Graph Neural Networks in Clinical Cancer Feature Prediction

NELSON ALEXANDRE GEADA TRINDADE

Graphs are used to represent a wide range of information, from social interactions to road map structures, and applied in various fields such as biomedical research, from molecular fingerprint up to drug discovery and response. Advances in molecular profiling and biomedical technologies led to a significant increase in the ability to collect and study omic data from cancer patients and their tumors, with proteomics being a vital field for understanding their interactions, identifying cancer types, or even studying drug response. This thesis aims to develop Graph Neural Network models that can effectively handle the high-dimensionality of recent biomedical datasets to integrate molecular data and biological networks. Furthermore, this model aims to predict clinically relevant features of cancer, such as Cancer Type and subtype, and response to different treatments. To this end, a novel proteomic dataset that spans over 40 cancer types across $\sim 1,000$ cancer cell lines will be harnessed. Crucially, the performance of this dataset together with Graph Neural Network models that integrate protein-protein interaction networks remains largely unexplored. The performance of the developed Graph Neural Network model will be systematically evaluated by comparing it against state-of-the-art approaches and classical machine learning methods. These new developments in Graph Neural Network models, along with the use of big data, will be key in building more accurate predictive models with direct application in the understanding and development of novel clinical approaches for cancer.

CCS Concepts: • **Information systems** → **Graph-based databases**; Natural language processing; Computer vision; • **Computing methodologies** → *Machine learning*.

Additional Key Words and Phrases: Graph Neural Network, Machine learning, Cancer Classification, Proteomic Data, Biological Networks.

.

1 INTRODUCTION

1.1 Motivation

Graphs are indeed prevalent in various aspects of life, from molecular structures to social interactions, and they offer versatile representation. Advances in technologies have enabled the collection of extensive omics data from cancer patients, particularly in proteomics, which has broad applications in fields like drug discovery.

GNNs have shown significant promise in biomedical research, potentially aiding in the identification of genetic variants associated with complex traits.

1.2 Objectives

GNN models analysis will be the main focus of this thesis, but three objectives can be fulfilled:

- Develop GNN models to integrate molecular data (i.e. proteomics) and biological networks (i.e. protein-protein interactions);
- Use these models (e.g. classifiers) to predict clinically relevant features of cancer (e.g. Cancer Type and subtype and drug responses);
- Systematically benchmark novel GNN designs against current state-of-the-art approaches and classical ML methods.

Evaluating the performance of trained GNN models using established classification metrics is crucial. These results will be compared to state-of-the-art deep learning models and traditional machine learning techniques to assess how GNNs, with their incorporation of prior network knowledge, enhance predictive power and transparency.

1.3 Outline

The thesis comprises six essential sections. "Background" introduces various learning techniques, emphasizing GNNs. "Related Work" reviews current research, with a focus on biomedical studies. "Data Exploration" analyzes the data used. "Baseline Models" presents fundamental benchmarks. "Graph Neural Network Models" details the core methodology. "Final Considerations" summarizes key findings and the achievement of main objectives from Section 1.2.

2 BACKGROUND

This section will provide an overview of the theory behind state-of-the-art machine learning algorithms, specifically focusing on classification algorithms that can classify records from different datasets. It will also describe the differences between supervised and unsupervised learning, and explain why classification is the chosen method for this study.

Additionally, a section on evaluation metrics will present the most widely used metrics in data science and biomedical fields, which will be useful for evaluating the performance of the classifiers tested and determining their accuracy.

2.1 Predictors and Learning Methods

Classifiers are machine learning algorithms used for data categorization, which can be binary or multi-category. They are a key component in classification models.

There are two main types of learning techniques: supervised and unsupervised. Unsupervised learning doesn't require labeled data and identifies data through pattern recognition and anomalies. It's commonly used in clustering and dimensionality reduction. Clustering groups data based on similarities, while dimensionality reduction minimizes features while preserving vital information.

Supervised learning uses labeled datasets to predict and adjust to target labels, making it more accurate in most cases. In practice, datasets are divided into training and test sets. The training set teaches the model to recognize features and labels, while the test set evaluates the model's performance.

In this thesis, supervised learning will play a significant role as it leverages labeled information to understand and classify data, including future records.

Author's address: Nelson Alexandre Geada Trindade, nelson.trindade@tecnico.ulisboa.pt.

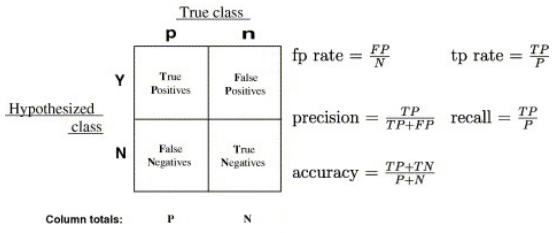

Fig. 1. Metrics Concepts, and common Performance Metrics. From [10]

2.2 Evaluation Metrics

To compare different classification models, simple but effective evaluation metrics can be used, all defined in Figure 1, and some more approaches can be used like:

- Accuracy: Provides a measure of classifier confidence in predictions.
- F1-Score [6, 26]: A popular metric in machine learning, it combines precision and recall, with F1-Score being the most common variant. It ranges from 0 to 1, with 1 indicating high precision and recall.

The utilization of these metrics will facilitate the comparison of various classifiers and will enable a determination of the suitability of GNN as a method within the specific context under examination for the task in question.

2.3 Linear Classifiers

Linear classifiers seek to establish a linear boundary, often a line or hyperplane, for the purpose of distinguishing between different labels in a dataset. This hyperplane is typically represented by the sum of weighted features. The learning process involves adjusting these weights and the bias to align with the dataset. Linear classifiers are particularly suited for situations where the relationship between features and class labels is approximately linear or when the data can be separated by a straight line or hyperplane.

Linear classifiers are highly interpretable. The weights of each feature provide insight into their importance for prediction, making them useful for fields like healthcare and finance.

2.3.1 Logistic Function. Logistic Function is a sigmoid function, σ, that maps numerical input to a value between 0 and 1, respecting Equation 1. In the equation's graph, it shows that as the independent variable becomes more negative, the dependent variable approaches 0. Conversely, as the independent variable becomes more positive, the dependent variable approaches 1.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{1}$$

2.3.2 Perceptron. The perceptron, a fundamental component of neural networks, resembles a human brain neuron. It operates with four key elements: input values, weights, a weighted sum, and an activation function.

Inputs are multiplied by corresponding weights to represent their importance. The weighted sum combines these values, which then undergo an activation function. The unit step function is a basic example, producing either 0 or 1 based on a defined threshold. The hyperbolic tangent ranges from -1 to 1.

For an n-dimensional input vector (x), random weights (w), and a bias (b), the perceptron's output (y) is computed using an activation function (h). During training, weights can be updated using the true label ($\hat{y}$).

The logistic function is an alternative activation function, outputting values between 0 and 1. The bias acts as a threshold, affecting the decision boundary for output. Adjusting it can optimize classifier performance.

2.3.3 Linear Support Vector Machines. LSVM is a well-known machine learning algorithm primarily used for classification tasks. It achieves high accuracy with minimal computational resources when dealing with small and linearly separable datasets. The goal of LSVM is to determine the optimal hyperplane for effectively distinguishing binary labels. This hyperplane can be as simple as a line in 2D or more complex in higher dimensions. Support vectors, which are points near the decision boundary, play a crucial role in determining the hyperplane's position and orientation. Altering or removing a support vector can significantly impact the hyperplane's fit to the data. The SVM's objective is to maximize the margin, which represents the distance between the support vectors and the defined hyperplane. In cases where the data is not linearly separable, more advanced techniques, such as the kernel trick, are required to transform the data into a higher-dimensional space for successful separation, which LSVM cannot perform.

2.4 Non-linear Classifiers

Non-linear classifiers are a type of machine learning technique that can learn complex relationships between input data and target variables, allowing them to model patterns in the data. They tend to be more accurate but are more challenging to train, evaluate, and interpret than linear methods.

A glance will be taken at some popular non-linear classifiers, like decision trees, random forests, and focus on neural networks and GNN algorithms.

2.4.1 Random Forest. Random Forest (RF) was developed to reduce the main problems of decision trees [5]. This algorithm joins a collection of decision trees, each tree formed from a training set with a replacement, process called bagging. This randomness adds more diversity to the dataset and reduces the correlation among the trees. The outcome will vary depending on the task at hand, selecting the majority vote of all decision trees for the predicted class. This algorithm has three main parameters that can be adjusted: the node size, the number of trees generated, and the number of features sampled.

In comparison to decision trees, random forests are a time and resource-consuming algorithm, since they have to compute large numbers of decision trees. However, they reduce the risk of overfitting, since the prediction comes from most of a robust number of uncorrelated decision trees. It can also provide flexibility since it maintains accuracy even with missing values.

2.4.2 K-Nearest Neighbours. The KNN algorithm [7, 22] is a supervised learning classifier that operates based on the concept of

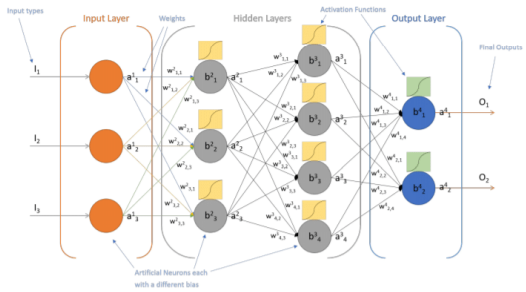

Fig. 2. Visual representation of an Artificial Neural Network. From [29]

proximity. It predicts the group to which a data point belongs by identifying its nearest neighbors in the feature space. The parameter 'K' determines how many neighbors are considered for classification, with K=1 assigning the same class as the nearest neighbor.

KNN's main goal is to find the closest neighbors to a query point and assign a class label based on a majority vote among these neighboring points. The label that occurs most frequently among nearby data points is assigned to the target data point.

To measure proximity, the algorithm employs various distance metrics, including Euclidean Distance, Manhattan Distance, and Minkowski Distance. KNN is known for its flexibility and ease of use, as it retains all training data in memory and requires only a few hyperparameters, making it adaptable to new data. However, it has limitations, including susceptibility to overfitting, scalability issues due to memory and time constraints, and challenges with high-dimensional data, known as the "curse of dimensionality", leading to classification problems.

2.4.3 Neural Network. NN [19] is the heart of deep learning algorithms [18]. They are composed of input and output layers, and between them, there can be $n \in \mathbb{R}$ hidden layers. These layers are composed of nodes, also called artificial neurons, that when activated send data to the next layer of the network. In order to communicate with other nodes, each connection has an associated weight and bias, and the data is sent when the specified threshold is reached.

The input is multiplied by the respective weights and then summed. If the output of this equation is bigger or equal to the threshold value of each node, the message is passed through the next layer, a process called a feedforward network. The value is then passed through an activation function (while the logistic function has been discussed, other examples such as ReLU [1] can also be applied) since it will reduce the impact of any change of a single variable, and even in the output of the neural network.

To train a Feedforward Neural Network (FNN), the backpropagation algorithm is commonly used [18]. This involves forwarding input through the FNN to calculate the output and then adjusting the connection weights between nodes based on the error between desired and actual output. This method is widely used in training FNN and forms the basis of many deep learning models.

To assess model accuracy, cost functions like cross-entropy for classification [15] or mean squared error for regression [4] are commonly used. The goal is to minimize these functions to ensure a good fit for observations. Gradient descent [23] helps the model find the right direction to reduce errors.

There are various types of neural networks. Convolutional Neural Networks (CNNs) [3] are popular for computer vision tasks [20], like face recognition [17]. Recurrent Neural Networks (RNNs) are used with time-series data, like stock market predictions. There are more types beyond these, and 'deep neural networks' typically have many layers, often more than two.

2.5 Graph Neural Network

Graphs are simply the aggregation of entities (nodes) and their relations (edges). They can represent a variety of concepts, from the basic molecular structure or social interactions to more complex examples, such as images or text.

Graphs can be even more complex and can aggregate more information. Some of the more complex graphs are Multigraphs, Hypergraphs, and Hierarchical Graphs.

Integrating graphs in a NN requires making the data from graphs compatible with this deep learning model. Graphs have up to four types of information that will be useful to gather information, such as nodes, edges, global context, and connectivity. Graph-structured data enables three types of predictions based on the provided information:

- **Graph-level tasks:** Involves predicting a single property for the entire graph. Comparable to image classification, it is akin to identifying a label for the entire image.
- **Node-level tasks:** Focuses on predicting the role of each individual node in the graph. In the context of image processing, this is akin to classifying each pixel in an image, where nodes represent objects in the image.
- **Edge-level tasks:** Encompass predicting the properties or presence of edges in a graph. In the context of images, this equates to forecasting which nodes (objects) are connected by edges or determining the value associated with those edges.

In these tasks, managing information can be challenging. Nodes, edges, and global context can be turned into a node feature matrix by assigning each node an index (i) and storing its features. However, representing the graph's connectivity with an adjacency matrix is not recommended due to its sparsity when dealing with a large number of nodes and edges. Instead, adjacency lists are a more efficient way to represent this connectivity.

GNN models optimize the transformation of all graph attributes, like nodes, edges, and global context, while preserving graph symmetries. GNN takes a graph as input with information in its nodes, edges, and global context, and progressively transforms these embeddings without changing the graph's connectivity.

A GNN applies a separate MLP or differentiable models to each part of the graph, known as a GNN layer. This process involves applying the MLP to node vectors, edge vectors, and the global context vector. These GNN layers can be stacked, similar to neural network layers.

Since a GNN doesn't alter the input graph's connectivity, the output graph can be described using the same adjacency list and feature vectors as the input graph. However, the embeddings are

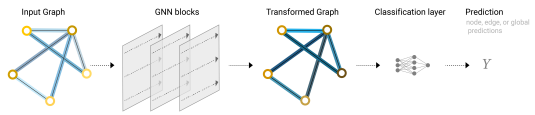

Fig. 3. End-to-End prediction task with Graph Neural Network. From [25]

updated due to the GNN's transformations. You can find a visual representation in Figure 3 [24].

2.5.1 Message Passing and Available Architectures. Message Passing enhances node embeddings in GNNs by considering graph connectivity. It involves gathering neighboring node embeddings, aggregating them using a function, and applying an update function. This process applies to both nodes and edges, and stacking multiple layers allows nodes to capture information from the entire graph.

In graph classification, four main architectures are used: graph convolution networks, graph attention networks, graph sample and aggregate networks, and graph isomorphism networks. They all rely on message passing, with the last three being adaptations of graph convolution networks. For node classification, the first two are also commonly employed.

Graph Convolution Networks have emerged as a powerful and widely discussed type of neural network, especially in the context of artificial intelligence. They are closely related to CNN and are particularly noteworthy for their ability to transform image representations into graph structures. In this graph, each node corresponds to a pixel, and its features are determined by the RGB values associated with that pixel.

In the context of GCN, the concept of permutation invariance holds substantial significance. Permutation invariance denotes the network's ability to process data, irrespective of the order of node presentation within the graph. This is achieved through the use of aggregation functions, typically the sum or mean, treating all neighbors uniformly. Consequently, the output of a GCN layer remains consistent, regardless of the nodes' arrangement within the graph.

One of the main problems of GCN is that similar nodes can easily forget about node-specific attributes. Graph Attention Networks focus on solving this flaw by adding an attention mechanism to the graph, to give different importance/weights on the nodes. Attention describes a weighted average of multiple elements with weights dynamically computed based on an input query and elements key. Each node creates a message using a linear/weight matrix. It uses the message from the node itself as a query, and the messages to average as both keys and values (also with a message to itself).

3 RELATED WORK

In High Energy Physics [28], GNN overcame challenges in understanding relationships between nodes in particle physics data by representing it in a graph format with better inductive bias and reduced parameters. In blockchain technology [32], GCN were able to detect vulnerabilities in smart contracts by converting code into contract graphs and normalizing them with a degree-free method, achieving better results than conventional neural networks. In social recommendation systems [9], GNN outperformed state-of-the-art approaches by utilizing social networks modeled as graphs, addressing challenges such as distinguishing good or bad opinions. In web mapping services [8], GNN improved estimated arrival time by defining the road network as a graph, benefiting both users and dependent services.

3.1 Graph Neural Networks In Biology and Biomedicine

In cheminformatics [16], molecular fingerprinting is crucial for virtual screening and mapping chemical space using deep learning algorithms. This approach transforms small molecules into molecular graph convolutions, enabling deep learning systems to conduct virtual screening efficiently.

Histological images [2] provide micro-anatomy information and aid in diagnosing diseases. Graph Convolutional Networks (GCNs) introduce global contextual information, facilitating a deeper understanding of tissue microstructure. GCNs can be applied to cell-tissue graphs, detecting disease progression and classifying conditions like Alzheimer's disease from magnetic resonance images.

Electronic Health Records (EHR) often use ICD codes [2] to build a comprehensive medical knowledge network, including various data types. GNNs with attention mechanisms offer personalized predictions and can capture relationships within EHR data. Recent methods utilize temporal dependencies for time-dependent prediction tasks, combining different models to represent patient status and recommend medication combinations.

3.2 Representation And Integration Of Molecular Networks

GML models [13, 31], which analyze and predict data in graph format while considering the relationships between elements in the graph, can model biomolecule structures, functional relationships, and integrate multi-omic datasets. Gaudelet et al. [11] provided insights into the use of GML in multi-omic data, where proteins and biomolecules are represented as graphs to capture spatial and structural relationships. They utilized a knowledge graph to generate hypotheses for SARS-CoV-2 treatment, employing the RotatE model. Different studies in this field have demonstrated that various models and architectures can yield different results.

Single-cell transcriptomics data can be transformed into a graph structure, with nodes representing cells or genes and relationships captured using appropriate distance measures. This graph serves various purposes [14], including predicting cell types and classifying cancer types. Additionally, link prediction tasks can be performed on the graph by leveraging positional encoding and training in an unsupervised manner based on the graph's topology.

3.2.1 Multi-omics Graph Convolutional Networks. Additionally, GNN are already being studied to classify patient diseases and biomarker identification. MOGONET [30] is a method presented for biomedical classification that can take advantage of connected omics data, for example, mRNA expression data, DNA methylation data, and microRNA expression data, to predict a label. To do this, it is created a weighted sample similarity network, that uses cosine similarity to compare each data expression. The GCN is then trained on the omics features, using the corresponding similarity network, to produce initial predictions of class labels. These labels are then joined in a

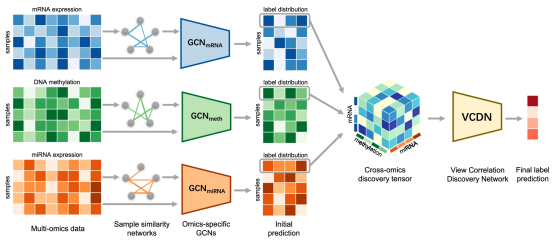

Fig. 4. Multi-omics Graph Convolutional Networks Architecture. From [30]

cross-omic discovery tensor, which is analyzed by a View Correlation Discovery Network to produce the final label shown in Figure 4.

The effectiveness of this method was tested on three main datasets: LGG (Lower Grade Glioma, a type of brain cancer), ROSMAP (Relative Oxygenation Status Maps, gene expression for Alzheimer's disease), and BRCA (Breast Invasive Carcinoma, gene expression for breast cancer). The results showed that the GCN outperformed state-of-the-art classification methods such as KNN, SVM, and even NN in the majority of cases. This demonstrates the superiority of graph learning models over traditional deep learning algorithms.

MOGONET was also proven to be extended to accommodate different numbers of omics data types. These tests show that MOGONET can clearly classify datasets with the three refereed expression data, but the metrics decrease for two expression data classifications and decrease more when using only one expression data classification. Nevertheless, it still produced better results than state-of-the-art classifiers. These classification tests proved that MOGONET could identify biomarkers related to Alzheimer's disease and biomarkers related to breast cancer.

3.2.2 Multi-Omics Late Integration. MOLI [27] introduces a unique perspective in the context of feature optimization. MOLI is a deep neural network that predicts the drug response for a given sample, represented by its multi-omics profile, and for a given drug. It is a model that takes somatic mutation, copy number aberration, and gene expression data as input, and integrates them for drug response prediction, using different sub-networks.

It starts with multiple feedforward encoding sub-networks, one for each input omics data type. Each encoding sub-network receives its corresponding omics data and encodes it into a learned feature space. The learned features are integrated (utilizing a late integration approach) and concatenate the learned features of the different single-omics data types to obtain one multi-omics representation. In order to optimize this representation, a combined cost function consisting of a triplet loss and a binary cross-entropy loss is used. The learned features will be used by a classifier that predicts the drug response. Therefore, the last sub-network of MOLI is a classification layer with the Sigmoid activation function, using dropout and weight decay for regularization.

The MOLI model was compared with early integration methods, which outperformed six out of seven when compared to early integration via deep neural networks and totally with modals with early integration via non-negative matrix factorization. When comparing single-omics data vs multi-omics data, deep neural networks trained on multi-omics data achieve better performance than those trained on single-omics data.

4 DATA EXPLORATION

4.1 Datasets Available and Challenges

Access to a proteomics dataset [12] is available, containing 6,671 unique proteins within 949 distinct cell line records. This dataset plays a central role in all study phases, from establishing a baseline to integrating node information into the GNN models. The dataset includes a unique identifier for each cell line, providing a strong foundation for investigating cancer-related questions. An additional table was used to classify three label categories: Tissue Type, Cancer Type, and Cancer Subtype, allowing for a comprehensive exploration of these classifications.

One challenge arises from the proteomics dataset, which has more features than records, a common issue known as the curse of dimensionality. This means there are more proteins (features) in the dataset than registered cell line records. This can lead to computational inefficiencies and suboptimal model performance. Feature Selection is crucial in addressing this challenge. By carefully choosing a relevant subset of features, the training and testing processes can be more efficient and improve result accuracy.

The absence of a Cell-Line Correlation Network is a significant concern, as there are limited datasets that align with the available data. To address this, we've created a correlation network among the records, forming a cell-line record graph. This internally generated network has advantages for the research framework, particularly in facilitating node-level classification. However, incorporating a well-established, previously examined network could further enhance result precision.

During label imputation, some records may be excluded as a precaution due to the low occurrence of specific labels. This approach ensures a rigorous and thorough labeling process, contributing to the precision and credibility of subsequent analysis.

4.2 Proteomics Dataset Preprocessing

For accurate comparisons and baseline establishment in evaluating the developed GNN models, extensive efforts were dedicated to processing the proteomics dataset, beginning with a thorough quality and reliability assessment.

In the initial stage of preprocessing the proteomics dataset, an endeavor was undertaken to retain solely fundamental data. This entailed the exclusion of certain proteins manifesting an excessive prevalence of missing values, following a subsequent selection of pivotal proteins in the dataset. The decision to eliminate specific proteins was motivated by the circumstance of having more proteins than cell-line records.

4.2.1 Missing Values. Amongst a comprehensive dataset encompassing 6,330,779 data points, a notable subset of 3,033,456 entries was identified as missing (accounting for 48% of the dataset), necessitating meticulous treatment. Additionally, a subset of 2,332 proteins exhibited a prevalence of 75% or greater missing values. Hence, it was decided to exclude these proteins to maintain a dataset with a higher proportion of reliable information, as their inclusion would offer limited or unreliable data for analysis.

4.2.2 Standardization. When analyzing the range of values for different proteins, it becomes evident that each protein exhibits a diverse and varied range of values. This heterogeneity highlights the need for standardization techniques [21] to establish a more consistent and uniform practice.

Upon applying the standardization process, the standardized data exhibits a noteworthy characteristic: the alignment of median values along the y-axis at zero. This observation highlights a key benefit of the standardization approach, which involves replacing missing values with a constant, such as zero. This observation underscores a key advantage of standardization, which entails substituting missing values with a constant, typically zero. Notably, this replacement has minimal to no impact on the overall data range, but it plays a crucial role in handling missing values. By embracing this strategy, the dataset's integrity is upheld, thereby enabling a more precise analysis and elucidation of the protein data.

4.2.3 Feature Selection. Addressing the challenge of having a surplus of proteins compared to cell-line records, the study utilized the Variance Threshold method. This method helps identify which proteins have significant variance, indicating their potential importance in distinguishing or characterizing different data points. To establish the appropriate threshold, a Gaussian Mixture model was employed. This visualization provided insights into protein data distribution, aiding in the selection of relevant proteins. The choice of threshold is pivotal, as it determines which features are retained.

The use of GMM was crucial in predicting an optimal threshold for standard deviation in this process. This analytical approach yielded significant results when selecting the second interception of the Figure. It enabled the identification of 1, 528 unique proteins, which accounted for 23% of the total proteins present in the original dataset. These proteins correspond to proteins that have a standard deviation greater or equal to 0.947.

The approach employed, which combines GMM with identifying relevant standard deviation thresholds, highlights their inherent ability to tackle the complex challenges involved in meticulous data analysis.

4.3 Correlation Network

Creating a network from complex cell-line records was a challenging task, given their intricate interdependencies.

The core of building the correlation network involved evaluating various correlation matrices, using metrics like Cosine, Spearman, Pearson, and Kendall as the foundation. The focus was on positive connections while excluding self-loops to simplify the network.

Identifying pivotal connections required a thorough investigation. This approach was influenced by MOGONET's work, which developed similarity networks using cosine similarity and tested different average node counts 2, 5, 10. In this thesis, the approach involved testing different cosine thresholds from 1 to 0 in 0.1 intervals, with variations in the number of nodes, edges, and average nodes, including graphs with average nodes of 50 and 100.

The process involved applying various thresholds and calculating average node edges in the GCC, enhancing the understanding of network behavior. The Cosine similarity metric played a pivotal role, and rigorous testing confirmed its effectiveness.

In certain scenarios, efforts were made to reduce the number of edges per node while retaining pivotal proteins (hubs) by applying equation to individual nodes, retaining only the edges with the highest confidence during the procedure.

4.4 Distribution of Labels

This study's foundational approach involves incorporating a supplementary classification framework, emphasizing the need for precise labeling techniques. This methodology deepens the understanding of cancer data, revealing complex cellular behavior and connections to malignancy.

There are three label categories: Tissue Type with 28 labels, Cancer Type with 45 labels, and Cancer Subtype, resulting in 135 possible combinations when combined with Cancer Type (not shown in the graph). To create an effective classifier, a minimum of 10 records per label is required for Cancer Type and Subtype. This ensures a strong classification system with meaningful results.

The Tissue Type label category was extensively examined with different thresholds to determine the optimal number of records for effective classification. Systematic evaluation of these thresholds determined the point where dataset size and classification performance are balanced for Tissue Type accuracy. Studying different thresholds for Tissue Type enhances the approach, leading to a more nuanced understanding of tissue-related classifications and contributing to the overall classification framework's precision and robustness.

5 BASELINE MODELS

To enable a comprehensive comparison of state-of-the-art classification models, a diverse set of models, including Logistic Regression, Support Vector Machine, Random Forest, K-Nearest Neighbors, and Multi-Layer Perceptron, serves as the baseline.

Comparability of models relies on standardized evaluation using four key metrics: Precision, Recall, Accuracy, and F1-Score, providing insights into predictive capabilities and overall performance.

Initial model generation revealed significant overfitting in the training set across all metrics, prompting the need for a refined approach to model development and evaluation. Some models exhibited a notable disparity between expected and actual F1-Score performance in the test set, raising questions about their generalization capabilities.

Addressing overfitting requires diverse strategies due to data constraints. Exploring cross-validation techniques and balance techniques like SMOTE and undersampling can help mitigate overfitting risks. A grid search was performed to find the best hyperparameter values, tailoring parameters to maximize the F1-Score for each model to discover the best-performing configuration.

5.1 Validation Techniques

Examining the dataset's label distribution, it's evident that there is a significant imbalance, particularly with the Tissue type labels. When dealing with a small dataset, a conventional 80%-20% split for training and testing may lead to insufficient representation of certain labels during testing due to their scarcity.

To address this issue, two common techniques were explored: K-Fold and Stratified K-Fold cross-validation. K-Fold involves dividing the dataset into "K" subsets, with one serving as a validation set and the rest for training in each iteration. Stratified K-Fold, on the other hand, not only divides the data into "K" folds but also maintains the original label distribution within each fold, which is crucial for imbalanced datasets.

After careful examination, the decision was made to exclusively use the Stratified K-Fold technique due to its ability to ensure fair label distribution. This leads to more representative training and validation sets, improving model performance and generalization.

Determining the optimal number of folds is essential. An initial investigation considered 3, 5, and 7 folds to strike a balance between reliable evaluation and computational efficiency. Ultimately, five-fold cross-validation was chosen, as it offered competitive performance while being computationally efficient.

Balancing the dataset is vital for the testing phase. Techniques such as Oversampling (e.g., SMOTE) and Undersampling can be employed during training. Undersampling, specifically Random Undersampling, is used for labels with a prevalence exceeding 60% of the maximum occurrence count among all labels.

Random Undersampling aims to reduce the instances associated with dominant labels to a predefined threshold, which is set at 40% of the highest occurrence count across all labels. Subsequently, SMOTE is used to generate synthetic records for labeled data points.

Undersampling before SMOTE is essential to control the proliferation of synthetic records. It's important to note that variations in F1-Scores were observed across different thresholds, demonstrating the effectiveness of these methodologies in mitigating overfitting.

In conclusion, these strategies contribute to achieving better model generalization and mitigating overfitting, highlighting the importance of a well-balanced dataset and the combination of Undersampling and SMOTE for managing imbalanced data.

5.2 Optimal Model Selection

In the endeavor to identify the most optimal models, a systematic grid search was conducted to ascertain the optimal model for each state-of-the-art classifier introduced in the initial sections of this chapter. This procedure involved the careful specification of predefined model parameters. Subsequently, the grid search protocol was employed to subject the models to an array of metrics, encompassing both the test and train datasets. Importantly, this evaluation was fortified by the rigorous implementation of cross-validation techniques, further augmented by judicious oversampling strategies, both mentioned above. This meticulous approach aids in enhancing the reliability and robustness of the model selection process.

5.3 Baseline Results

Upon the completion of comprehensive preprocessing procedures applied to the proteomics dataset, alongside an earnest investigation of foundational models to mitigate overfitting risks intrinsic to modest datasets, a meticulous curation of optimal models was orchestrated via an exhaustive grid search. Subsequently, these discerned optimal models were visually represented using bar charts, facilitating a judicious comparative analysis.

In the pursuit of a nuanced understanding of model performance, visualizations of confusion matrices were meticulously generated. These matrices offered a lucid representation of instances where the model's predictions deviated from verifiable data. Notably, this exploration extended further by examining the potential interplay between label frequency and prediction disparities. To this end, supplementary bar charts were generated for each label, affording a distinctive assessment of correlations between label prevalence and the magnitude of discrepancies between actual and predicted labels. This analytical approach contributes to a deeper comprehension of the intricate dynamics between label distributions and predictive fidelity, enriching the evaluation of model accuracy and efficacy.

Among all label categories, Logistic Regression consistently outperformed other models, achieving an impressive F1-Score of 70.4% in predicting Tissue types, 56.2% in predicting Cancer Types, and 49% in predicting Cancer Subtypes.

In the context of Tissue type prediction, a closer examination of the confusion matrix reveals notable discrepancies. Specifically, the model tends to predict Stomach Tissues as belonging to the large intestine category, while also showing a tendency to misclassify esophagus samples as Head and Neck or Stomach Tissues. Furthermore, it demonstrates a similarity between Head and Neck and Esophagus tissues in its predictions, and occasionally, Bone Tissue samples are incorrectly categorized as Central Nervous System.

In the evaluation of the model's performance in Cancer Type prediction, several noteworthy trends and challenges come to the forefront. Notably, the model exhibits varying degrees of accuracy when predicting different cancer types. Plasma Cell Myeloma and Ewing's Sarcoma consistently emerge as standout successes in the model's predictions. They tend to align closely with their true labels, yielding correct or nearly correct classifications. Similarly, for T-lymphoblastic leukemia, Neuroblastoma, and Acute Myeloid Leukemia, the model demonstrates a commendable ability to provide predictions that are, for the most part, almost correct. This suggests a reasonable level of success in classifying these cancer types.

However, a significant challenge arises when it comes to the Non-Small Cell Lung Carcinoma Cancer Type. This category, characterized by a substantial number of records, poses a formidable obstacle for the model. The model's accuracy in predicting this Cancer Type stands at only 53%. The intricacies and heterogeneity within this category likely contribute to the model's struggle to achieve more precise predictions. Beyond this, there are instances where the model misclassifies cancer types. For instance, Glioma is predicted as Glioblastoma, Gastric Carcinoma tends to be misclassified as Colorectal Carcinoma, and in certain cases, Other Sarcomas are erroneously categorized as Ovarian Carcinoma.

In the context of Cancer Subtype prediction, an intriguing observation emerges as the model consistently aligns its predictions closely with those generated by the Best Model for Cancer Type classification. This synchronization between predictions at the Cancer Subtype and Cancer Type levels suggests a certain level of intrinsic correlation and underscores the model's ability to maintain overall consistency in its predictions across these classification tiers.

Amidst this consistency, notable challenges persist, particularly when confronting the category of Non-Small Cell Lung Carcinoma

and its diverse array of subtypes. Except for lung adenocarcinoma, this specific set of subtypes presents an enduring conundrum for the model, posing formidable obstacles to accurate prediction.

5.4 Discussion of Results

5.4.1 Tissue Type Classification. Various machine learning models for proteomic data classification show distinct performance trends. LR consistently outperforms other models in evaluated metrics. It's followed closely by SVM and MLP. SVM is particularly effective at minimizing false positives. Surprisingly, Random Forest, initially considered ideal for bioinformatics data, didn't perform well after an extensive grid search.

A common challenge for all models is classifying stomach-related data due to low label counts and misclassification with the large intestine because of similar protein abundance. Comparing SVM and LR, SVM excels in "Central Nervous System", "Kidney", "Head and Neck", and "Lung". K-Nearest Neighbors performs less favorably overall but better for "Kidney". Random Forest excels in "Kidney" but struggles with "Skin" vs. "Central Nervous System". It also performs well in "Head and Neck". Multi-Layer Perceptron is strong in "Kidney" and "Esophagus" but faces challenges in distinguishing "Skin" from "Central Nervous System" and performs well in "Head and Neck".

5.4.2 Cancer Type Classification. In a comprehensive analysis, LR consistently outperforms other models, but common misclassifications persist. "Gastric Carcinoma" is often mistaken for "Colorectal Carcinoma", and "Glioma" and "Glioblastoma" are frequently confused. "Burkitt's Lymphoma" and "B-Cell Non-Hodgkin's Lymphoma" exhibit similar misclassification trends. The K-Nearest Neighbors model misclassifies "Other Solid Carcinoma" as "Head and Neck Carcinoma". Support Vector Machine excels with "Gastric Carcinoma" but struggles with "Other Solid Sarcomas". Random Forest has difficulties classifying "Glioma", primarily with "Other Solid Carcinomas", and "Esophageal Carcinoma" is often confused with "Head and Neck Carcinoma".

The Multi-Layer Perceptron model faces challenges with "Glioma" and "Glioblastoma", as well as misclassifications between "Other Solid Carcinoma" and "Head and Neck Carcinoma" and "Small Cell Lung Carcinoma". "Other Sarcomas" is frequently misclassified, especially with "Thyroid Gland Carcinoma" and "Glioblastoma". "Bladder Carcinoma" is prone to misclassification as "Glioblastoma Carcinoma" and "Breast Carcinoma". "Thyroid Gland Carcinoma" is misclassified as "Glioblastoma", and "Cervical" is confused with "Head and Neck Carcinoma".

These misclassifications often stem from shared biological characteristics among the malignancies. For example, "Gastric Carcinoma" and "Colorectal Carcinoma" share a common origin in the gastrointestinal system and analogous genetic markers. The confusion of "Glioma" with "Glioblastoma" results from their location in the brain and histological similarities. Challenges with B-cell cancers like "Burkitt's Lymphoma" and "B-Cell Non-Hodgkin's Lymphoma" arise from immunophenotypic resemblances. Proximate anatomical locations, such as "Head and Neck Carcinoma" and "Esophageal Carcinoma", increase the likelihood of diagnostic errors, emphasizing the need for a robust understanding of tumor biology in machine learning algorithms for cancer classification.

5.4.3 Cancer Sub-Type Classification. The challenges echo those in Cancer Type classification. Every base model in the study exhibits recurrent misclassifications, primarily due to shared biological traits, histopathological similarities, and genetic markers among subtypes. These difficulties also stem from the complex, overlapping nature of individual records within cancer types.

Furthermore, the dataset's composition adds complexity, with a significant 74% of the dataset consisting of subtypes, each having fewer than 20 records. This results in 23 unique labels, further complicating the classification task.

6 GRAPH NEURAL NETWORK MODELS

6.1 Graph Neural Networks Capabilities

GNNs are powerful tools for analyzing data in graph structures, including biological networks. They allow for classification at different levels, including graph-level, node-level, and edge-level predictions.

Due to the absence of edge information in available datasets, predictions can only be made at the node-level and graph-level. Node-level predictions involve characterizing individual nodes within a correlation network between different cell lines. Selecting an appropriate threshold helps create a concise graph, capturing relevant connections.

Node-level predictions offer a personalized approach, enhancing accuracy and providing insights into cell line heterogeneity. Protein abundance values serve as features for each node.

Graph-level predictions involve analyzing a unique graph for each cell line, leveraging a protein interaction network. Each graph corresponds to a distinct label, and nodes represent proteins with edges denoting connections using the PKN. Integrating protein abundance values as features allows capturing collective behavior across all cell lines.

The challenge in graph-level predictions is associating a graph with each cell line, possibly involving proteomic abundance clustering or identifying significant proteins.

GNNs are explored for predictions in the available datasets, with node-level and graph-level approaches considered for analysis. Further details are presented in subsequent sections.

6.2 Node-Level Predictions

In light of the aforementioned constraints, it becomes evident that the ability to engage in node-level predictions hinges upon the presence of labels for some nodes within the graph. To facilitate this process, each training node must be paired with its corresponding label.

Considering the current dataset landscape, node-level prediction capabilities are constricted to employing individual proteomics records as discrete nodes within the graph. This limitation arises from the fact that labels are possessed exclusively for each patient, thereby necessitating this approach.

In constructing the graph, it is crucial to note that despite extensive research, there is no identifiable cellular network that can accurately distinguish similarities between patients in the proteomics dataset. This ability to assess patient similarity is a crucial factor

that could significantly affect the overall effectiveness of the approach. To address this challenge, the most elementary strategy entails the utilization of a similarity / correlation metric. This metric serves as a means to systematically compare each cell-line record, thereby elucidating the degree of similarity between them. Further elaboration on this approach will be provided in the subsequent subsubsection.

6.2.1 Defining the Graph. The study began by examining correlation matrices with four primary metrics: Cosine, Pearson, Spearman, and Kendall. Positive values were considered, avoiding self-loops, as GNN can incorporate them.

Histogram analysis revealed that Kendall often scored below 0.5, potentially limiting model performance. Spearman occasionally reached high scores, up to 0.8, while Cosine and Pearson showed similar patterns. To find the best threshold, similarity/correlation metrics were used across a range from 0 to 1, in 0.1 increments. This helped understand how thresholds influenced data characteristics, including node count, edge count, and average neighbor count. Key findings showed that an average neighbor value of around 50 achieved the desired node count while maintaining a good number of edges. However, exploring a higher threshold of 100 increased node count and connectivity but resulted in a denser network. Selecting the right threshold in graph construction and data analysis is crucial.

6.2.2 Base Models Validation. To determine the most suitable metric, a comprehensive assessment was conducted using a straightforward GCN model. Findings from these experiments are available and offer valuable insights into metric performance.

Additional techniques were employed to enhance results, including reducing the number of edges in the graph by testing various values for parameter k (0.5, 2, and 4). Higher k values resulted in a greater reduction in the number of edges, with the order being 0.5 > 2 > 4.

In the pursuit of further improvements, an examination was conducted to select the most suitable similarity/correlation metric. The Kendall correlation metric showed the highest F1-Score but limited the dataset to 660 nodes. As a result, the cosine similarity metric, ranking second in F1-Score among the considered metrics, was chosen, aligning with recent studies like MOGONET.

To fine-tune the model, hyperparameter optimization was carried out using the Optuna framework. The model was refined using 50 nodes and the cosine similarity metric, and different average node numbers (50, 75, and 100) were evaluated with various similarity/correlation metrics.

6.2.3 Node-Level Results. To facilitate a comprehensive comparison between the baseline and the GCN node-level approach, identical graphs were generated: a bar chart was employed to provide a visual comparison of the various model performances over four metrics: F1-Score, Precision, Recall, and Accuracy; a confusion matrix to depict the model's predictions against the true labels as determined by the GCN node-level model.

This analytical approach allows for a rigorous evaluation of the model's effectiveness and provides a clear understanding of its performance in comparison to the baseline.

6.2.4 Discussion of Results. In Tissue Type, The performance of the Graph Neural Network (GNN) model in classifying tissue types reveals distinct characteristics. Key metrics, such as F1-Score, Accuracy, Precision, and Recall, exhibit suboptimal results, typically around 50-60%. The GNN model tends to favor predicting Ovary Tissue Type. Notably, some labels like 'Breast' and 'Haematopoietic and Lymphoid' are challenging to predict accurately, despite the latter having a significant number of records. In contrast, 'Esophagus' and 'Peripheral Nervous System' are predicted with higher accuracy. However, 'Lung,' which has the most records, is only predicted correctly in about 41% of cases. These findings suggest a potential correlation between ovarian records and the remaining proteomic data when selecting connections in the GNN model.

In Cancer Type, the model's performance across all key metrics remains subpar, typically around 35%. Certain cancer types pose significant challenges for the model, such as "Other Solid Carcinomas", "Kidney Carcinoma", "Glioblastoma", and more. On a positive note, the model excels in predicting "Thyroid Gland Carcinoma" and "Acute Myeloid Leukemia". It tends to confuse certain cancer types, like "Burkitt's Lymphoma" with "Glioma" and "Glioblastoma" with lung carcinomas. Additional misclassifications occur between "Kidney Carcinoma" and "Burkitt's Lymphoma", as well as "Ovarian Carcinoma" and "Gastric Carcinoma", among others.

In Cancer Subtype, the model's performance metrics register at their lowest, with values around 27%. Notably, a significant portion of labels, approximately 32%, is misclassified by the model. There are exceptions where the model accurately identifies "Hepatocellular Carcinoma" and "Neuroblastoma". The overall poor performance presents challenges, and the relationship between subtype and Cancer Type classification is not apparent in the current analysis. This underscores the need for further refinement and investigation to enhance the model's performance in this complex domain.

7 FUTURE WORK AND FINAL CONSIDERATIONS

7.1 Graph-Level Predictions

This study introduces a graph-level prediction model that uses a proteomic network to connect proteins, improving classification through protein-protein interactions.

In this approach, a network was built with nodes representing proteins from the proteomics dataset. Connections were established using an existing proteomic network, and Tissue Type labels were integrated for classification. This created individual graphs for each cell line with distinct features and labels.

To construct the proteomic network, the STRING database was used, providing extensive protein interaction information.

Efforts were made to develop an effective GNN model for graph classification, involving hyperparameter tuning with Optuna. However, initial tests resulted in a low F1-Score, below 30%. Further exploration using a unique GCN layer coupled with an MLP model showed improved performance, suggesting that the quality of the generated graph might not be the primary issue.

The main problem was the uniform graph approach for every patient with distinct node features. To achieve proficient graph-level classification, it's essential to explore creating a unique graph for

each patient with distinct labels, requiring more robust techniques in the future.

7.2 Future Work

There will always be work to improve, like improving the node-level methodology and refining the graph-level approach. For node-level enhancement, addressing dataset imbalance can involve using GraphSMOTE, which generates new samples in an embedding space to maintain authenticity. An edge generator models relation information to bolster GNN classifier performance, especially in scenarios with class imbalances.

On the graph-level, a customized approach for each cell line in the proteomics dataset is recommended. Clustering can group similar cell lines based on protein abundance profiles, allowing for the construction of unique graphs for each cluster, facilitating graph-level analysis. It's worth noting that this approach is effective when prevalent proteins differ among clusters. If the same proteins exist in all clusters, the approach results in a homogeneous graph.

The thesis also aimed to investigate the model's effectiveness in predicting drug responses using the IC50 Dataset. Unfortunately, time constraints limited this exploration, but it remains a promising avenue for future research.

REFERENCES

[1] Abien Fred Agarap. 2018. Deep Learning using Rectified Linear Units (ReLU). *CoRR* abs/1803.08375 (2018). arXiv:1803.08375 http://arxiv.org/abs/1803.08375

[2] David Ahmedt-Aristizabal, Mohammad Ali Armin, Simon Denman, Clinton Fookes, and Lars Petersson. 2022. A survey on graph-based deep learning for computational histopathology. *Computerized Medical Imaging and Graphics* 95 (2022), 102027. https://doi.org/10.1016/j.compmedimag.2021.102027

[3] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. 2017. Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*. Ieee, 1–6.

[4] Alexei Botchkarev. 2018. Performance metrics (error measures) in machine learning regression, forecasting and prognostics: Properties and typology. *arXiv preprint arXiv:1809.03006* (2018).

[5] Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (01 Oct 2001), 5–32. https://doi.org/10.1023/A:1010933404324

[6] Nancy Chinchor. 1992. MUC-4 Evaluation Metrics. In *Fourth Message Uunderstanding Conference (MUC-4): Proceedings of a Conference Held in McLean, Virginia, June 16-18, 1992*. https://aclanthology.org/M92-1002

[7] Thomas Cover and Peter Hart. 1967. Nearest neighbor pattern classification. *IEEE transactions on information theory* 13, 1 (1967), 21–27.

[8] Austin Derrow-Pinion, Jennifer She, David Wong, Oliver Lange, Todd Hester, Luis Perez, Marc Nunkesser, Seongjae Lee, Xueying Guo, Brett Wiltshire, Peter W. Battaglia, Vishal Gupta, Ang Li, Zhongwen Xu, Alvaro Sanchez-Gonzalez, Yujia Li, and Petar Velickovic. 2021. ETA Prediction with Graph Neural Networks in Google Maps. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. ACM. https://doi.org/10.1145/3459637.3481916

[9] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph Neural Networks for Social Recommendation. In *The World Wide Web Conference* (San Francisco, CA, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 417–426. https://doi.org/10.1145/3308558.3313488

[10] Tom Fawcett. 2006. An introduction to ROC analysis. *Pattern recognition letters* 27, 8 (2006), 861–874.

[11] Thomas Gaudelet, Ben Day, Arian R Jamasb, Jyothish Soman, Cristian Regep, Gertrude Liu, Jeremy B R Hayter, Richard Vickers, Charles Roberts, Jian Tang, David Roblin, Tom L Blundell, Michael M Bronstein, and Jake P Taylor-King. 2021. Utilizing graph machine learning within drug discovery and development. *Briefings in Bioinformatics* 22, 6 (05 2021). https://doi.org/10.1093/bib/bbab159 arXiv:https://academic.oup.com/bib/article-pdf/22/6/bbab159/41087478/bbab159.pdf bbab159.

[12] Emanuel Gonçalves, Rebecca C. Poulos, Zhaoxiang Cai, Syd Barthorpe, Srikanth S. Manda, Natasha Lucas, Alexandra Beck, Daniel Bucio-Noble, Michael Dausmann, Caitlin Hall, Michael Hecker, Jennifer Koh, Howard Lightfoot, Sadia Mahboob, Iman Mali, James Morris, Laura Richardson, Akila J. Seneviratne, Rebecca Shepherd, Erin Sykes, Frances Thomas, Sara Valentini, Steven G. Williams, Yangxiu Wu, Dylan Xavier, Karen L. MacKenzie, Peter G. Hains, Brett Tully, Phillip J. Robinson, Qing Zhong, Mathew J. Garnett, and Roger R. Reddel. 2022. Pan-cancer proteomic map of 949 human cell lines. *Cancer Cell* 40, 8 (2022), 835–849.e8. https://doi.org/10.1016/j.ccell.2022.06.010

[13] William L. Hamilton. 2020. Graph Representation Learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 14, 3 (2020), 1–159.

[14] Leon Hetzel, David S Fischer, Stephan Günnemann, and Fabian J Theis. 2021. Graph representation learning for single-cell biology. *Current Opinion in Systems Biology* 28 (2021), 100347.

[15] Like Hui and Mikhail Belkin. 2020. Evaluation of Neural Architectures Trained with Square Loss vs Cross-Entropy in Classification Tasks. *CoRR* abs/2006.07322 (2020). arXiv:2006.07322 https://arxiv.org/abs/2006.07322

[16] Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. 2016. Molecular graph convolutions: moving beyond fingerprints. *Journal of Computer-Aided Molecular Design* 30, 8 (01 Aug 2016), 595–608. https://doi.org/10.1007/s10822-016-9938-8

[17] S. Lawrence, C.L. Giles, Ah Chung Tsoi, and A.D. Back. 1997. Face recognition: a convolutional neural-network approach. *IEEE Transactions on Neural Networks* 8, 1 (1997), 98–113. https://doi.org/10.1109/72.554195

[18] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (01 May 2015), 436–444. https://doi.org/10.1038/nature14539

[19] Warren S McCulloch and Walter Pitts. 1943. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics* 5, 4 (1943), 115–133.

[20] C. Nebauer. 1998. Evaluation of convolutional neural networks for visual recognition. *IEEE Transactions on Neural Networks* 9, 4 (1998), 685–696. https://doi.org/10.1109/72.701181

[21] S. Gopal Krishna Patro and Kishore Kumar Sahu. 2015. Normalization: A Preprocessing Stage. *CoRR* abs/1503.06462 (2015). arXiv:1503.06462 http://arxiv.org/abs/1503.06462

[22] Leif E Peterson. 2009. K-nearest neighbor. *Scholarpedia* 4, 2 (2009), 1883.

[23] Sebastian Ruder. 2016. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747* (2016).

[24] Benjamin Sanchez-Lengeling, Emily Reif, Adam Pearce, and Alexander B. Wiltschko. 2021. A Gentle Introduction to Graph Neural Networks. *Distill* (2021). https://doi.org/10.23915/distill.00033 https://distill.pub/2021/gnn-intro.

[25] Benjamin Sanchez-Lengeling, Emily Reif, Adam Pearce, and Alexander B Wiltschko. 2021. A gentle introduction to graph neural networks. *Distill* 6, 9 (2021), e33.

[26] Yutaka Sasaki et al. 2007. The truth of the F-measure. *Teach tutor mater* 1, 5 (2007), 1–5.

[27] Hossein Sharifi-Noghabi, Olga Zolotareva, Colin C Collins, and Martin Ester. 2019. MOLI: multi-omics late integration with deep neural networks for drug response prediction. *Bioinformatics* 35, 14 (07 2019), i501–i509. https://doi.org/10.1093/bioinformatics/btz318 arXiv:https://academic.oup.com/bioinformatics/article-pdf/35/14/i501/28913828/btz318.pdf

[28] Jonathan Shlomi, Peter Battaglia, and Jean-Roch Vlimant. 2020. Graph neural networks in particle physics. *Machine Learning: Science and Technology* 2, 2 (2020), 021001.

[29] P. Subramaniam and M. J. Kaur. 2019. Review of Security in Mobile Edge Computing with Deep Learning. In *2019 Advances in Science and Engineering Technology International Conferences (ASET) (2019 Advances in Science and Engineering Technology International Conferences (ASET))*. 1–5. https://doi.org/10.1109/ICASET.2019.8714349

[30] Tongxin Wang, Wei Shao, Zhi Huang, Haixu Tang, Jie Zhang, Zhengming Ding, and Kun Huang. 2021. MOGONET integrates multi-omics data using graph convolutional networks allowing patient classification and biomarker identification. *Nature Communications* 12, 1 (08 Jun 2021), 3445. https://doi.org/10.1038/s41467-021-23774-w

[31] Feng Xia, Ke Sun, Shuo Yu, Abdul Aziz, Liangtian Wan, Shirui Pan, and Huan Liu. 2021. Graph Learning: A Survey. *CoRR* abs/2105.00696 (2021). arXiv:2105.00696 https://arxiv.org/abs/2105.00696

[32] Yuan Zhuang, Zhenguang Liu, Peng Qian, Qi Liu, Xiang Wang, and Qinming He. 2020. Smart Contract Vulnerability Detection using Graph Neural Network.. In *IJCAI*. 3283–3290.

Received 31 October 2023