Report

Machine Learning for Natural Sciences

Analysis of Omics Data using OmiEmbed

Contributors:

Syed Imami 2020113012
Urvish Pujara 2020101036

Contents

1 Introduction **3**

2 Problem Statement **3**

3 Data sets **3**

4 Model Implementation **4**
4.1 Baseline Model 4
4.1.1 Process 6
4.1.2 Implementation Output 6
4.2 Baseline Model Results 7
4.3 Implementation of next part of problem statement 7

5 Conclusion **9**

Abstract

"OmiEmbed" is a unified deep learning framework for the integration and analysis of multi-omics data. Multi-omics data, which involves the integration of multiple types of molecular data such as genomics, transcriptomics, proteomics, and metabolomics, has become increasingly important in biological research. OmiEmbed is a multi-task learning framework that can simultaneously predict various molecular traits using different types of multi-omics data. The model is built on a graph convolutional neural network that can learn the complex relationships among different types of molecular data. The authors evaluated the model on multiple datasets and achieved state-of-the-art performance compared to existing methods. Overall, OmiEmbed is a promising tool for the analysis of multi-omics data and has the potential to advance our understanding of complex biological systems/ VAE learns to encode omics data into a lower-dimensional representation. OmiEmbed can be used to learn joint representations of multiple types of biological data. VAEs can help identify new biological insights and biomarkers that are difficult to detect using single-omics.

Keywords: Multi-omics , gene expression, miRNA, DNA Methylation, Y- chromosome, Deep learning, Multi-task learning, Omics analysis

1 Introduction

Multi-omics analysis is an emerging field that aims to integrate multiple types of biological data, such as genomics, transcriptomics, proteomics, and metabolomics, to gain a more comprehensive understanding of biological systems. VAE learns to encode omics data into a lower-dimensional representation and then decodes this representation back into the original input data.VAE can be trained to generate new samples of gene expression data that are consistent with other types of biological data, such as protein expression data. By learning joint representations of multiple types of biological data, VAEs can help to identify new biological insights and biomarkers that would be difficult to detect using single-omics analysis like cancer classification. These techniques will help us understanding the complex interactions between different types of biological data.

2 Problem Statement

OmiEmbed supports multiple tasks for omics data including dimensionality reduction, tumour type classification, multi-omics integration, demographic and clinical feature reconstruction, and survival prediction. This can further be extended to providing feature importance for each type of cancer. Using the age predictions, demographic analysis can be performed such as the analysis frequency of people at various stages of cancer in all types of cance. By incorporating clinical features, the model aims to provide a more comprehensive analysis that takes into account factors that can affect patient outcomes. Overall, the new model aims to improve patient care by providing accurate and personalized predictions of survival outcomes.

3 Data sets

Two publicly available datasets were used as examples to demonstrate the ability of OmiEmbed: the Genomic Data Commons (GDC) pan-cancer multi-omics dataset [17] and the DNA methylation dataset of human central nervous system tumours (GSE109381)

Dataset Info	**GDC**			**BTM**
Domain	Pan-cancer			Brain tumour
Tumour type	33 (TCGA) + 3 (TARGET) + 1 (normal)			86 + 8 (normal)
Additional label	Disease stage, primary site, gender, age, survival			Disease stage, gender, age
Omics type	Gene expression	DNA methylation	miRNA expression	DNA methylation
Feature number	60,483	485,577	1881	485,577
Sample number	11,538	9736	11,020	3905

Figure 1: Dataset

4 Model Implementation

4.1 Baseline Model

Certain probes were removed during the feature filtering step according to the following criteria: probes targeting the Y chromosome (n = 416), probes containing the dbSNP132Common single- nucleotide polymorphism (SNP) (n = 7998), probes not mapping to the human reference genome (hg19) uniquely (one mismatch allowed) (n = 3965), probes not included in the latest Infinium MethylationEPIC BeadChip (EPIC) array (n = 32,260), the SNP assay probes (n = 65), the non-CpG loci probes (n = 3091) and probes with missing values (N/A) in more than 10% of samples (n = 2). We followed some of the criteria mentioned in the original paper of this dataset [5]. Overall, 46,746 probes were filtered out, which results in a final DNA methylation feature set of 438,831 CpG sites.

Feature filtering was applied to the gene expression data: targeting Y chromosome (n = 594) and zero expression in all samples (n = 1904). In total, 2440 genes were removed, leaving 58,043 molecular features for further analyses. All of the miRNA identifiers were kept in our experiments. For both the gene expression and miRNA expression profiles, the expression values were normalised to the range of 0 to 1 due to the input requirement of the OmiEmbed framework.

A multi-layer fully connected network was applied to classification-type downstream tasks, including diagnostic tasks such as tumour type classification, primary site prediction and disease stage (i.e., primary tumour, recurrent tumour, metastatic tumour or normal control) prediction and demographic tasks, e.g., the prediction of gender.

$$\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \log p_\theta(\mathbf{x}|\mathbf{z}) - D_{\mathrm{KL}}\big(q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})\big)$$

The loss function can be given as the following :

$$\mathcal{L}_{embed} = \frac{1}{M}\sum_{j=1}^{M} BCE\left(\mathbf{x}_j, \mathbf{x}'_j\right) + D_{\mathrm{KL}}(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}) \| \mathcal{N}(\mathbf{0}, \mathbf{I}))$$

Figure 2: loss function

Classification loss is given by :

$$\mathcal{L}_{classification} = CE(y, y')$$

Regression Loss :

$$\mathcal{L}_{regression} = MSE(y, y')$$

Total Loss :

$$\mathcal{L}_{total} = \lambda \mathcal{L}_{embed} + \mathcal{L}_{down}$$

4.1.1 Process

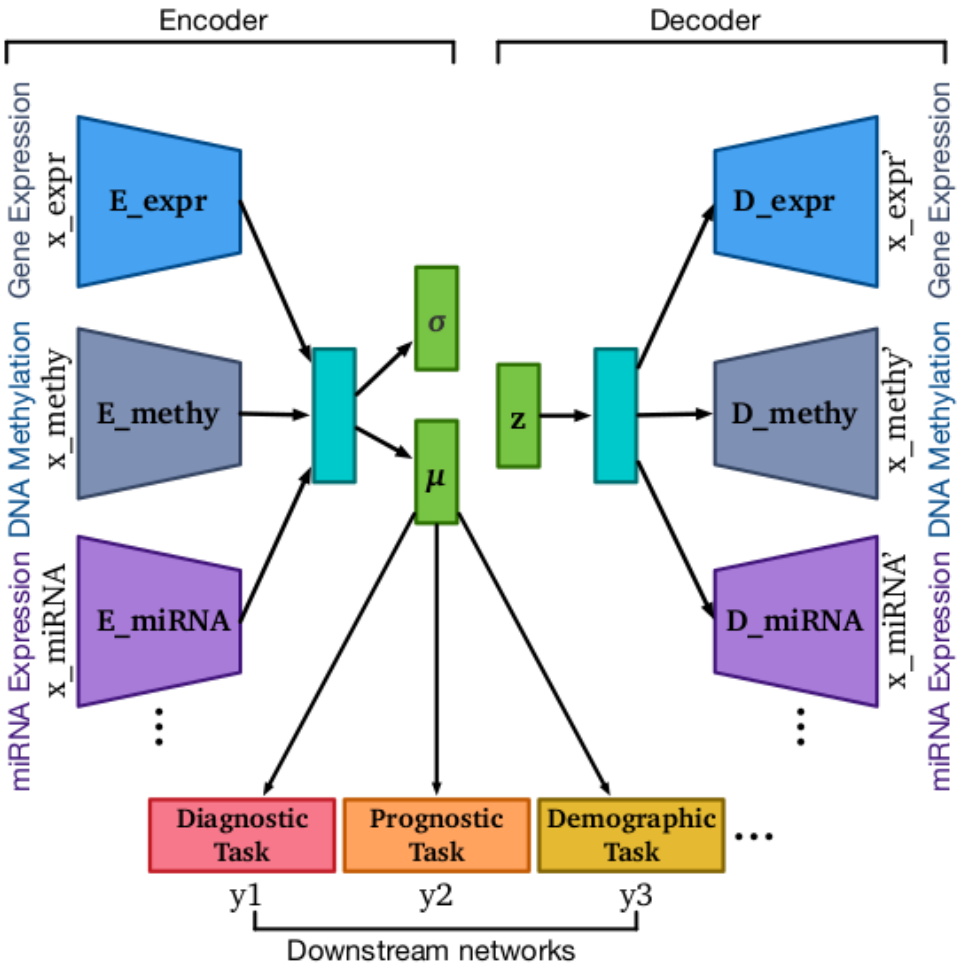
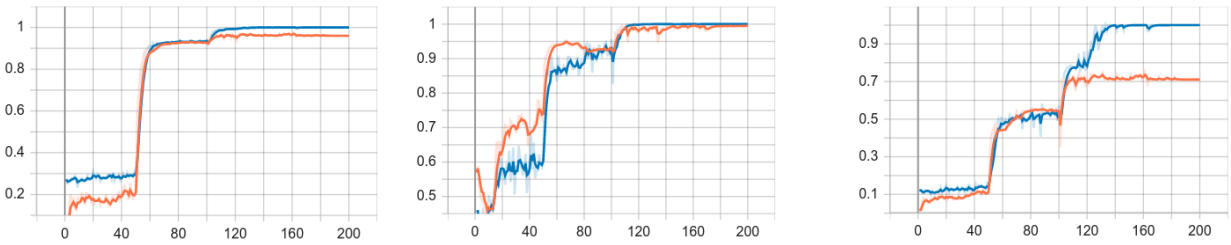


Figure 3: Model

4.1.2 Implementation Output

TSV are obtained as a result of the training and testing in which 5 metrics are measured to ensure the classification is done properly

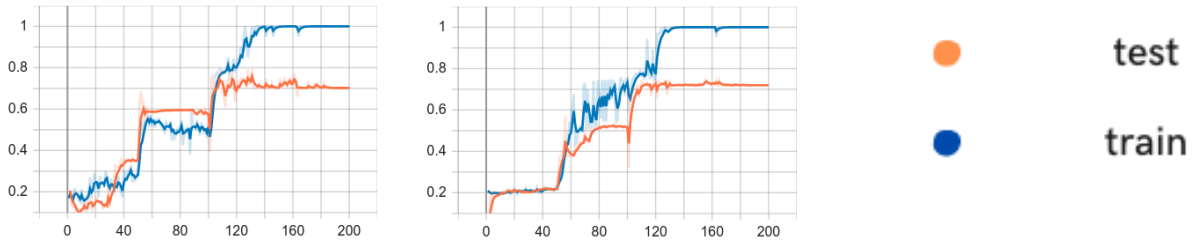


(a) Accuracy Metric

(b) AUC metric

(c) F1 metric

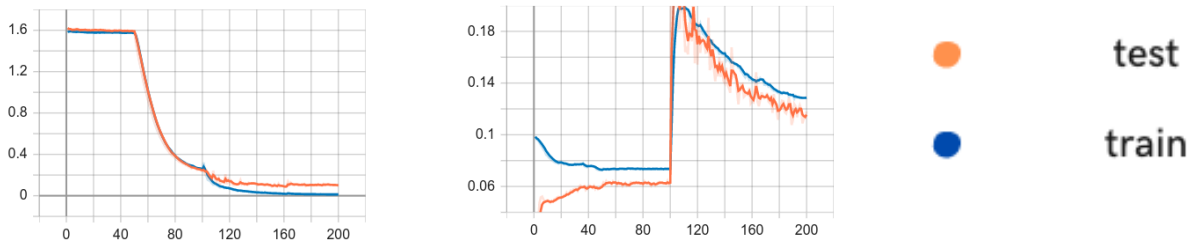
Figure 4: Metrics Calculation for the Classification



(a) Recall Metric **(b)** Precision metric **(c)** Labelling

Figure 5: Metrics Calculation for the Classification

Loss Calculations are also being done for the process and the results of them are attached below



(a) Classifier Loss **(b)** KL loss **(c)** Labelling

Figure 6: Loss

4.2 Baseline Model Results

Table 1: Metrics Values for both training and testing

S.no	Measurement	Training	Testing
1	Accuracy	1	0.96
2	AUC	1	0.994
3	F1	1	0.7103
4	Precision	1	0.7195
5	Recall	1	0.7022
6	Classifier Loss	0.01282	0.09333
7	KL loss	0.1287	0.1181

4.3 Implementation of next part of problem statement

OmiEmbed supports multiple tasks for omics data including dimensionality reduction, tumor type classification, multi-omics integration, demographic and clinical feature reconstruction, and survival prediction. This can further be extended to age predictions, demographic analysis such as frequency of people at various stages of cancer in all types of cancer and estimating the feature importance. By incorporating clinical features, we can aim on using only

important features for training and testing further. The model aims to provide a more comprehensive analysis that takes into account factors that can affect patient outcomes.

One way to indirectly estimate feature importance from a trained VAE is to analyze the learned latent representation. Since the VAE learns a lower-dimensional representation of the input data in the latent space, the dimensions or components of the latent representation that have higher variance or larger magnitude can indicate more important features in the input data. This can be interpreted as the VAE implicitly learning a form of dimensionality reduction, where the more important features are retained in the learned latent representation, and the less important features are discarded or compressed. This can be done by training the VAE using a dataset of input data with known features. The VAE should be trained to accurately reconstruct the input data during training, and the encoder and decoder parameters should be optimized to minimize the reconstruction error. Extract the learned latent representation: Once the VAE is trained, encode the input data using the trained encoder to obtain the learned latent representation for each data point. The learned latent representation typically consists of a mean and standard deviation for each dimension or component of the latent space. Analyze the variance or magnitude of the latent dimensions. Rank the dimensions or components of the latent representation based on their computed variance or magnitude. After extracting the learned latent representation, we fit linear regression model with latent representation as input and original feature as output. The slope of linear regression represents the feature importance. The dimensions or components of the latent representation with higher variance or larger magnitude may correspond to more important features in the input data, but further analysis and interpretation may be needed to draw meaningful conclusions about feature importance.

Another way to potentially estimate feature importance from a trained VAE is to analyze the reconstruction quality of the input data. If the VAE is trained to accurately reconstruct the input data during training, then the reconstruction error, which is the difference between the input data and its reconstructed data, can provide an indication of the importance of different features. We can train a VAE on the input data, and then reconstruct the input data using the trained VAE. For each feature, we can calculate the reconstruction error, which represents how well the VAE is able to reconstruct that feature from its latent representation. A higher reconstruction error for a feature indicates that the VAE struggles to accurately reconstruct that feature, and hence, that feature may be more important. On the other hand, a lower reconstruction error for a feature indicates that the VAE is able to accurately reconstruct that feature, and hence, that feature may be less important. You can sort the features based on their reconstruction error values in descending order, where higher reconstruction error values represent more important features, and lower reconstruction error values represent less important features. This way, you can determine the relative importance of features in your dataset using the reconstruction error method with VAEs. However, it's important to note that the reconstruction error method may have limitations and assumptions, and it's always a good practice to validate the results with other feature importance techniques and domain knowledge. Features that are important for accurately reconstructing the input data are likely to have a larger impact on the reconstruction error, while less important features may have a smaller impact.

5 Conclusion

The goal of the project is to develop a machine learning model that can effectively train on a large dataset with reduced features, resulting in faster model training times without significant loss of performance. The project aims to explore and implement various feature reduction techniques or a combination of these techniques, to identify and retain the most important features while discarding redundant or irrelevant features from the dataset. The project will involve evaluating the impact of feature reduction on the training time of the model, along with monitoring the model's performance in terms of accuracy, precision, recall, and other relevant metrics, to ensure that the reduction in features does not result in a significant degradation of the model's predictive performance. The ultimate goal of the project is to develop a more efficient and scalable machine learning model that can be trained faster on large datasets, thereby improving the productivity and performance of the overall machine learning pipeline.

References

1. OmiEmbed: A Unified Multi-Task Deep Learning Framework for Multi-Omics Data
2. Performance Comparison of Deep Learning Autoencoders for Cancer Subtype Detection Using Multi-Omics Data
3. Multi-omics Data Integration, Interpretation, and Its Application