under the optimal parameters, which means that the method of classifier score union captures the intrinsic connection between data features more effectively.

Under the benchmark of the complete model's AUC being 0.949, the results of the ablation experiment show that the AUC of the model is maintained between 0.936 and 0.946 (data not shown) after removing any one of the five types of features alone. This result suggests that there may be feature redundancy among the five classes, resulting in insignificant differences in the importance of each feature when removed separately.

Table 3. Evaluation of classifier performance: integrated classifier scores and direct feature fusion results.

Method	Categories	AUC	AUPR	Precision	Sensitivity	F1
Score unions	LogisticRegression	0.9341 ± 0.0054	0.8199 ± 0.0103	0.9263 ± 0.0155	0.5111 ± 0.0223	0.6584 ± 0.0183
	GaussianNB	0.9413 ± 0.0046	0.8289 ± 0.0070	0.8930 ± 0.0213	0.5850 ± 0.0207	0.7066 ± 0.0146
	MLPClassifier	0.9413 ± 0.0050	0.8294 ± 0.0067	0.9489 ± 0.0242	0.4303 ± 0.0765	0.5878 ± 0.0732
	XGBoost	0.8912 ± 0.0170	0.7706 ± 0.0178	0.9491 ± 0.0122	0.4191 ± 0.0239	0.5811 ± 0.0237
	RandomForest	0.8858 ± 0.0066	0.7866 ± 0.0114	0.9690 ± 0.0137	0.5438 ± 0.0224	0.5438 ± 0.0224
	SVM	0.9485 ± 0.0037	0.8425 ± 0.0068	0.8036 ± 0.0299	0.7371 ± 0.0223	0.7683 ± 0.0096
Direct feature fusion	SVM	0.9395 ± 0.0057	0.8189 ± 0.0065	0.8255 ± 0.0145	0.6473 ± 0.0197	0.7254 ± 0.0115

3.3.2. Baseline Comparison in *E. coli*

After obtaining the source code of these models, we trained them using our positive and negative sample sets. All models adopted the identical data partitioning as our proposed method. For fair comparison, we re-collected all required inputs (including GO terms and other features) strictly following each baseline model's original requirements.

We compared our proposed method with the following methods:

- GRSMF leverages graph-regularized self-representative matrix factorization to reconstruct the SL interaction graph, incorporating GO-based functional similarities to enhance the learning process [23].
- SL^{2}MF employs logistic matrix factorization to learn gene latent representations from observed SL data, incorporating gene similarities based on GO annotations and PPI networks to predict SL pairs [19].
- DDGCN introduces a dual-dropout mechanism in a graph convolutional network (GCN) to address overfitting on sparse SL graphs [45].
- SLMGAE utilizes a multi-view graph autoencoder (GAE) to integrate the known SL graph, GO annotations, and PPI data, reconstructing the SL interaction graph for improved prediction accuracy [46].

Our method performs the best among all compared models, as shown in Table 4. With an AUC of 0.9485 and an AUPR of 0.8425, our approach outperforms the second-best model. While our F1 score is slightly lower than SLMGAE, it still surpasses other baselines, including GRSMF, SL^{2}MF, and DDGCN. Our method outperforms existing approaches by leveraging a unique framework that integrates five diverse feature categories transformed into gene-gene graph structures. Unlike matrix factorization-based methods (GRSMF, SL^{2}MF), which rely on gene similarity, or GNN-based models (DDGCN, SLMGAE), which focus on SL graphs and multi-view learning, our approach uses Node2Vec for graph embedding to capture complex gene interactions and a VAE to handle missing data, significantly enhancing robustness.

Table 4. Performance comparison of our method with baselines in AUC, AUPR, and F1 score.

Method	AUC	AUPR	Precision	Sensitivity	F1
GRSMF	0.8622 ± 0.0159	0.7065 ± 0.0222	0.7050 ± 0.0201	0.6583 ± 0.0385	0.6804 ± 0.0195
SL^{2}MF	0.8852 ± 0.0143	0.7215 ± 0.0158	0.7152 ± 0.0215	0.7247 ± 0.0189	0.7197 ± 0.0180
DDGCN	0.8996 ± 0.01090	0.7119 ± 0.0270	0.6712 ± 0.0236	0.7758 ± 0.0326	0.7288 ± 0.0171
SLMGAE	0.9244 ± 0.0084	0.8328 ± 0.0113	0.7541 ± 0.0351	0.7830 ± 0.0377	0.7968 ± 0.0150
Our methods	0.9485 ± 0.0037	0.8425 ± 0.0068	0.8036 ± 0.0299	0.7371 ± 0.0223	0.7683 ± 0.0096

4. Independent Testing

To fully evaluate the performance of our trained SVM model, we employed an independent testing approach. The positive samples were derived from SL pairs predicted based on the *i*AF1260 metabolic model of *E. coli* [47], while the negative samples were derived from the data obtained in the experiment not used in the training [30,31]. From the 69 metabolic model-predicted positives, we removed pairs overlapping with our training set, obtaining 44 positives. These were combined with 274,318 negatives (remaining after removing 20,000 training negatives), forming a final independent test set of 274,362 samples. Subsequently, we applied the complete-samples (3207 positive samples and 20,000 negative samples) model built using the joint classifier scores to this independent test set and performed predictive analysis. Consequently, in the independent test, the model achieved an AUC of 0.821. The results show that the model shows good generalization ability on the independent data sets, and also reflects the efficient ability of the model to distinguish SL pairs. The lower AUC than the five-fold cross-validation may be partly caused by the fact that the metabolic model generated an independent set that probably has false predictions, and the independent test would be considered as a rough reference.

5. Discussions

In this work, we proposed a framework to predict synthetic lethal genes in *E. coli*. Previously, many models were developed for human SLs [19–21]. However, there are scarce reports on microbes. Here, we investigated this issue for the bacterium *E. coli*. Our study adopted five omics datasets and transformed them into a total of 19 graphs. Using the graph embedding method, we extracted 2496 features. Among them, we devised a novel graph construction procedure for the sequence data. For each type of embedded features, we constructed one SVM classifier and combined the outputs of the five classifiers into a final SVM prediction model. This final model could get an AUC of 0.949 in five-fold cross-validation. Our work differs from usual GNN-based studies in that they transformed all graphs into a primary uniform graph. This graph fusion method would compromise the structural specificity inherent to individual data modalities. Compared with these baseline methods [19,23,45,46], our modality-specific modeling strategy illustrated better performance. However, the preliminary preparation work of feature extraction takes a long time and is difficult to popularize. It is difficult to verify our prediction results with experiments. This is also where we need to focus on improvement in the future.

When constructing association networks for different types of features, we face the challenge of data integrity. Due to inconsistencies in the number of genes across different feature types during network construction, some gene information is missing in embedding feature integration. To solve this problem, we adopted VAE to fill in the missing values, a practical model that learns the underlying data distribution to generate and fill in missing points [48]. In our study, the AUC remained relatively stable after VAE imputation compared to the pre-imputation values, indicating that the VAE effectively filled the data gap without compromising data integrity or model discriminative power.

During our research, we raised concerns about the accuracy of the experimental samples. Hence, we utilized the ensemble classifier to predict the positive samples from the two wet experiments and calculated the recall rate, respectively [30,31]. The specific scores are detailed in Figure 3a, and this result may indicate that the second SL experiment, which was performed later, is more reliable than the first. We note that the two experiments are from the same group. One research investigated the synthetic lethal genes coupling with 82 nutrient stress genes [30], while the other concentrated on SLs associated with 111 cell-shape perturbing genes [31]. We think the authors should improve the precision of their experimental screening and produce more reliable results. Initially, we had used the raw expression features to predict the positive samples with the leave-one-out method and found that the higher frequency of positive samples from the second SL group (the remaining positive samples were from the first SL group) performed better on the test set (Figure 5b). These findings suggest that the positive samples from Experiment 1 would not be as accurate as those from Experiment 2. In fact, most wet experiments could not produce 100% accurate sample values [49]. However, computational biologists must use them as a gold standard because wet experiments will provide more confident validation than cross-validation based on different computational predictions [50,51].

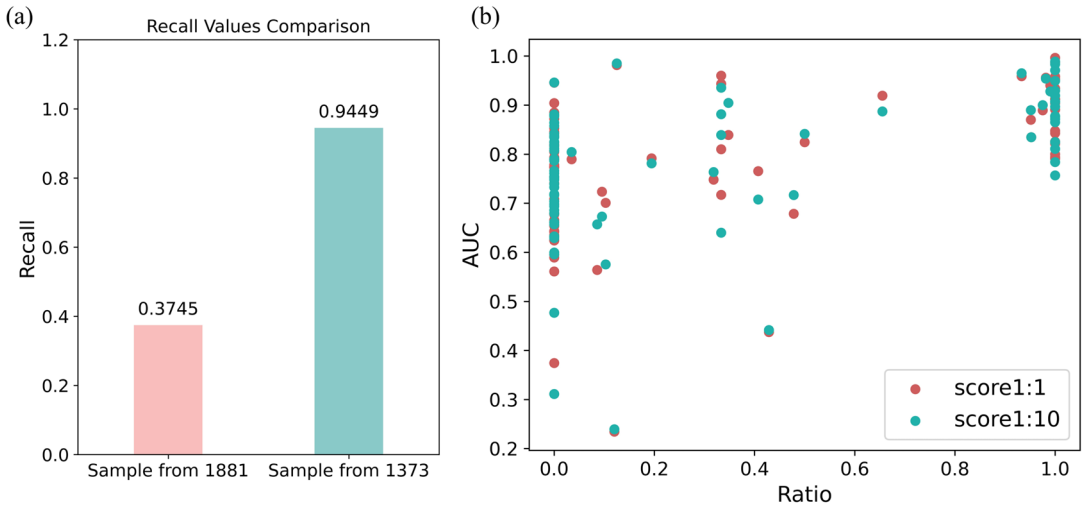

Figure 5. Results of sample accuracy experiments. (**a**) Comparison of recall results from two wet experiments. (**b**) Schematic representation of the performance of the leave-one-out experiment using the original expression data with various positive sample frequencies from Experiment 2. The horizontal axis represents the proportion of genes in the positive samples chosen from Experiment 2. With the change of the proportion, the total number of positive samples remains unchanged. The "1:1" and "1:10" denote the ratio of positive and negative sample sizes.

Supplementary Materials

The additional data and information can be downloaded at: https://media.sciltp.com/articles/others/260121133531367 3/eMicrobe-25100086-Supplementary-Materials.pdf. Figure S1: Example graph of distribution histograms and nuclear density (KDE) curves of degree centrality characteristics for different sample sets. Table S1: The fitness features in conjunction with different n-value and model performance.

Author Contributions

Q.X. and Y.F.: Conceptualization, methodology; Y.S.: validation; H.G.: formal analysis; Q.X.: data curation, writing—original draft preparation; Y.F.: writing—review and editing; X.C.: visualization; H.S. and J.F.: supervision; F.G.: project administration; F.G.: funding acquisition. All authors have read and agreed to the published version of the manuscript.

Funding

This work was funded by the National Natural Science Foundation of China (grant no. 32370696).

Institutional Review Board Statement

Not applicable.

Informed Consent Statement

Not applicable.

Data Availability Statement

All the datasets and source codes implementing our method are uploaded to https://github.com/Christal6/ECSL-Predict/ (accessed on 3 March 2025).

Acknowledgments

We thank Hongtu Cui and Xiang Lian at the Guo lab for their useful discussions and valuable suggestions.

Conflicts of Interest

The author declares no conflict of interest.

Use of AI and AI-Assisted Technologies

No AI tools were utilized for this paper.

References

1. Bridges, C.B. Current Maps of the Location of the Mutant Genes of Drosophila Melanogaster. *Proc. Natl. Acad. Sci. USA* **1921**, *7*, 127–132.
2. Güell, O.; Sagués, F.; Serrano, M.A. Essential Plasticity and Redundancy of Metabolism Unveiled by Synthetic Lethality Analysis. *PLoS Comput. Biol.* **2014**, *10*, e1003637.
3. Sambamoorthy, G.; Raman, K. Understanding the Evolution of Functional Redundancy in Metabolic Networks. *Bioinformatics* **2018**, *34*, i981–i987.
4. Pallotta, M.M.; Di Nardo, M.; Musio, A. Synthetic Lethality between Cohesin and WNT Signaling Pathways in Diverse Cancer Contexts. *Cells* **2024**, *13*, 608.
5. Hartwell, L.H.; Szankasi, P.; Roberts, C.J.; et al. Integrating Genetic Approaches into the Discovery of Anticancer Drugs. *Science* **1997**, *278*, 1064–1068.
6. Sigurdsson, G.; Fleming, R.M.; Heinken, A.; et al. A Systems Biology Approach to Drug Targets in Pseudomonas aeruginosa Biofilm. *PLoS ONE* **2012**, *7*, e34337.
7. Lord, C.J.; Ashworth, A. PARP Inhibitors: Synthetic Lethality in the Clinic. *Science* **2017**, *355*, 1152–1158.
8. Guo, J.; Liu, H.; Zheng, J. SynLethDB: Synthetic Lethality Database Toward Discovery of Selective and Sensitive Anticancer Drug Targets. *Nucleic Acids Res.* **2016**, *44*, D1011–D1017.
9. Wang, J.; Wu, M.; Huang, X.; et al. SynLethDB 2.0: A Web-Based Knowledge Graph Database on Synthetic Lethality for Novel Anticancer Drug Discovery. *Database* **2022**, *2022*, baac030.
10. Zhu, S.-B.; Jiang, Q.-H.; Chen, Z.-G.; et al. Mslar: Microbial Synthetic Lethal and Rescue Database. *PLoS Comput. Biol.* **2023**, *19*, e1011218.
11. Rahiminejad, S.; De Sanctis, B.; Pevzner, P.; et al. Synthetic Lethality and the Minimal Genome Size Problem. *mSphere* **2024**, *9*, e00139-24.
12. Lee, S.J.; Lee, S.-J.; Lee, D.-W. Design and Development of Synthetic Microbial Platform Cells for Bioenergy. *Front. Microbiol.* **2013**, *4*, 92.
13. Yeh, C.-S.; Wang, Z.; Miao, F.; et al. A Novel Synthetic-Genetic-Array–Based Yeast One-Hybrid System for High Discovery Rate and Short Processing Time. *Genome Res.* **2019**, *29*, 1343–1351.
14. Stojic, L.; Lun, A.T.; Mascalchi, P.; et al. A High-Content RNAi Screen Reveals Multiple Roles for Long Noncoding RNAs in Cell Division. *Nat. Commun.* **2020**, *11*, 1851.
15. Wang, J.; Zhang, Q.; Han, J.; et al. Computational Methods, Databases and Tools for Synthetic Lethality Prediction. *Brief. Bioinform.* **2022**, *23*, bbac106.
16. Li, J.; Lu, L.; Zhang, Y.H.; et al. Identification of Synthetic Lethality Based on a Functional Network by Using Machine Learning Algorithms. *J. Cell. Biochem.* **2019**, *120*, 405–416.
17. Kranthi, T.; Rao, S.; Manimaran, P. Identification of Synthetic Lethal Pairs in Biological Systems through Network Information Centrality. *Mol. Biosyst.* **2013**, *9*, 2163–2167.
18. Liany, H.; Jeyasekharan, A.; Rajan, V. Predicting Synthetic Lethal Interactions Using Heterogeneous Data Sources. *Bioinformatics* **2020**, *36*, 2209–2216.
19. Liu, Y.; Wu, M.; Liu, C.; et al. SL2MF: Predicting Synthetic Lethality in Human Cancers via Logistic Matrix Factorization. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2019**, *17*, 748–757.
20. Wang, S.; Xu, F.; Li, Y.; et al. KG4SL: Knowledge Graph Neural Network for Synthetic Lethality Prediction in Human Cancers. *Bioinformatics* **2021**, *37*, i418–i425.
21. Long, Y.; Wu, M.; Liu, Y.; et al. Graph Contextualized Attention Network for Predicting Synthetic Lethality in Human Cancers. *Bioinformatics* **2021**, *37*, 2432–2440.
22. Zhang, K.; Wu, M.; Liu, Y.; et al. KR4SL: Knowledge Graph Reasoning for Explainable Prediction of Synthetic Lethality. *Bioinformatics* **2023**, *39*, i158–i167.
23. Huang, J.; Wu, M.; Lu, F.; et al. Predicting Synthetic Lethal Interactions in Human Cancers Using Graph Regularized Self-Representative Matrix Factorization. *BMC Bioinform.* **2019**, *20*, 657.
24. Zhang, G.; Chen, Y.; Yan, C.; et al. MPASL: Multi-Perspective Learning Knowledge Graph Attention Network for Synthetic Lethality Prediction in Human Cancer. *Front. Pharmacol.* **2024**, *15*, 1398231.
25. Hoang, V.T.; Jeon, H.-J.; You, E.-S.; et al. Graph Representation Learning and Its Applications: A Survey. *Sensors* **2023**, *23*, 4168.

26. Perozzi, B.; Al-Rfou, R.; Skiena, S. Deepwalk: Online Learning of Social Representations. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 24–27 August 2014; pp 701–710.
27. Grover, A.; Leskovec, J. node2vec: Scalable Feature Learning for Networks. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp 855–864.
28. Forster, D.T.; Li, S.C.; Yashiroda, Y.; et al. BIONIC: Biological Network Integration Using Convolutions. *Nat. Methods* **2022**, *19*, 1250–1261.
29. Cho, H.; Berger, B.; Peng, J. Compact Integration of Multi-Network Topology for Functional Analysis of Genes. *Cell Syst.* **2016**, *3*, 540–548.e5.
30. Côté, J.-P.; French, S.; Gehrke, S.S.; et al. The Genome-Wide Interaction Network of Nutrient Stress Genes in Escherichia coli. *mBio* **2016**, *7*, e01714-16.
31. French, S.; Côté, J.-P.; Stokes, J.M.; et al. Bacteria Getting into Shape: Genetic Determinants of E. coli Morphology. *mBio* **2017**, *8*, e01977-16.
32. Minchin, S.; Lodge, J. Understanding Biochemistry: Structure and Function of Nucleic Acids. *Essays Biochem.* **2019**, *63*, 433–456.
33. Duan, Z.-H.; Hughes, B.; Reichel, L.; et al. The Relationship between Protein Sequences and Their Gene Ontology Functions. *BMC Bioinform.* **2006**, *7*, 89.
34. De Las Rivas, J.; Fontanillo, C. Protein–Protein Interaction Networks: Unraveling the Wiring of Molecular Machines within the Cell. *Brief. Funct. Genom.* **2012**, *11*, 489–496.
35. Szklarczyk, D.; Kirsch, R.; Koutrouli, M.; et al. The STRING Database in 2023: Protein–Protein Association Networks and Functional Enrichment Analyses for Any Sequenced Genome of Interest. *Nucleic Acids Res.* **2023**, *51*, D638–D646.
36. Liu, G.; Yong, M.Y.J.; Yurieva, M.; et al. Gene Essentiality is a Quantitative Property Linked to Cellular Evolvability. *Cell* **2015**, *163*, 1388–1399.
37. Wei, W.; Ye, Y.-N.; Luo, S.; et al. IFIM: A Database of Integrated Fitness Information for Microbial Genes. *Database* **2014**, *2014*, bau052.
38. Wen, Q.-F.; Wei, W.; Guo, F.-B. Geptop 2.0: Accurately Select Essential Genes from the List of Protein-Coding Genes in Prokaryotic Genomes. In *Essential Genes and Genomes: Methods and Protocols*; Springer: Berlin/Heidelberg, Germany, 2022; pp 423–430.
39. Hazra, A.; Gogtay, N. Biostatistics Series Module 6: Correlation and Linear Regression. *Indian J. Dermatol.* **2016**, *61*, 593–601.
40. Hassanat, A.B. Two-Point-Based Binary Search Trees for Accelerating Big Data Classification Using KNN. *PLoS ONE* **2018**, *13*, e0207772.
41. Huang, M.-W.; Tsai, C.-F.; Tsui, S.-C.; et al. Combining Data Discretization and Missing Value Imputation for Incomplete Medical Datasets. *PLoS ONE* **2023**, *18*, e0295032.
42. Chen, C.-Y.; Chang, Y.-W. Missing Data Imputation Using Classification and Regression Trees. *PeerJ Comput. Sci.* **2024**, *10*, e2119.
43. Qiu, Y.L.; Zheng, H.; Gevaert, O. Genomic Data Imputation with Variational Auto-Encoders. *Gigascience* **2020**, *9*, giaa082.
44. Kingma, D.P.; Welling, M. Auto-Encoding Variational Bayes. *arXiv* **2013**, arXiv:1312.6114.
45. Cai, R.; Chen, X.; Fang, Y.; et al. Dual-Dropout Graph Convolutional Network for Predicting Synthetic Lethality in Human Cancers. *Bioinformatics* **2020**, *36*, 4458–4465.
46. Hao, Z.; Wu, D.; Fang, Y.; et al. Prediction of Synthetic Lethal Interactions in Human Cancers Using Multi-View Graph Auto-Encoder. *IEEE J. Biomed. Health Inform.* **2021**, *25*, 4041–4051.
47. Dehghan Manshadi, M.; Setoodeh, P.; Zare, H. Rapid-SL Identifies Synthetic Lethal Sets with an Arbitrary Cardinality. *Sci. Rep.* **2022**, *12*, 14022.
48. Singh, A.; Ogunfunmi, T. An Overview of Variational Autoencoders for Source Separation, Finance, and Bio-Signal Applications. *Entropy* **2021**, *24*, 55.
49. Jaksik, R.; Iwanaszko, M.; Rzeszowska-Wolny, J.; et al. Microarray Experiments and Factors Which Affect Their Reliability. *Biol. Direct* **2015**, *10*, 46.
50. Robinson, M.D.; Cai, P.; Emons, M.; et al. Ten Simple Rules for Computational Biologists Collaborating with Wet Lab Researchers. *PLoS Comput. Biol.* **2024**, *20*, e1012174.
51. Li, H.; Sun, X.; Cui, W.; et al. Computational Drug Development for Membrane Protein Targets. *Nat. Biotechnol.* **2024**, *42*, 229–242.