FluxGAT: Integrating Flux Sampling with Graph Neural Networks for Unbiased Gene Essentiality Classification

Kieren Sharma[*], **Lucia Marucci, Zahraa S. Abdallah**
{kieren.sharma, lucia.marucci, zahraa.abdallah}@bristol.ac.uk
School of Engineering Mathematics and Technology,
University of Bristol,
Bristol, UK

Abstract

Gene essentiality, the necessity of a specific gene for the survival of an organism, is crucial to our understanding of cellular processes and identifying drug targets. Experimental determination of gene essentiality requires large growth screens that are time-consuming and expensive, motivating the development of *in silico* approaches. Existing methods predominantly utilise flux balance analysis (FBA), a constraint-based optimisation algorithm; however, they are fundamentally limited by the necessity of a predefined cellular objective function. This requirement introduces an element of observer bias, as the objective function often reflects the researcher's assumptions rather than the cell's biological goals. Here, we present FluxGAT, a graph neural network (GNN) model capable of predicting gene essentiality directly from graphical representations of flux sampling data. Flux sampling removes the need for objective functions, thereby eliminating observer bias. FluxGAT leverages the unique strengths of GNNs in learning representations of complex relationships within metabolic reaction networks. The success of our approach in predicting experimentally determined gene essentiality, with almost double the sensitivity of FBA, explores the possibility of predicting cellular phenotypes in cases when objectives are less understood. Thus, we demonstrate a method for more general gene essentiality predictions across a broader spectrum of biological systems and environments.

1 Introduction

Gene essentiality, defined by the necessity of specific genes for cellular reproduction [1], is critical for understanding the basic requirements of a cell [2], accelerating drug target discovery [3, 4, 5, 6, 7], and guiding the engineering of organisms for chemical production [8]. While advancements in genome sequencing and CRISPR-based genome editing have improved our ability to identify essential genes, experimental methods remain time-consuming, costly, and not always feasible across all organisms. This has led to a shift towards *in silico* approaches for predicting gene essentiality, which analyse intrinsic genomic features, topological features of biological networks [9], and combinations of features using machine learning (ML) [10].

ML approaches involve training classifiers using known *essential* and *non-essential* genes, integrating features such as gene and protein sequences, network topology, gene ontology, and homology [10, 9, 11]. While traditional ML methods like decision trees (DTs) and support vector machines

[*]Corresponding author.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

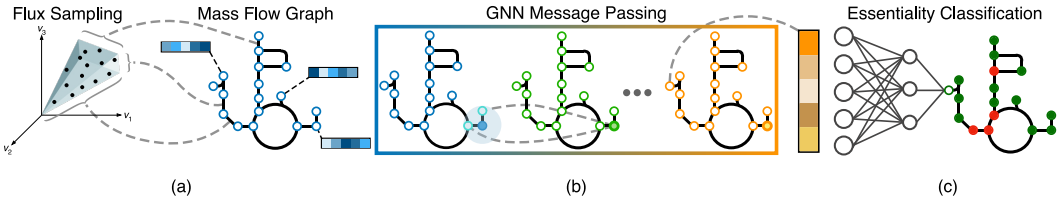


Figure 1: Architectural diagram of FluxGAT - where (a) flux sampling is used to construct a weighted graph of a metabolic reaction network. (b) The adjacency matrix, along with node features containing chemical properties of reactions, are then passed through an optimised graph attention network (GAT) which iteratively generates new representations for each reaction using neural message-passing layers. (c) Lastly, a dense neural layer and sigmoid activation function are used to perform binary classification from each node's final embedding, learning to predict if an enzymatic reaction is essential (red) or non-essential (green) for cell growth.

(SVMs) are commonly used, deep learning neural networks are less frequent due to challenges with interpretability of "black-box" models.

Network-based methods focus on interactions within biological networks and can predict essentiality from structural properties without the need for sequencing data. These methods analyse the topological characteristics of nodes in biological networks, such as protein-protein interaction networks (PINs), to infer gene essentiality. Graph neural networks (GNNs), tailored for graph-structured data [12], are well-suited for tasks requiring an understanding of network relationships, and are therefore likely to become central to single-cell analysis within biomedicine in the coming years [13].

In metabolic engineering, an expanding field of synthetic biology, flux balance analysis (FBA) is arguably the most popular mathematical approach for studying cellular phenotypes and is commonly used to predict gene essentiality. FBA utilises constraint-based modelling (CBM) and linear programming to predict the distribution of metabolites across a metabolic network under steady-state conditions, assuming a predefined cellular fitness objective [14]. However, defining a cellular objective introduces observer bias, as the chosen objective may not fully encapsulate the actual biological goals of an organism [15, 16, 17, 17, 18], which are inherently dynamic and environment-dependent.

Flux sampling addresses this limitation by exploring a wide range of possible flux states without requiring a predefined objective function [19]. Sampling algorithms generate feasible solutions until the solution space is analysed, producing probability distributions of steady-state reaction fluxes for downstream analysis. Flux sampling has demonstrated superior performance over FBA in predicting flux distributions [20, 21], particularly in complex mammalian cell lines where a single objective function may not accurately represent natural conditions.

In this work, we introduce FluxGAT (Figure 1) a novel approach for metabolic gene essentiality prediction, focusing on its capability to learn meaningful representations from flux sampling data derived from the most complete GSMM of Chinese hamster ovary (CHO) cells [22], the most commonly used mammalian hosts in the biopharmaceutical industry [23]. The utilisation of GNNs and flux sampling data is particularly novel as it mitigates the inherent observer bias typically encountered in traditional methods that require the definition of a cellular objective. This innovation is crucial for mammalian and non-model organisms where existing models fall short. The following sections outline the construction of a graphical representation of flux sampling data, where nodes represent enzymatic reactions, the architectural design of FluxGAT, and demonstrate its predictive accuracy through comprehensive evaluations compares to traditional methods.

2 Related Work

ML and Genome-Scale Metabolic Models ML-based methods have demonstrated comparable performance to FBA for predicting metabolic gene essentiality, using only wild-type flux distributions where cellular objectives are more likely to hold [11]. By only applying FBA to the wild-type cell, they eliminate the need to assume that deletion strains optimise the same fitness objective as the wild type. This work utilised four traditional ML classifiers to leverage manually extracted features from a graph-structured representation of metabolic fluxes, where nodes represent enzymatic reactions,

and edges quantify the mass flow of metabolites between these reactions. Although the model could capture most of the essential genes determined experimentally, there was a high rate of false positive predictions.

GNNs and Genome-Scale Metabolic Models The same research group recently introduced an enhancement to their prior work by employing a GNN framework for increased predictive power over conventional ML methods for classifying essentiality using the same graphical representation [24]. Comparative analysis of their model's predictions with FBA reveals a performance comparable to FBA in identifying essential genes. However, it is noteworthy that FBA exhibits significantly superior performance in identifying non-essential genes. The significance of their results lies in the generalisation ability of the model to accurately predict gene essentiality across various growth conditions. A critical limitation; however, is the reliance on the assumption of a wild-type fitness objective, which restricts the model's applicability to model organisms where such behaviour is well-documented. This work motivates the analysis of enzymatic reaction networks in predicting gene essentiality.

GNNs and Protein-Protein Interaction Networks A GNN model was employed to predict gene essentiality using PPI network data [25], significantly outperforming traditional ML and network-based methods. However, challenges arise due to the inherent noise and the prevalence of false positive connections in PPI networks. Additionally, the existence of multiple protein interaction databases, each presenting considerably varied network structures, can lead to inconsistent and unstable outcomes. This highlights the importance of GSMMs as highly utilised tools in systems biology, offering more robust frameworks for studying cellular phenotypes.

Flux Sampling and Gene Essentiality To our knowledge, there is currently no existing work, utilising ML or other methodologies, that focuses on analysing GSMM flux sampling data for the specific aim of predicting gene essentiality. Consequently, this limits the direct comparability of our approach to existing methods, making FBA the primary benchmark for evaluating performance.

3 Methodology

3.1 Task Definition

In this study, we employ a GNN model to address the challenge of binary node label classification to determine the biological essentiality of reactions and their associated genes. Our approach is inherently a semi-supervised task as the GNN leverages the connectivity of the entire network, including test nodes whose features and labels are masked during training. The data utilised is a single graph, $G = (V, E)$, which is a reconstruction of the CHO cell metabolic reaction network using flux sampling data, where V is the set of nodes (reactions) and E is the set of edges (interactions). The features used by our model, denoted as X, represent the network topology with weighted edges alongside node features that encapsulate the chemical properties of the reactions. The binary node labels, Y, being predicted, signify experimentally determined essentiality values. The graph's data structure, comprising a weighted and directed graph, reflects the intricacies of metabolic pathways, correlating the network topology and node features directly to the essentiality of reactions. This setup allows for a comprehensive understanding and prediction of gene essentiality.

3.2 Flux Sampling

When modelling reaction networks within an organism at the genome scale, which often involves thousands of reactions and metabolites, the classical approach in systems biology of dynamical modelling becomes infeasible due to the time and resources required to obtain kinetic parameters experimentally.

Constraint-based modelling is an approach to analysing such networks without detailed kinetic data. For a given reaction network, the relation between the m-metabolites and n-reactions is described in the mxn stoichiometric matrix S. A positive stoichiometric coefficient $S_{i,j}$ means that metabolite i is produced by reaction j, and a negative entry indicates that the metabolite is consumed in that reaction. The primary constraints limit the rate and directionality of each reaction in the network.

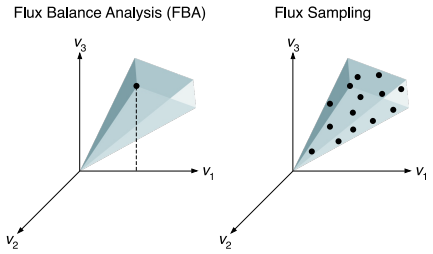


Figure 2: Flux balance analysis (FBA) generates a single non-unique solution, and flux sampling generates a probability distribution of feasible solutions of the constrained solution space (blue region) of reaction fluxes v_1, v_2, v_3.

Then, a steady-state assumption ensures that all metabolite concentrations stay constant over time (mass balance). The dynamics of the network at steady-state are defined as,

$$\frac{d\mathbf{x}}{dt} = S\mathbf{v} = 0, \qquad \mathbf{v}^{lb} \leq \mathbf{v} \leq \mathbf{v}^{ub}. \tag{1}$$

Where $\mathbf{x}$ is a vector of metabolite concentrations and $\mathbf{v}$ is a vector of flux rates, constrained by lower and upper bounds for each reaction. FBA introduces an objective function and uses linear programming to find a non-unique flux distribution $\mathbf{v}^*$, which satisfies the constraints. Flux sampling, on the other hand, aims to uniformly sample the feasible solution space, given the constraints, avoiding the need to specify an objective function. Figure 2 visualises this distinction.

The Hit-and-Run (HR) algorithm is a deterministic Markov chain Monte Carlo (MCMC) method that has emerged as the primary means for sampling large, genome-scale networks due to its efficiency and convergence time [26]. In this work, an implementation of the artificial centring hit-and-run (ACHR) algorithm, known as *optGpSampler* [27], was used due to its established application to GSMMs.

In the first iteration ($a = 0$) of the ACHR algorithm, an arbitrary starting point $\mathbf{v}^{(0)} \in P$, is selected within an N dimensional convex set P, giving an initial centre of $\hat{\mathbf{c}} = \mathbf{v}^{(0)}$. The algorithm then generates a chain of samples by iteratively performing the following three steps. First, computing a direction to travel in $\boldsymbol{\theta}$,

$$\boldsymbol{\theta}^{(a)} = \frac{\mathbf{v}^{(i)} - \hat{\mathbf{c}}}{\left\|\mathbf{v}^{(a)} - \hat{\mathbf{c}}\right\|}, \quad \text{where } i \sim U\{0, 1, \ldots, a\}. \tag{2}$$

Then, after randomly generating a step size $\lambda^{(a)}$, a new sample is calculated $\mathbf{v}^{(a+1)} = \mathbf{v}^{(a)} + \lambda^{(a)}\boldsymbol{\theta}^{(a)}$. Lastly, updating the artificial centre by setting,

$$\hat{\mathbf{c}} = \frac{a\hat{\mathbf{c}} + \mathbf{v}^{(a)}}{a+1}. \tag{3}$$

This process is repeated until convergence, as confirmed by diagnostic tests, resulting in probability distributions of possible flux values for each reaction. Then, a single flux distribution is calculated by averaging the probability distribution for each reaction to generate an m-dimensional flux vector $\bar{\mathbf{v}}$, facilitating downstream analysis, as was done in [20].

3.3 Graph Construction

There are numerous ways to construct a graphical representation of a given metabolic reaction network using its stoichiometry; however, many fail to capture the directionality of reactions and are dominated by pool metabolites that appear in many reactions, such as water. The mass flow graph (MFG) was designed to address these common limitations by utilising flux distributions to encode the directionality of metabolic flows whilst mitigating the over-representation of pool metabolites [28], producing a directed graph. The MFG also exhibits systemic changes to its topological and

community structure under environmental and genetic perturbations, revealing redundancies and core structures within a reaction network. It is, therefore, well-suited for highlighting essential components.

An MFG can be constructed from a stoichiometry matrix S whereby nodes represent reactions and a directed edge connects two nodes if the source reaction produces a metabolite consumed by the target reaction. As detailed in [28], starting with flux vector, obtained either through FBA $\mathbf{v}^*$, or in our case, averaged across flux sampling iterations $\bar{\mathbf{v}}$, we first unfold the vector into the forward and reverse components of each reaction,

$$\bar{\mathbf{v}}_{2m} = \begin{bmatrix} \bar{\mathbf{v}}^+ \\ \bar{\mathbf{v}}^- \end{bmatrix} = \frac{1}{2} \begin{bmatrix} \mathrm{abs}\,(\bar{\mathbf{v}}) + \bar{\mathbf{v}} \\ \mathrm{abs}\,(\bar{\mathbf{v}}) - \bar{\mathbf{v}} \end{bmatrix}. \tag{4}$$

Then, we can define the modified stoichiometry matrix as,

$$\mathbf{S}_{2m} = \begin{bmatrix} \mathbf{S} & -\mathbf{S} \end{bmatrix} \begin{bmatrix} \mathbf{I}_m & 0 \\ 0 & \mathrm{diag}\,(\mathbf{r}) \end{bmatrix}, \tag{5}$$

where r is an m-dimensional Boolean reversibility vector such that $r_j = 1$ if reaction j is reversible and 0 otherwise. Lastly, the adjacency matrix of the MFG is given by,

$$\mathbf{M}\,(\bar{\mathbf{v}}) = \left(\mathbf{S}^+_{2m}\bar{\mathbf{V}}\right)^T \mathbf{J}^\dagger_v \left(\mathbf{S}^-_{2m}\bar{\mathbf{V}}\right), \tag{6}$$

where $\bar{\mathbf{V}}$ and $\mathbf{J}_v$ are defined as $\mathrm{diag}\,(\bar{\mathbf{v}}_{2m})$ and $\mathrm{diag}\left(\mathbf{S}^+_{2m}\bar{\mathbf{v}}_{2m}\right)$ respectively, and $\dagger$ denotes the matrix pseudoinverse, with,

$$\begin{array}{ll} \text{Production}: & \mathbf{S}^+_{2m} = \frac{1}{2}\left(\mathrm{abs}\,(\mathbf{S}_{2m}) + \mathbf{S}_{2m}\right), \\ \text{Consumption}: & \mathbf{S}^-_{2m} = \frac{1}{2}\left(\mathrm{abs}\,(\mathbf{S}_{2m}) - \mathbf{S}_{2m}\right). \end{array} \tag{7}$$

3.4 Node Label Generation

Considering that the MFG comprises a set of reactions as nodes rather than a set of genes, we must initially map the gene essentialities onto the corresponding set of reaction essentialities to train and test a classifier. To perform this mapping, the method outlined in [29] is utilised, where the authors address the inconsistent treatment of gene protein reaction (GPR) rules by defining a standard approach to handling them. GPR rules describe how genes relate to protein complexes and the reactions they catalyse. They contain Boolean logic operators that allow us to classify reactions as active/inactive using gene expression.

These GPR rules can also be processed to translate gene essentiality into node essentiality labels according to the following function f, applied to each reaction,

$$f(\oplus, g_1, \ldots, g_n) = \begin{cases} \max(g_1, \ldots, g_n) & \text{if only ANDs} \\ \min(g_1, \ldots, g_n) & \text{if only ORs} \\ \text{Resolve ANDs} \\ \text{then ORs} & \text{if mixed} \end{cases} \tag{8}$$

where,

- $\oplus$: Represents the logical operator or the combination of operators within the GPR rule for a given reaction.
- $g_1, \ldots, g_n$: The sequence of gene essentiality values (1 for essential and 0 for non-essential).
- *Resolve ANDs then ORs*: For mixed operators, standard logical precedence rules apply (AND operations are typically evaluated before OR operations unless parentheses indicate otherwise).

3.5 Node Feature Generation

GNNs exhibit state-of-the-art performance across various network-based tasks, primarily due to their ability to combine representations of network topology with *side information* such as node features. This section outlines the node (reaction) feature generation process to create embeddings that complement the adjacency matrix, M.

Our generated node features contain biologically meaningful information for the downstream classification task of essentiality prediction. This approach involved compiling a list of the reactants and products of each reaction, representing additional chemical properties not captured within the MFG. These lists encode each reaction's starting material and end result in a structured format, emphasising relationships between nodes.

A one-hot encoding was employed to transform node features into a numerical format suitable for a GNN. To achieve this, we let R be the set of all reactants and products across reactions, with $|R|$ being its cardinality. For each reaction i, a one-hot encoded vector $\mathbf{o}_i$ is constructed, where $\mathbf{o}_i \in \{0,1\}^{|R|}$. For each reactant or product j in R, the corresponding element in $\mathbf{o}_i$, denoted as o_{ij}, is defined as:

$$o_{ij} = \begin{cases} 1 & \text{if reactant/product } j \text{ is involved in reaction } i, \\ 0 & \text{otherwise.} \end{cases}$$

3.6 Graph Representation Learning

We use a graph neural network (GNN) model to perform representation learning on the MFG due to its ability to combine features of individual reactions with the interaction patterns defining the reaction network topology. Specifically, we employ a graph attention network (GAT) [30], which integrates self-attention mechanisms, allowing the model to weight neighbouring nodes based on their relevance to a given reaction in the context of the learning task.

Formally, GNNs contain *neural message passing* layers where vector messages are exchanged between nodes and updated using neural networks. In each message-passing layer, the hidden embedding $\mathbf{h}_u^{(i)}$ for node $u \in \mathcal{V}$ is updated by aggregating information from its neighbourhood $\mathcal{N}(u)$:

$$\begin{aligned} \mathbf{h}_u^{(i+1)} &= \text{UPDATE}^{(i)}\Big(\mathbf{h}_u^{(i)}, \\ &\qquad \text{AGGREGATE}^{(i)}(\{\mathbf{h}_v^{(i)}\, \forall v \in \mathcal{N}(u)\})\Big) \qquad (9a) \\ &= \text{UPDATE}^{(i)}\left(\mathbf{h}_u^{(i)}, \mathbf{m}_{\mathcal{N}(u)}^{(i)}\right), \qquad (9b) \end{aligned}$$

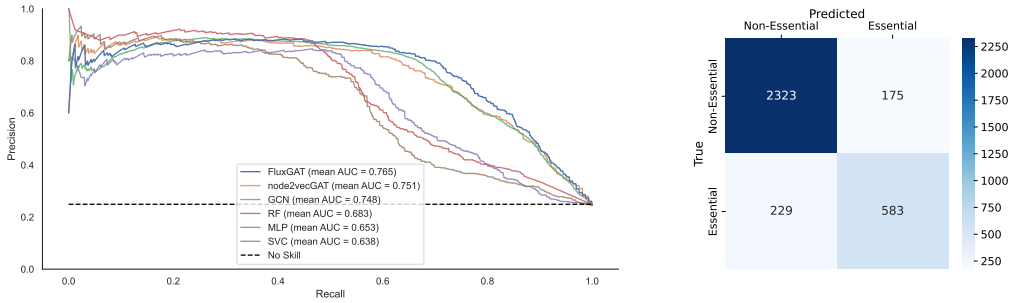
where UPDATE and AGGREGATE are arbitrary differentiable functions and $\mathbf{m}_{\mathcal{N}(u)}$ is the "message" that is aggregated from u's one-hop graph neighbourhood $\mathcal{N}(u)$ at iteration i. Here, we use a modified version of the original GAT, assigning a learnable attention weight to each neighbour based on node and edge features. This attention weight determines the *importance* of nodes during the aggregation step, with each message defined as,

$$\mathbf{m}_{\mathcal{N}(u)} = \sum_{v \in \mathcal{N}(u)} \alpha_{u,v} \mathbf{h}_v, \qquad (10)$$

where $\alpha_{u,v}$ denotes the attention given to neighbour $v \in \mathcal{N}(u)$ during aggregation at node u. The attention weights are calculated based on the importance of the features of the source node, the target node, and the edge, as follows,

$$\alpha_{u,v} = \frac{\exp\left(\mathbf{a}^\top \left[\mathbf{W}\mathbf{h}_u \oplus \mathbf{W}\mathbf{h}_v \oplus \mathbf{W}\mathbf{e}_{u,v}\right]\right)}{\sum_{v' \in \mathcal{N}_(u)} \exp\left(\mathbf{a}^\top \left[\mathbf{W}\mathbf{h}_u \oplus \mathbf{W}\mathbf{h}_{v'} \oplus \mathbf{W}\mathbf{e}_{u,v'}\right]\right)}, \qquad (11)$$

where $\mathbf{a}$ is a trainable attention vector, $\mathbf{W}$ are trainable matrices unique to each feature, and $\oplus$ denotes the concatenation operation. Multi-dimensional edge features between nodes u and v are represented



(a) (b)

Figure 3: (a) Precision-recall (PR) curves for FluxGAT, node2vecGAT, GCN and three standard binary classifiers, averaged across 25 evaluations (5 repetitions of 5-fold cross-validation). The dashed line represents the precision (0.249) of the no-skill classifier given the class imbalance. (b) Confusion matrix showing the average classification performance of FluxGAT on the 3,310 experimentally determined reaction essentiality labels.

by $\mathbf{e}_{u,v'}$, which, in our case, are equal to scalar edge weights of the MFG representing chemical mass flow between reactions.

The initial node embeddings at $i = 0$, $\mathbf{h}_u^{(0)}$, are set to the one-hot encodings detailed in Section 3.5. After running N iterations of the GNN message passing, we can use the output of the final layer to define the embeddings for each node, i.e.,

$$\mathbf{z}_u = \mathbf{h}_u^{(N)}, \forall u \in \mathcal{V}. \tag{12}$$

Then, to perform node classification, $\mathbf{z}_u$ is passed through a dense neural layer to generate a single logit z_u, which is passed through a sigmoid activation function to determine the binary class (essentiality) for node (reaction) u.

4 Experiment

4.1 Dataset

The artificial centring hit-and-run (ACHR) algorithm, detailed in Section 3.2, was applied to the wild-type iCHO2291 GSMM of CHO cells [22], which contains $m = 3,972$ metabolites and $n = 6,236$ reactions. Sampling was performed [2] for 50,000 iterations with a *thinning* value of 1000 to reduce the memory footprint of the sampling results whilst still ensuring meaningful convergence [31]. Averaging the probability distributions for each reaction resulted in a single flux vector $\bar{\mathbf{v}}$ with 5,316 non-zero reaction fluxes. According to the methods in Section 3.3, an MFG was constructed containing 4,733 nodes. The MFG also contains 335,024 weighted edges, representing the directionality and strength of metabolite flow between reactions. Function f (Equation 8) was applied to generate binary essentiality labels for nodes in the MFG using experimental gene essentiality values taken from a genome-wide CRISPR knockout screen of CHO cells [32]. This study identified 1,980 genes out of the 15,028 targeted which negatively affect cell proliferation and are therefore deemed essential. The iCHO2291 model contains 2,291 functionally modelled genes representing CHO cell metabolism, 392 of which are essential according to [32]. Applying f to all reactions within the model containing GPR rules (4,182), classified 3,296 reactions as non-essential and 886 as essential. These reaction essentiality values allowed us to generate node labels for 3,310 nodes within the MFG, with 2,498 being non-essential (0) and 812 essential (1).

Table 1: Comparative performance metrics providing a detailed overview of the mean and standard deviation for test accuracy, precision, recall, and F1 score, aggregated over 25 evaluations (comprising five repetitions of 5-fold cross-validation) for each binary classifier.

Method	Accuracy	Precision	Recall	F1 Score
FluxGAT	$\mathbf{0.871 \pm 0.012}$	0.769 ± 0.030	$\mathbf{0.718 \pm 0.037}$	$\mathbf{0.743 \pm 0.023}$
node2vecGAT	0.864 ± 0.014	0.741 ± 0.040	0.693 ± 0.038	0.714 ± 0.023
GCN	0.870 ± 0.013	0.754 ± 0.030	0.699 ± 0.032	0.725 ± 0.030
RF	0.848 ± 0.010	0.786 ± 0.040	0.520 ± 0.041	0.625 ± 0.039
MLP	0.854 ± 0.011	0.813 ± 0.040	0.528 ± 0.028	0.639 ± 0.027
SVC	0.827 ± 0.010	$\mathbf{0.843 \pm 0.050}$	0.366 ± 0.027	0.509 ± 0.025

4.2 Model Architecture and Training

FluxGAT (Figure 1) utilises a supervised learning approach to discern and comprehend complex patterns within node features and relationships within the MFG, as can be seen in Figure S3 of the supplementary material. This training method enabled the model to learn biologically meaningful representations to predict essentiality labels. During evaluation, the graph was partitioned into training (80%) and testing (20%) sets, ensuring node features designated for testing were masked during the training phase.

We employed k-fold cross-validation to partition the graph systematically into five distinct folds for training (2,648 nodes) and testing (662 nodes), ensuring each fold served as a testing set once. This procedure allowed every reaction in the graph to be part of both the training and testing sets across different iterations, mitigating potential biases and testing the generalisability of the model. FluxGAT contains an initial embedding layer to transform the sparse one-hot gene vectors into a lower-dimensional, continuous space, facilitating more efficient and meaningful feature processing in the subsequent GAT message-passing layers.

To fine-tune the performance of FluxGAT, a grid-search method was employed for hyperparameter optimisation of the underlying GAT architecture [2]. The optimal architecture is initialised with 150-dimensional embeddings to transform 3,972-dimensional one-hot encodings into dense vectors. Two GAT message-passing layers, containing multi-head attention (2 heads) and 150 hidden channels, transform the dense node embeddings into a final representation, which is fed to a fully connected layer to consolidate the features into a single logit per node, setting the stage for binary classification using a sigmoid activation function with an optimised threshold of 0.65. A weighted loss function helps to address the class imbalance in essentiality labels within the MFG to ensure a more balanced treatment of each class. The weight was determined based on the ratio of non-essential to essential reactions, precisely the number of zeros (2,498) to the number of ones (812). For training FluxGAT, the AdamW optimiser was employed to improve generalisation performance [33], complemented by regularisation strategies to prevent overfitting, which included weight decay, dropout and an early-stopping procedure.

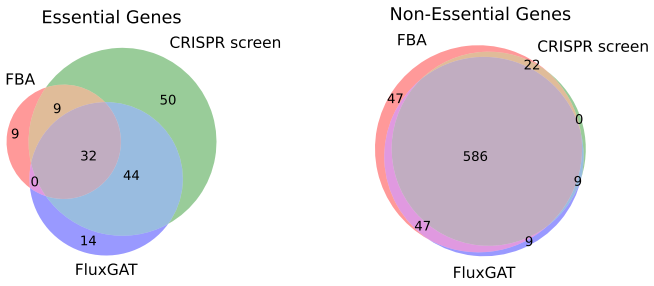
4.3 FluxGAT Performance

We begin by comparing FluxGAT against three standard binary classifiers: support vector classifier (SVC), multilayer perceptron (MLP), and random forest (RF). The performance of each model, including accuracy, precision, recall, and F1 score, was averaged across five repetitions of 5-fold cross-validation to ensure robustness and consistency in our evaluation.

A manual feature extraction process was employed to generate node features from the MFG to implement the standard binary classifiers. This process involved creating vectors for each node, composed of the incoming and outgoing edge weights associated with each node, effectively capturing the flow of metabolites into and out of each metabolic reaction's one-hop neighbourhoods.

Figure 3a presents the precision-recall (PR) curves for FluxGAT and the baseline classifiers, averaged across the 25 evaluations. This choice of PR curves was due to the class imbalance in the MFG, as they provide a more informative performance measure under such conditions compared to traditional

[2] See supplementary material for a detailed description of hyperparameter optimisation, computational resources, software, and libraries used to conduct experiments. Full source code is also provided.



(a) (b)

Figure 4: Comparison of gene essentiality predictions with experimental CRISPR screen, for genes involved in MFG reactions with a unique inverse mapping. (a) Essential gene predictions by FBA and FluxGAT. (b) Non-essential gene predictions by the same methods.

receiver operating characteristic (ROC) curves. FluxGAT achieves the highest precision-recall area under the curve (PR-AUC) at 0.765, a significant improvement over the *no-skill* classifier, which has a precision of 0.249, indicative of the class imbalance.

In addition to comparing FluxGAT with standard binary classifiers, we further evaluated its performance against the same GAT architecture but implemented with *node2vec* embeddings [34]. This comparison allows us to determine the impact of the custom embeddings on FluxGAT's performance. The node2vecGAT model serves as a GNN benchmark, as it represents a well-established method for generating node embeddings based on graph structure but without the domain-specific feature engineering employed in FluxGAT. Lastly, we evaluated the performance of our framework by substituting a graph convolutional network (GCN) for the GAT architecture. Both GNN benchmarks surpassed the standard binary classifiers, underscoring the efficacy of the message-passing framework detailed in Section 3.6 in surpassing manual feature extraction for creating topological representations of nodes. Notably, FluxGAT, with its incorporation of attention mechanisms and custom node features, achieved a significant enhancement in Precision-Recall Area Under the Curve (PR-AUC). Table 1 compares FluxGAT's performance to baseline models.

4.4 Comparison with Flux Balance Analysis

This section details converting FluxGAT's binary reaction predictions into gene essentiality labels. To perform this conversion, we applied the inverse function of f (referenced in Equation 8) to the reactions where GPR rules allow a unique inverse mapping. Specifically, this includes reactions with one-to-one mappings to genes, essential reactions governed only by OR operators, and non-essential reactions exclusively involving AND operators in their GPR rules. Out of the 2,085 genes present in the MFG, 752 are involved in uniquely invertible mappings.

Applying this inverse mapping to reactions labelled by FluxGAT, we can compare the essentiality classification of genes to those made by FBA (also across five repetitions). Figure 4 shows the agreement between FluxGAT and FBA gene essentiality labels with those determined experimentally by a genome-wide CRISPR screen [32]. FluxGAT can identify 32 of the 41 essential genes correctly labelled by FBA and an additional 44 genes missed by FBA. However, this comes at the cost of a further 14 false-positive predictions. FluxGAT can also identify nearly all non-essential genes correctly labelled by FBA but with only 56 false negatives instead of 94.

As observed by Strain et al. [20], the iCHO2291 GSMM displays very high specificity, rarely classifying a non-essential gene as essential, but shows lower sensitivity, failing to capture all essential genes when using FBA. We also observed this in our analysis when classifying the 752 genes in the MFG, whereby FBA displays a specificity of 0.985 but a sensitivity of 0.304. FluxGAT shows similar specificity, with 0.977, but an increased sensitivity of 0.576, capturing nearly double the number of essential genes as FBA.

5 Discussion

In this research, we introduced FluxGAT, a GNN-based approach for predicting metabolic gene essentiality without the observer bias inherent in FBA, the most commonly used method in systems biology. Applying FluxGAT to Chinese hamster ovary (CHO) cells demonstrated a significant advancement in accurately predicting gene essentiality by learning from flux sampling distributions and metabolic reaction network topology.

The core innovation of FluxGAT lies in combining data derived from flux sampling with a graph attention network framework. This methodology enables prediction based on intrinsic cellular characteristics, eliminating the need for predefined cellular objectives. This feature is particularly advantageous for studying mammalian and non-model organisms where objectives are not well-defined, offering an unbiased approach to predicting cellular phenotypes.

Our findings show that FluxGAT surpasses FBA in terms of sensitivity when applied to a widely used mammalian cell line, enhancing applications ranging from identifying drug targets to improving outcomes in metabolic engineering. However, the model's complexity and computational resource requirements may pose challenges for scaling and broader applications.

Future research will focus on assessing FluxGAT's performance across various cell types and environmental conditions to verify its generalisation abilities. This presents challenges due to the scarcity of lab-based genome-wide growth screens. We are also interested in examining the attention weights learned by FluxGAT when predicting the essentiality of nodes. By interpreting these weights, we aim to uncover the biological significance of each node within the context of the entire reaction network.

In conclusion, FluxGAT's ability to predict gene essentiality directly from genotype, free from observer bias, demonstrates a method for more general gene essentiality prediction across diverse biological systems. This advancement highlights the significant potential of deep learning techniques in enhancing tools within systems biology, representing a substantial stride toward advancing *in silico* methods used in personalised medicine and targeted drug development.

References

[1] G. Rancati, J. Moffat, A. Typas, and N. Pavelka, "Emerging and evolving concepts in gene essentiality," *Nature Reviews Genetics*, vol. 19, no. 1, pp. 34–49, 2018.

[2] J.-C. Lachance, S. Rodrigue, and B. O. Palsson, "Minimal cells, maximal knowledge," *Elife*, vol. 8, p. e45379, 2019.

[3] P. Cacheiro, V. Muñoz-Fuentes, S. A. Murray, M. E. Dickinson, M. Bucan, L. M. Nutter, K. A. Peterson, H. Haselimashhadi, A. M. Flenniken, H. Morgan *et al.*, "Human and mouse essentiality screens as a resource for disease gene discovery," *Nature communications*, vol. 11, no. 1, p. 655, 2020.

[4] X. Ji, D. K. Rajpal, and J. M. Freudenberg, "The essentiality of drug targets: an analysis of current literature and genomic databases," *Drug Discovery Today*, vol. 24, no. 2, pp. 544–550, 2019.

[5] B. Bosch, M. A. DeJesus, N. C. Poulton, W. Zhang, C. A. Engelhart, A. Zaveri, S. Lavalette, N. Ruecker, C. Trujillo, J. B. Wallach *et al.*, "Genome-wide gene expression tuning reveals diverse vulnerabilities of m. tuberculosis," *Cell*, vol. 184, no. 17, pp. 4579–4592, 2021.

[6] W. Wang, J. Bao, S. Zheng, S. Huang, J. Aldahdooh, Y. Wang, J. Eriksson, Z. Tanoli, X. Zhang, M. Gaetani *et al.*, "A gene essentiality signature enables predicting the mechanism of action of drugs," 2022.

[7] L. Chang, P. Ruiz, T. Ito, and W. R. Sellers, "Targeting pan-essential genes in cancer: challenges and opportunities," *Cancer cell*, vol. 39, no. 4, pp. 466–479, 2021.

[8] K. Niu, Q. Fu, Z.-L. Mei, L.-R. Ge, A.-Q. Guan, Z.-Q. Liu, and Y.-G. Zheng, "High-level production of l-methionine by dynamic deregulation of metabolism with engineered nonauxotroph escherichia coli," *ACS Synthetic Biology*, vol. 12, no. 2, pp. 492–501, 2023.

[9] X. Li, W. Li, M. Zeng, R. Zheng, and M. Li, "Network-based methods for predicting essential genes or proteins: a survey," *Briefings in bioinformatics*, vol. 21, no. 2, pp. 566–583, 2020.

[10] O. Aromolaran, D. Aromolaran, I. Isewon, and J. Oyelade, "Machine learning approach to gene essentiality prediction: a review," *Briefings in bioinformatics*, vol. 22, no. 5, p. bbab128, 2021.

[11] L. J. Freischem, M. Barahona, and D. A. Oyarzún, "Prediction of gene essentiality using machine learning and genome-scale metabolic models," *IFAC-PapersOnLine*, vol. 55, no. 23, pp. 13–18, 2022.

[12] L. Wu, P. Cui, J. Pei, L. Zhao, and X. Guo, "Graph neural networks: foundation, frontiers and applications," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 4840–4841.

[13] K. Lazaros, D. E. Koumadorakis, P. Vlamos, and A. G. Vrahatis, "Graph neural network approaches for single-cell data: A recent overview," *arXiv preprint arXiv:2310.09561*, 2023.

[14] J. D. Orth, I. Thiele, and B. Ø. Palsson, "What is flux balance analysis?" *Nature biotechnology*, vol. 28, no. 3, pp. 245–248, 2010.

[15] R. Schuetz, L. Kuepfer, and U. Sauer, "Systematic evaluation of objective functions for predicting intracellular fluxes in escherichia coli," *Molecular systems biology*, vol. 3, no. 1, p. 119, 2007.

[16] C. E. García Sánchez and R. G. Torres Sáez, "Comparison and analysis of objective functions in flux balance analysis," *Biotechnology progress*, vol. 30, no. 5, pp. 985–991, 2014.

[17] B. Schnitzer, L. Österberg, and M. Cvijovic, "The choice of the objective function in flux balance analysis is crucial for predicting replicative lifespans in yeast," *Plos one*, vol. 17, no. 10, p. e0276112, 2022.

[18] S. Ravi and R. Gunawan, "δfba—predicting metabolic flux alterations using genome-scale metabolic models and differential transcriptomic data," *PLoS Computational Biology*, vol. 17, no. 11, p. e1009589, 2021.

[19] H. A. Herrmann, B. C. Dyson, L. Vass, G. N. Johnson, and J.-M. Schwartz, "Flux sampling is a powerful tool to study metabolism under changing environmental conditions," *NPJ systems biology and applications*, vol. 5, no. 1, p. 32, 2019.

[20] B. Strain, J. Morrissey, A. Antonakoudis, and C. Kontoravdi, "How reliable are chinese hamster ovary (cho) cell genome-scale metabolic models?" *Biotechnology and Bioengineering*, 2023.

[21] P. E. Gelbach, H. Cetin, and S. D. Finley, "Flux sampling in genome-scale metabolic modeling of microbial communities," *BMC bioinformatics*, vol. 25, no. 1, p. 45, 2024.

[22] H. C. Yeo, J. Hong, M. Lakshmanan, and D.-Y. Lee, "Enzyme capacity-based genome scale modelling of cho cells," *Metabolic engineering*, vol. 60, pp. 138–147, 2020.

[23] N. Yamano-Adachi, R. Arishima, S. Puriwat, and T. Omasa, "Establishment of fast-growing serum-free immortalised cells from chinese hamster lung tissues for biopharmaceutical production," *Scientific Reports*, vol. 10, no. 1, p. 17612, 2020.

[24] R. Hasibi, T. Michoel, and D. A. Oyarzún, "Integration of graph neural networks and genome-scale metabolic models for predicting gene essentiality," *npj Systems Biology and Applications*, vol. 10, no. 1, p. 24, 2024.

[25] A. T. Joo Schapke and M. Recamonde-Mendoza, "Epgat: Gene essentiality prediction with graph attention networks," *arXiv preprint arXiv:2007.09671*, 2020.

[26] S. Fallahi, H. J. Skaug, and G. Alendal, "A comparison of monte carlo sampling methods for metabolic network models," *Plos one*, vol. 15, no. 7, p. e0235393, 2020.

[27] W. Megchelenbrink, M. Huynen, and E. Marchiori, "optgpsampler: an improved tool for uniformly sampling the solution-space of genome-scale metabolic networks," *PloS one*, vol. 9, no. 2, p. e86587, 2014.

[28] M. Beguerisse-Díaz, G. Bosque, D. Oyarzún, J. Picó, and M. Barahona, "Flux-dependent graphs for metabolic networks," *NPJ systems biology and applications*, vol. 4, no. 1, p. 32, 2018.

[29] M. Ponce-de León, I. Apaolaza, A. Valencia, and F. J. Planes, "On the inconsistent treatment of gene-protein-reaction rules in context-specific metabolic models," *Bioinformatics*, vol. 36, no. 6, p. 1986, 2020.

[30] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[31] B. G. Galuzzi, L. Milazzo, and C. Damiani, "Best practices in flux sampling of constrained-based models," in *International Conference on Machine Learning, Optimization, and Data Science*. Springer, 2022, pp. 234–248.

[32] K. Xiong, K. J. la Cour Karottki, H. Hefzi, S. Li, L. M. Grav, S. Li, P. Spahn, J. S. Lee, I. Ventina, G. M. Lee *et al.*, "An optimized genome-wide, virus-free crispr screen for mammalian cells," *Cell reports methods*, vol. 1, no. 4, 2021.

[33] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[34] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 855–864.

A Appendix / supplemental material

A.1 Mass Flow Graph

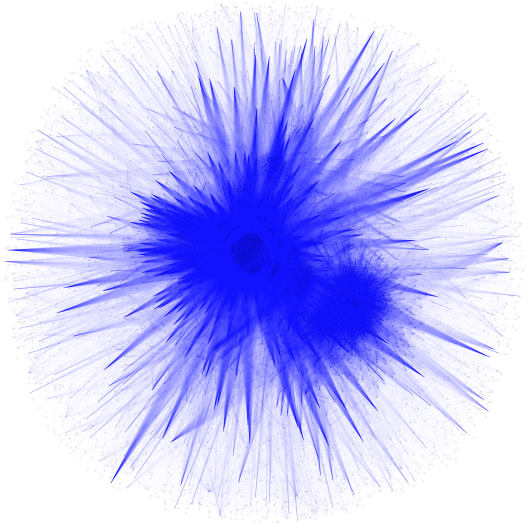


Figure 5: Mass flow graph (MFG) representation of the CHO cell metabolic reaction network, constructed using flux sampling data, containing 4,733 nodes (reactions) and 335,024 edges (interactions).

A.2 Hyperparameter Optimisation

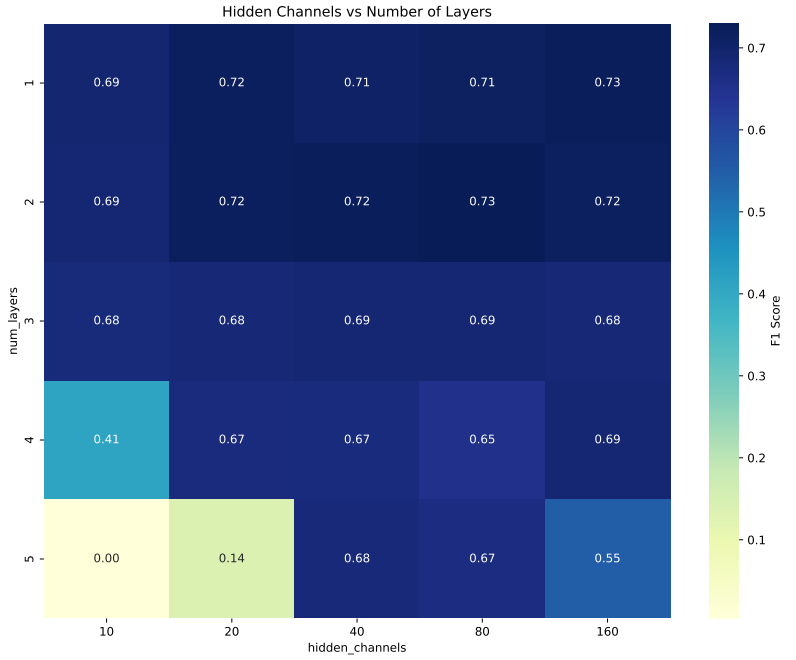
Optimal hyperparameter selection for FluxGAT was crucial to enhance its learning efficiency and predictive accuracy. This section focuses on the systematic approach we adopted to tune these hyperparameters. Our objective was to improve the model's generalisation capability and to ensure a balance between computational efficiency and performance. By exploring a range of hyperparameter values, we aimed to find the best combination that maximised the effectiveness of the GAT architecture on the task of biological essentiality prediction within the Chinese hamster ovary (CHO) cell mass flow graph (MFG).

We employed a grid search strategy to identify the optimal hyperparameters for FluxGAT. This method involved creating a grid of possible values for three key hyperparameters: the number of hidden channels, the dimension of the initial embedding layer, and the number of message-passing layers. Our objective was to maximise the F1 score to balance the model's precision and recall trade-off. By exhaustively searching through the combinations of these hyperparameters, we were able to assess the model's performance under various configurations.

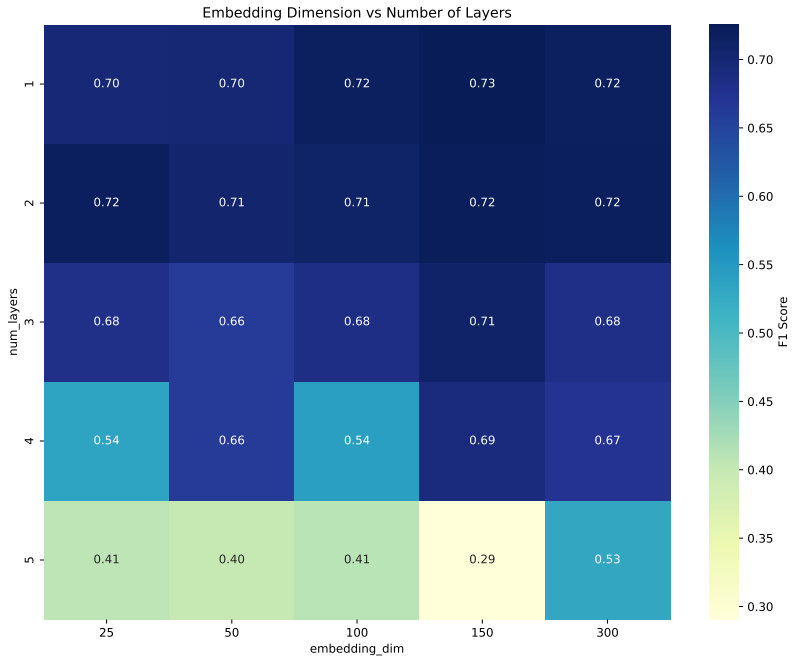
We defined a discrete search space composed of specific values for each hyperparameter. The hidden channels hyperparameter was varied across five possible values: 10, 20, 40, 80, and 160. For the embedding layer, we considered 25, 50, 100, 150, and 300-dimensional embeddings, and for the number of message-passing layers, the range was set from 1 to 5, inclusive. Mathematically, this approach created a Cartesian product of these sets, resulting in a total of 125 unique hyperparameter combinations to be evaluated. The numerical ranges were carefully chosen based on a combination of empirical testing and domain knowledge, ensuring relevance and potential effectiveness for FluxGAT.

Figure 6 shows two heatmaps: (6a) the number of hidden channels against the number of message-passing layers, and (6b) the dimension of the embedding layer against the number of message-passing layers. Each heatmap represents the average F1 scores corresponding to various combinations of the hyperparameters.

Out of the 125 unique hyperparameter combinations, the optimal was determined to be 150-dimensional embeddings, 150 hidden channels, and two message-passing layers. This configuration balances the model's complexity and its capacity for feature representation. The choice of 150-dimensional embeddings and hidden channels provides a robust yet efficient framework for feature



(a)



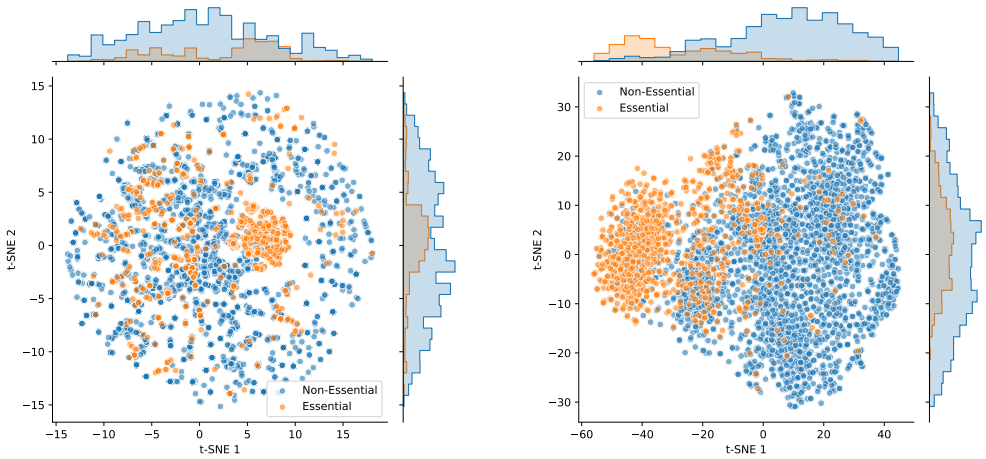
(b)

Figure 6: Heatmaps showing the F1 score for varying hyperparameters in FluxGAT's architecture.

extraction and transformation, suggesting that this dimensional space is sufficiently large to capture the relevant features of the data without overfitting or excessive computational demand. Furthermore, the selection of 2 message-passing layers indicates an effective depth for capturing the graph structure and node interactions, optimising the balance between learning complex patterns and mitigating oversmoothing.

A.3 Node Embedding Space

For an exploration of the feature transformations enabled by FluxGAT, Figure 7 illustrates the evolution from initial input features to refined embeddings, highlighting the model's effectiveness in distinguishing between essential and non-essential nodes. Future work will focus on understanding the underlying mechanisms and features that drive this transformation, aiming to identify the specific attributes and interactions FluxGAT leverages to classify essentiality.



(a) (b)

Figure 7: t-SNE visualisations contrasting the structure of input features and learned node embeddings. (a) t-SNE reduction of the initial input features for each reaction, colour-coded as *essential* and *non-essential*, with marginal histograms depicting the distribution of t-SNE components for each class. (b) t-SNE reduced high-dimensional embeddings generated by FluxGAT, highlighting the model's ability to refine and separate nodes based on their essentiality through learned representations. Both plots used a perplexity of 30.

A.4 Computational Resources

A.5 Hardware Specifications

Simulations were conducted using a high-performance computing (HPC) cluster, which is equipped with the following node types:

- Lenovo nx360 m5 compute nodes, each featuring two Intel E5-2680 v4 (Broadwell) CPUs with 14 cores at 2.4 GHz, and 128 GiB of RAM.
- High-memory nodes, each with 512 GiB of RAM.
- GPU-equipped nodes, each housing two graphics processing units.

For the experiments conducted, every node type was strategically utilised, based on the specific requirements of the job and the experiment phase in question. High-memory nodes were specifically employed for flux sampling of the iCHO2291 GSMM, whereas GPU nodes facilitated both the construction of MFGs and the training/evaluation of FluxGAT along with other GNN models. All remaining tasks were efficiently executed on the standard compute nodes.

A.6 Software and Libraries

- Python 3.9
- COBRApy 0.29.0
- PyTorch 2.1.1
- PyTorch Geometric 2.4.0

- NetworkX 2.8.8
- node2vec 0.4.6

A.7 Code Availability

To further support the reproducibility and transparency of our work, an anonymous version of the complete source code for FluxGAT, including Python code and detailed instructions necessary for replication, is available at the following URL: `https://github.com/kierensharma/FluxGAT`. This version of the code has been modified to allow execution on personal computers, although users should note that this may significantly increase processing time compared to running the code on an HPC cluster. Additionally, should the reviewers require it, we are prepared to provide any specific parts of the code relevant to their inquiries to facilitate a thorough and efficient review process.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state that our custom FluxGAT framework significantly enhances the sensitivity of gene essentiality prediction by addressing limitations of flux balance analysis (FBA). This is supported by our results and comparison in Section 4.4 and Figure 4, which demonstrate significant improvements over FBA.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: In Section 5, we acknowledge that our method currently relies on flux sampling data from a single cell line due to the limited availability of genome-wide growth screens, which may limit its generalisability to other organisms. Additionally, we discuss the computational intensity of flux sampling and how it may impact scalability. We also note that our model's performance could be affected by the quality of the metabolic network reconstruction used.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: Our paper focuses on empirical results and does not introduce new theoretical results that require formal proofs. Therefore, this question does not apply to our work.

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: In Section 3, we provide detailed descriptions of our methodology, including all relevant formulas and algorithms relating to; flux sampling procedures, graph construction methods, and data preprocessing steps. Additionally, we specify the model architecture of FluxGAT, along with all hyperparameters, training procedures, and evaluation metrics, in Section 4 and the supplementary materials.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Appendix A.7 details an anonymous version of the complete source code for FluxGAT, available at `https://github.com/kierensharma/FluxGAT`. This repository includes all scripts and resources necessary to generate the flux sampling dataset, as well as to train and test the model on a personal computer. Comprehensive instructions are provided in the README file, guiding users through each step of the reproduction process.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: In Sections 4.2 and 4.3, we provide comprehensive information on the data splits used for training and k-fold cross-validation. We detail the hyperparameters of the FluxGAT model, along with how they were selected through grid search and validation on a hold-out set (Appendix A.2). We specify the type of optimiser used (AdamW) and any regularisation techniques applied in Section 4.2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.

- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: In Table 1, we provide a detailed overview of the mean and standard deviation (1-sigma error bars) for test accuracy, precision, recall, and F1 score, aggregated over 25 evaluations (comprising five repetitions of 5-fold cross-validation) for each binary classifier. The error bars capture variability due to different train/test splits from the cross-validation and random initialisation in each repetition.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In Appendix A.5, we detail the compute resources used for the experiments described in Section 4, as well as the software versions used in Appendix A.6.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

Justification: We have thoroughly reviewed the NeurIPS Ethical Guidelines and ensured that our work adheres to all ethical standards.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: In Section 1, we highlight the positive impact of our model in improving gene essentiality prediction across diverse organisms, which can advance drug discovery and metabolic engineering. In Section 5, we address potential negative impacts, such as the environmental concerns associated with high computational resource usage and the risks of incorrect predictions.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper does not involve the release of data or models that have a high risk for misuse. Our model requires significant computational resources and specialised scientific expertise to operate and act upon the results, which reduces the potential for misuse.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have properly credited the creators of the iCHO2291 genome-scale metabolic model (GSMM) of Chinese hamster ovary (CHO) cells developed by Yeo et al. [22]. Since there is no publicly available license information for this model, we have used it solely for non-commercial academic research purposes. Additionally, all software packages used are listed in Appendix A.6 with their respective versions, and we have adhered to their terms of use.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We introduce a new asset in the form of the FluxGAT codebase, which is well documented and provided alongside the asset. The code repository includes detailed documentation, installation instructions, and usage examples to facilitate understanding and reproduction of our work. We have provided this information in an anonymous repository (as mentioned in Appendix A.7) to comply with the review process.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: Our research does not involve crowdsourcing experiments or research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: Our research does not involve research with human subjects, so there were no potential risks to participants, and no Institutional Review Board (IRB) approval was required.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.