CORA Cork Open Research Archive
Cartlann Taighde Oscailte Chorcaí

Title	MANet: a deep learning framework for multi-cancer microarray analysis, classification, and visualization
Authors	Younis, Haseeb;Azeem, Muhammad;Ronan, Isabel;Minghim, Rosane
Publication date	2024
Original Citation	Younis, H., Azeem, M., Ronan, I. and Minghim, R. (2024) 'MANet: a deep learning framework for multi-cancer microarray analysis, classification, and visualization', in 2024 International Conference on Frontiers of Information Technology (FIT), Islamabad, Pakistan, 9-10 December 2024, pp. 1–6. https://doi.org/10.1109/FIT63703.2024.10838450
Type of publication	Conference item
Link to publisher's version	10.1109/FIT63703.2024.10838450
Rights	© 2024, IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.
Download date	2026-11-17 06:40:41
Item downloaded from	https://hdl.handle.net/10468/17086

UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

MANet: A Deep Learning Framework for Multi-Cancer Microarray Analysis, Classification, and Visualization

Haseeb Younis*†[a‡], Muhammad Azeem*[b], Isabel Ronan*[c], Rosane Minghim*[d],
* Department of Computer Science, University College Cork, Ireland
† Machine Learning Group, Intel Ireland Limited, Ireland
[a] haseeb.younis@cs.ucc.ie, haseeb.younis@intel.com, [b] m.azeem@cs.ucc.ie, [c] i.ronan@cs.ucc.ie, [d] r.minghim@cs.ucc.ie

***Abstract*—Machine learning (ML) methods have been used much more frequently in recent years to extract gene expression data from microarray studies, especially in cancer research. Even after the continued interest in applying ML to scientific cancer research, there is still no universal approach for categorizing cancer microarray data. A system is needed that can detect and classify a normal profile and a cancer profile, specifying the type of cancer. Due to the variance and high dimensionality of microarray data, it is difficult to extract the relevant descriptors and provide insights that can be helpful in identifying cancer types and stages. In this paper, we proposed MANet: a methodology using cancer microarray data based on Deep Learning (DL) to classify 13 different types of cancers as well as normal profiles. To implement this methodology, we used a Curated Microarray Database (CuMiDa) that has 78 datasets for different types of cancers. Due to the diverse feature vectors for each dataset, we used Principal Component Analysis (PCA) for uniform feature engineering. Our single model has the capability to learn the patterns, cluster instances into their corresponding classes and classify the cancer. We also used the Uniform Manifold Approximation and Projection (UMAP) to visualise the instance separation on original data. Additionally, this UMAP visualises segregation done by our methodology in low dimensions. Using the proposed methodology, we achieved approximately 80% average accuracy, precision, recall, and F1 Score for 14 classes using a single model.**

***Index Terms*—Cancer Microarray, Principal Component Analysis, Microarray Cancer Classification, Transfer Learning**

I. Introduction

Causing 10 million deaths worldwide in 2020, cancer is a growing health problem according to the World Health Organization [1]. Cancer can be described as cellular amalgamations that emerge from certain body parts; these cells typically move to distant metastatic areas [2].

The fast-developing field of molecular diagnostics is utilizing gene-based biomarkers to provide insights into disease causes. Previously, diagnosis and prognosis evaluation relied on indirect markers, ignoring changes in gene expression. Microarrays offer a high-throughput method for global expression analysis, enabling the simultaneous interrogation of thousands of DNA molecules. This results in better early diagnosis and novel therapeutic methods for cancer, offering valuable information on disease pathophysiology, development, resistance to therapy, and reactions to cellular microenvironments. Researchers can observe gene expression levels in abnormal and healthy tissues, compute differentially expressed amounts of each gene in malignant and non-cancerous samples, and use pathway and gene ontology enrichment analyses to uncover possible biological processes involved in diseases. [3].

DNA Microarrays are slides that measure the fluorescence intensity of fluorescently-labelled cDNA molecules, allowing for the simultaneous monitoring of thousands of genes. These slides, known as DNA/Gene chips, are pre-spotted with hundreds of probes complementary to the fluorescently-marked cDNA molecules. The ability to ascribe distinct patterns of gene expression to individual genes is made possible by the known positions of the probes on the chip. Identifying patterns in microarray data is crucial for disease identification and prognostication in the early stages. Machine learning (ML) methods can offer quick and precise expression pattern recognition, making them useful tools for cancer prediction models, tumour diagnosis, stratification, and survival prediction [4].

Numerous facets of ML-related cancer research, including RNA-Seq, microarray, and cancer classification, are explored in biomarker research for various cancer types. The study [5] addresses technical issues with cancer prediction models and measurement instruments, highlighting the potential of detecting biomarker gene expression patterns to predict cancer risk and guide personalized care. Several studies have utilized machine learning techniques to classify cancer using microarray datasets. Neuroevolution, a branch of ML that blends neural networks and evolutionary computation, was used in FS-NEAT. The method successfully identified gene subsets that convey biologically significant information, with 177 genes found [6]. Multiple ML algorithms, including Decision Tree, K-nearest Neighbour, and Random Forest, were used to classify microarray samples and identify cancer types [7]. However, the study's limitations include the inability to achieve good accuracy and find the best algorithm for all datasets.

There is a database of cancer microarray named CuMiDA [6] that contains around 78 datasets of 13 different kinds of cancers. Several other studies [8]–[10] have used machine

This publication has emanated from research supported in part by a grant from Science Foundation Ireland under Grant number 18/CRT/6222.
‡ Primary and Corresponding Author
979-8-3315-1050-3/24/$31.00 ©2024 IEEE

learning (ML) approaches to improve CuMiDa results. However, the dimensionality of microarray data and low instances hindered the models' performance. Bohere et al. [11] and Barbieri et al. [12] conducted experiments using various ML approaches to improve results but couldn't agree on an algorithm for all datasets. Furthermore, like the other pre-trained models [13] for cancer classification based on images, there is a pretrained network needed that could improve microarray cancer data classification. As Microarray datasets have high dimensionality, we can take advantage of DL to find the pattern that can lead us to identify the disease with more confidence and reliability [9], [14]. In this paper, we propose MANet, a methodology that uses PCA [15] to converge the features into a uniform matrix, and a DL-based algorithm to identify patterns from Microarray datasets and classify multiple types of cancers. MANet achieved more than 80% accuracy, precision, recall, and F1 score. We also used visualization methods such as UMAP [16] and Violin Plots [17] to project the data separation and sparsity handling using our model, showing the robustness of our methodology. The technical details of our methodology are given in Section II, and the analysis is given in Section III.

II. Methodology

To implement our methodology, we used the datasets from Grisci et al. [6]. There were 78 datasets of different cancer types that have different kinds of features and unbalanced classes. Technical details of our methodology are outlined below.

A. Dataset

Multiple microarray datasets for different cancer subtypes, such as colorectal, gastric, pancreatic, liver, bladder, lung, throat, renal, brain, prostate, ovarian, leukemia, and breast cancers, were downloaded via GEO [18] in order to collect the dataset. A minimum of six samples per condition, detailed protocol descriptions, xenograft technique, availability of raw data, and the avoidance of chemotherapies, gene treatments, interfering agents, knockdown cultures, and induced mutations were among the selection criteria used for the datasets. Seventy-eight microarray datasets, including single- and dual-channel samples, were hand-picked via a rigorous curation procedure. Following collection, the datasets were subjected to normalization and analysis using the R package arrayQualityMetrics [19] to assess the sample quality of the selected microarrays. If a sample had poor quality in at least half of the parameters that arrayQualityMetrics measured, it was removed. The last step involved carefully choosing each final normalized matrix to remove any unwanted probes that had nothing to do with the nucleic acid sequences. Seventy-three of the seventy-eight datasets were used because the class information was missing.

B. Data Preprocessing and Transformation Using PCA

The datasets have varying features and classes, making it challenging to train a model. To address this, data is integrated and transformed using PCA, followed by standardisation of data to make sure that each descriptor has a standard deviation of one and a mean of zero.

1) Principal Component Analysis (PCA)

PCA is used to reduce the dimensionality of the data while retaining most of the variance. The steps involved are:

1 Determine the Number of Components

The number of components $n_{\text{components}}$ is determined by the smaller of the number of samples and the number of features, adjusted to ensure at least one component is used:

$$n_{\text{components}} = \min(N, M) - 1 \tag{1}$$

Where:

- N is the number of samples.
- M is the number of features.

2 Compute Covariance Matrix

The covariance matrix $\mathbf{C}$ of the standardized data $\mathbf{X}'$ is calculated as:

$$\mathbf{C} = \frac{1}{N-1}(\mathbf{X}')^T\mathbf{X}' \tag{2}$$

3 Eigen Decomposition

Eigen decomposition of the covariance matrix yields eigenvalues and eigenvectors:

$$\mathbf{C}\mathbf{v}_i = \lambda_i \mathbf{v}_i \tag{3}$$

Where:

- λ_i are the eigenvalues.
- $\mathbf{v}_i$ are the corresponding eigenvectors.

The eigenvectors corresponding to the largest $n_{\text{components}}$ eigenvalues are selected as the principal components.

4 Transform Data

The standardized data is projected onto the space defined by the principal components:

$$\mathbf{X}_{\text{PCA}} = \mathbf{X}'\mathbf{V} \tag{4}$$

Where:

- $\mathbf{X}_{\text{PCA}}$ is the data after PCA transformation.
- $\mathbf{V}$ is the matrix of selected eigenvectors.

C. Model Architecture

The model architecture consists of the following layers:

1) **Input Layer**:
   Input Shape $(T, 1)$ where T is the length of the input sequence.
2) **Convolutional Layer (Conv1D)**:
   - Filters: 64
   - Kernel Size: 5
   - Activation: ReLU

   The convolutional operation can be represented as:

$$y_{i,k} = \text{ReLU}\left(\sum_{j=0}^{4} x_{i+j,c} \cdot w_{j,k} + b_k\right) \tag{5}$$

   Where:

- $x_{i+j,c}$ is the input at position $i+j$ for channel c.
- $w_{j,k}$ are the weights of the k-th filter.
- b_k is the bias for the k-th filter.
- $y_{i,k}$ is the output of the convolution at position i for the k-th filter.

3) **Batch Normalization**:
Normalizes the activations of the previous layer:

$$y_{i,k} = \frac{y_{i,k} - \mu_k}{\sqrt{\sigma_k^2 + \epsilon}} \cdot \gamma_k + \beta_k \tag{6}$$

Where:
- μ_k is the mean of the activations.
- σ_k^2 is the variance of the activations.
- ϵ is a small constant for numerical stability.
- γ_k, β_k are learnable parameters.

4) **Max Pooling Layer (MaxPooling1D)**:

- Pool Size: 2

The max pooling operation can be represented as:

$$y_{i,k} = \max(x_{2i,k}, x_{2i+1,k}) \tag{7}$$

Where:
- $x_{2i,k}$ and $x_{2i+1,k}$ are the input values in the pool.
- $y_{i,k}$ is the output of the pooling operation.

5) **Dropout Layer (Dropout)**:

- Rate: 0.4

The dropout operation can be represented as:

$$y_{i,k} = \begin{cases} 0 & \text{with probability } 0.4 \\ \frac{x_{i,k}}{0.6} & \text{with probability } 0.6 \end{cases} \tag{8}$$

Where:
- $x_{i,k}$ is the input.
- $y_{i,k}$ is the output after dropout.

6) **Convolutional Layer (Conv1D)**:

- Filters: 128
- Kernel Size: 3
- Activation: ReLU

The convolutional operation can be represented as:

$$y_{i,k} = \text{ReLU}\left(\sum_{j=0}^{2} x_{i+j,c} \cdot w_{j,k} + b_k\right) \tag{9}$$

Where:
- $x_{i+j,c}$ is the input at position $i+j$ for channel c.
- $w_{j,k}$ are the weights of the k-th filter.
- b_k is the bias for the k-th filter.
- $y_{i,k}$ is the output of the convolution at position i for the k-th filter.

7) **Batch Normalization**:
Normalizes the activations of the previous layer:

$$y_{i,k} = \frac{y_{i,k} - \mu_k}{\sqrt{\sigma_k^2 + \epsilon}} \cdot \gamma_k + \beta_k \tag{10}$$

8) **Max Pooling Layer (MaxPooling1D)**:

- Pool Size: 2

The max pooling operation can be represented as:

$$y_{i,k} = \max(x_{2i,k}, x_{2i+1,k}) \tag{11}$$

9) **Dropout Layer (Dropout)**:

- Rate: 0.4

The dropout operation can be represented as:

$$y_{i,k} = \begin{cases} 0 & \text{with probability } 0.4 \\ \frac{x_{i,k}}{0.6} & \text{with probability } 0.6 \end{cases} \tag{12}$$

10) **Convolutional Layer (Conv1D)**:

- Filters: 256
- Kernel Size: 3
- Activation: ReLU

The convolutional operation can be represented as:

$$y_{i,k} = \text{ReLU}\left(\sum_{j=0}^{2} x_{i+j,c} \cdot w_{j,k} + b_k\right) \tag{13}$$

11) **Batch Normalization**:
Normalizes the activations of the previous layer:

$$y_{i,k} = \frac{y_{i,k} - \mu_k}{\sqrt{\sigma_k^2 + \epsilon}} \cdot \gamma_k + \beta_k \tag{14}$$

12) **Max Pooling Layer (MaxPooling1D)**:

- Pool Size: 2

The max pooling operation can be represented as:

$$y_{i,k} = \max(x_{2i,k}, x_{2i+1,k}) \tag{15}$$

13) **Dropout Layer (Dropout)**:

- Rate: 0.4

The dropout operation can be represented as:

$$y_{i,k} = \begin{cases} 0 & \text{with probability } 0.4 \\ \frac{x_{i,k}}{0.6} & \text{with probability } 0.6 \end{cases} \tag{16}$$

14) **Flatten Layer (Flatten)**:
This layer reshapes the input to a 1D array. If the input shape is (n, m), the output shape will be $(n \cdot m)$.

15) **Dense Layer (Fully Connected Layer)**:

- Units: 512
- Activation: ReLU

The dense layer operation can be represented as:

$$y_i = \text{ReLU}\left(\sum_{j=0}^{N-1} x_j \cdot w_{j,i} + b_i\right) \tag{17}$$

Where:

- x_j is the input from the previous layer.
- $w_{j,i}$ are the weights.
- b_i is the bias.
- y_i is the output.

16) **Dropout Layer (Dropout)**:

- Rate: 0.5

The dropout operation can be represented as:

$$y_{i,k} = \begin{cases} 0 & \text{with probability } 0.5 \\ \frac{x_{i,k}}{0.5} & \text{with probability } 0.5 \end{cases} \tag{18}$$

17) **Dense Layer (Fully Connected Layer)**:

- Units: 256
- Activation: ReLU

The dense layer operation can be represented as:

$$y_i = \text{ReLU}\left(\sum_{j=0}^{N-1} x_j \cdot w_{j,i} + b_i\right) \tag{19}$$

18) **Dropout Layer (Dropout)**:

- Rate: 0.5

The dropout operation can be represented as:

$$y_{i,k} = \begin{cases} 0 & \text{with probability } 0.5 \\ \frac{x_{i,k}}{0.5} & \text{with probability } 0.5 \end{cases} \tag{20}$$

19) **Output Dense Layer (Fully Connected Layer)**:

- Units: *num_classes*
- Activation: Softmax

The output layer in case of multiclass classification can be represented as:

$$y_i = \frac{\exp(z_i)}{\sum_{j=1}^{num_classes} \exp(z_j)}, \quad \text{for } i = 1, \ldots, num_classes \tag{21}$$

Where:

$$z_i = \sum_{j=0}^{N-1} x_j \cdot w_{j,i} + b_i \tag{22}$$

The detailed model summary is shown in Figure 1.

Hyperparameters and Callbacks:

We used a different range of hyperparameters to train the model, such as Adam optimizer with learning rates [0.01,0.001,0.0001,0.00001], and batch size [16,32,64,128,256]. The best parameters were adam with 0.0001, and batch size of 32. For model callbacks, the early stopping criteria to stop the training with the patience of 40 epochs and save the best model with minimum validation loss was used. Furthermore, a method to reduce the learning rate by a factor of 0.1 if validation loss doesn't improve for 10 epochs, with a minimum learning rate of 1e-6, was also used to make sure that the loss value doesn't stop decreasing.

Layer (type)	Output Shape	Param #
conv1d_1 (Conv1D)	(None, 352, 64)	384
batch_norm_1 (BatchNormalization)	(None, 352, 64)	256
max_pooling1d_1 (MaxPooling1D)	(None, 176, 64)	0
dropout_1 (Dropout)	(None, 176, 64)	0
conv1d_2 (Conv1D)	(None, 174, 128)	24704
batch_norm_2 (BatchNormalization)	(None, 174, 128)	512
max_pooling1d_2 (MaxPooling1D)	(None, 87, 128)	0
dropout_2 (Dropout)	(None, 87, 128)	0
conv1d_3 (Conv1D)	(None, 85, 256)	98560
batch_norm_3 (BatchNormalization)	(None, 85, 256)	1024
max_pooling1d_3 (MaxPooling1D)	(None, 42, 256)	0
dropout_3 (Dropout)	(None, 42, 256)	0
flatten_1 (Flatten)	(None, 10752)	0
dense_1 (Dense)	(None, 512)	5505536
dropout_4 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 256)	131328
dropout_5 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 14)	3598

Fig. 1. Summary of Our Model Architecture

D. Visual Analysis of Our Methodology

We used UMAP to visualize the distribution of data in two dimensions to show the robustness of our methodology. We also used the voilin plot, which is a combination of Kernal Density Estimate (KDE) and boxplots, to show the feature value ranges and distribution. The mathematical representation of this methodology is given below in equation 23.

Mathematical Representation for KDE

In KDE, the estimate of the density function of a random variable is given by:

$$\hat{f}_h(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right) \tag{23}$$

Where:

- n is the number of data points
- K is the kernel (a function that ensures that the density estimate is non-negative and integrates to 1)
- h is the bandwidth (determines the smoothness of the density estimate)
- $X_1, \ldots, X_n$ are the observed data points

E. Evaluation Measures

The following evaluation measures presented in the equations below were used to evaluate our methodology's performance.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{24}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{25}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{26}$$

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{27}$$

Where:

TP indicates True Positive, meaning the cancer profile is classified as a cancer profile.

TN indicates True Negative, meaning the normal profile is classified as a normal profile.

FP indicates False Positive, meaning the normal profile is classified as a cancer profile.

FN indicates False Negative, meaning the cancer profile is classified as a normal profile.

III. Results and Discussion

This section deals with the comparative analysis of results, our findings, and the validity of our methodology.

A. Dataset Analysis

In the database used, two bladder, two brain, thirteen breast, ten colorectal, three gastric, nine leukemia, ten liver, six lung, four ovary, one pancreatic, ten prostate, four renal, and four throat are among the thirteen distinct cancer types represented by the 78 microarray datasets. These datasets were unbalanced and ranged from two classes (cancer vs. normal) to seven classes (various stages and types of cancer). The lack of class information forced us to eliminate Gastric_GSE22804, Liver_GSE62043, Prostate_GSE8511, Prostate_GSE38241, and Prostate_GSE60329 from the experiments. The datasets have a small number of instances and a high number of different features for each dataset that lead to model underfitting and the "curse of dimensionality". This feature variance also makes it difficult to integrate a dataset into a single database or model. To overcome this problem, we used PCA to convert the feature set into a uniform shape for the model embeddings. However, the number of instances is not the same, which creates a problem in making a consistent feature vector; this problem was solved using padding. The maximum number of profiles in the largest dataset (Liver_GSE14520_U133A) was 357, so the padded length was fixed to that length. Each dataset has multiple categories of cancer, e.g. for Leukemia_GSE28497 (B-CELL-ALL, B-CELL-ALL-TCF3-PBX1, B-CELL-ALL-HYPERDIP, B-CELL-ALL-HYPO, B-CELL-ALL-MLL, B-CELL-ALL-T-ALL, B-CELL-ALL-EVT6-RUNX1); other datasets have different cancer categories according to their types. Furthermore, the number of profiles for each category is very small; using all types will lead to many classes, which will create low variance and bad model performance. To overcome this problem, we assigned the super-category to the sub-categories of each cancer type, thereby reducing the number of categories to 13 major cancer types.

B. Results of Our Methodology

After dataset preprocessing, we designed the network according to the nature of our data. The network has convolutional layers, followed by batch normalisation layers, a max-pooling layer, and an adjusted dropout layer. To handle overfitting, the methodology includes early stopping criteria and checkpointing. Our methodology performed very well for the 14 classes and handled the cancer microarray data with good performance. The results are shown in Figure 2, and the data point distribution done by our model is shown in Figure 3.

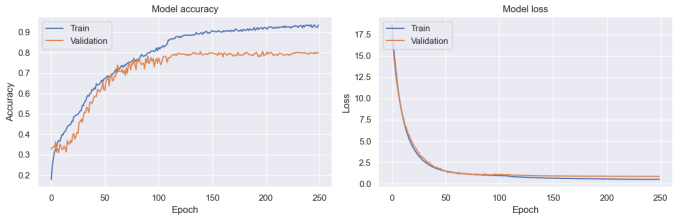


Fig. 2. Results of Our Methodology

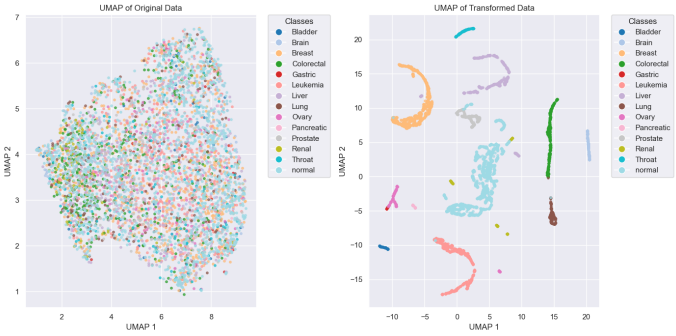


Fig. 3. Data Point Distribution of Original Dataset and Separated by our Model

The figures above show the results of our methodology as well as separation performance on the multi-class data. The points shown in Figure 3 represent the distribution of different cancers using different colours. The training accuracy in Figure 2 is increasing continuously due to the learning rate reduction on the plateau; validation accuracy is 80%.

C. Robustness of our Methodology

The proposed methodology achieved very promising results and performed very well on microarray cancer data with high variance features and low correlation. The model was trained using 250 epochs with early stopping criteria and a tolerance of 40 epochs. The reason for such tolerance was to make sure the model could learn on the dataset with high variance descriptors.

The fraction distribution for cancer biomarkers is shown in Figure 4. This figure shows the very sparse distribution of feature values which makes learning very difficult for the model, especially when dealing with a large number of classes. Our model handles this variance, showing that it is capable of handling biological data with outlier and large-range fractions. The utilization of UMAP demonstrates the model's capacity to produce distinct clusters for various classes; this capacity may be specifically utilized for labelling the unsupervised

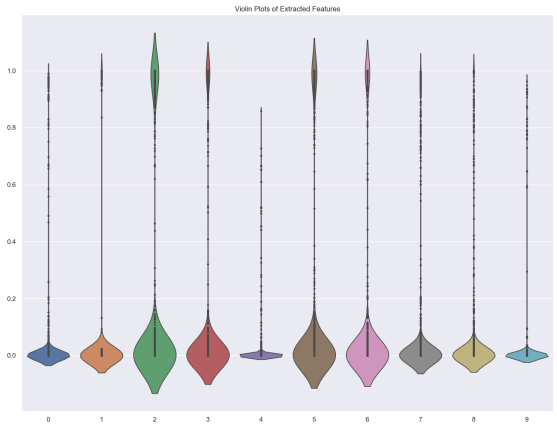


Fig. 4. Distribution of Top 10 Extracted Features

microarray cancer dataset. MANet has successfully converged the data points in a way that a model can distinguish them and learn easily. The learning ability of this architecture shows that it can be used to train more microarray data with other cancer types and can be used like other transfer learning models, such as MobileNet.

IV. Conclusion

Cancer is one of the most deadly diseases; there are various methods for collecting samples and classifying cancer types and stages. Microarray is one of the most reliable technologies for illness analysis, providing a wide range of descriptors. ML and other computational approaches are working hard to develop a mechanism for shortlisting relevant descriptors and classifying cancer. We proposed MANet, a lightweight methodology, to classify 13 different kinds of cancers and normal profiles based on the microarray descriptors. The MANet can work with numerous classes and create different clusters for each class. The biological dataset has large dimensions with a high variance of fractions, making it difficult to process and extract insights from relevant biomarkers. This architecture's learning ability reveals that it can perform well on such biological datasets and classify the different cancer types with accurate and reliable performance. Additionally, it has the capacity to manage descriptors with high variance and low correlation. Furthermore, we visualized the instance separation on the original data using the UMAP. This UMAP also shows the low-dimensional segregation that our methods carry out. This model can be used as a pre-trained model to learn different patterns for different cancer modalities in the future. Moreover, it can be used to train and predict cancer types and stages using microarray data.

Code and Datasets Availability

The datasets used in this research are available at CuMiDa and code will be provided after acceptance and publication at GitHub.

Acknowledgment

This publication has emanated from research supported in part by a grant from Science Foundation Ireland under Grant number 18/CRT/6222. For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

References

[1] W. H. O. (WHO), "Cancer," https://www.who.int/news-room/fact-sheets/detail/cancer, 2024, accessed: 2024-07-14.

[2] S. Shandilya and C. Chandankhede, "Survey on recent cancer classification systems for cancer diagnosis," in *2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*. IEEE, 2017, pp. 2590–2594.

[3] P. F. Macgregor and J. A. Squire, "Application of microarrays to the analysis of gene expression in cancer," *Clinical chemistry*, vol. 48, no. 8, pp. 1170–1177, 2002.

[4] R. Govindarajan, J. Duraiyan, K. Kaliyappan, and M. Palanisamy, "Microarray and its applications," *Journal of Pharmacy and Bioallied Sciences*, vol. 4, no. Suppl 2, pp. S310–S312, 2012.

[5] M. Khalsan, L. R. Machado, E. S. Al-Shamery, S. Ajit, K. Anthony, M. Mu, and M. O. Agyeman, "A survey of machine learning approaches applied to gene expression analysis for cancer prediction," *IEEE Access*, vol. 10, pp. 27 522–27 534, 2022.

[6] B. I. Grisci, B. C. Feltes, and M. Dorn, "Neuroevolution as a tool for microarray gene expression pattern identification in cancer research," *Journal of biomedical informatics*, vol. 89, pp. 122–133, 2019.

[7] F. Nelli, "Machine learning with scikit-learn," in *Python data analytics: with pandas, numPy, and matplotlib*. Springer, 2023, pp. 259–287.

[8] M. Ilyas, K. M. Aamir, A. Jaleel, and M. Deriche, "A novel leukemia gene features extraction and selection technique for robust type prediction using machine learning," *Arabian Journal for Science and Engineering*, pp. 1–19, 2024.

[9] B. I. Grisci, M. J. Krause, and M. Dorn, "Relevance aggregation for neural networks interpretability and knowledge discovery on tabular data," *Information sciences*, vol. 559, pp. 111–129, 2021.

[10] S. Osama, H. Shaban, and A. A. Ali, "Gene reduction and machine learning algorithms for cancer classification based on microarray gene expression data: A comprehensive review," *Expert Systems with Applications*, vol. 213, p. 118946, 2023.

[11] J. d. S. Bohrer and M. Dorn, "Enhancing classification with hybrid feature selection: A multi-objective genetic algorithm for high-dimensional data," *Expert Systems with Applications*, p. 124518, 2024.

[12] M. C. Barbieri, B. I. Grisci, and M. Dorn, "Analysis and comparison of feature selection methods towards performance and stability," *Expert Systems with Applications*, p. 123667, 2024.

[13] H. Younis, M. H. Bhatti, and M. Azeem, "Classification of skin cancer dermoscopy images using transfer learning," in *2019 15th International Conference on Emerging Technologies (ICET)*. IEEE, 2019, pp. 1–4.

[14] A. Khan and B. Lee, "Deepgene transformer: Transformer for the gene expression-based classification of cancer subtypes," *Expert Systems with Applications*, vol. 226, p. 120047, 2023.

[15] M. Greenacre, P. J. Groenen, T. Hastie, A. I. d'Enza, A. Markos, and E. Tuzhilina, "Principal component analysis," *Nature Reviews Methods Primers*, vol. 2, no. 1, p. 100, 2022.

[16] B. Ghojogh, M. Crowley, F. Karray, and A. Ghodsi, "Uniform manifold approximation and projection (umap)," in *Elements of Dimensionality Reduction and Manifold Learning*. Springer, 2023, pp. 479–497.

[17] M. Casals and P. Daunis-i Estadella, "Violinboxplot and enhanced radar plot as components of effective graphical dashboards: An educational example of sports analytics," *International Journal of Sports Science & Coaching*, vol. 18, no. 2, pp. 572–583, 2023.

[18] E. Clough and T. Barrett, "The gene expression omnibus database," *Statistical Genomics: Methods and Protocols*, pp. 93–110, 2016.

[19] A. Kauffmann, R. Gentleman, and W. Huber, "arrayqualitymetrics—a bioconductor package for quality assessment of microarray data," *Bioinformatics*, vol. 25, no. 3, pp. 415–416, 2009.