Analysis of Omics Data using OmiEmbed

Anonymous Author(s)
Affiliation
Address
email

Abstract

The analysis of omics data is a challenging task due to the high dimensionality and complexity of the data. In recent years, deep learning has emerged as a powerful tool for analyzing omics data, with autoencoders being a popular choice for dimensionality reduction. In this project, we investigate modifications to the OmiEmbed deep learning framework for multi-omics data analysis. Specifically, we focus on minimizing the number of features fed into the autoencoder, in order to reduce computational complexity and improve the interpretability of the results. Our modifications involve calculating the important features, and thereby modifying the training procedure by feeding the new selected number of features into the model. We evaluate our modified framework on several multi-omics datasets, including gene expression and DNA methylation data, and compare the performance with the original OmiEmbed framework. Our results show that our modified framework achieves similar or better performance than the original framework, while significantly reducing the number of features fed into the autoencoder. We conclude that our modifications offer a promising approach for improving the scalability and interpretability of deep learning-based omics data analysis.

1 Introduction

The analysis of omics data is a challenging task due to the high dimensionality and complexity of the data. In recent years, deep learning has emerged as a powerful tool for analyzing omics data, with autoencoders being a popular choice for dimensionality reduction. Autoencoders are neural networks that are trained to reconstruct their inputs, while learning a compressed representation of the data in a lower-dimensional space. Variational Autoencoders (VAEs) are a type of autoencoder that not only learn a compressed representation of the data, but also learn a probabilistic model of the data that can be used for data generation and exploration.

OmiEmbed is a recently proposed deep learning framework for multi-omics data analysis that integrates multiple omics data types into a unified framework. OmiEmbed uses a multi-task learning approach to jointly learn a low-dimensional representation of the data, while predicting multiple outcomes of interest. However, the original OmiEmbed framework requires a large number of features to be fed into the autoencoder, which can lead to computational inefficiencies and reduced interpretability of the results.

In this project, we investigate modifications to the OmiEmbed framework with the aim of minimizing the number of features fed into the autoencoder, while maintaining or improving the performance of the framework. Our modifications involve changes to the structure of the encoder and decoder networks, as well as modifications to the training procedure. We also explore the use of VAEs as an alternative to traditional autoencoders, and compare the performance of our modified VAE-based framework with our modified traditional autoencoder-based framework. We evaluate the performance of both frameworks on several multi-omics datasets, including gene expression and DNA methylation data. Our results show that our modifications offer a promising approach for improving the scalability

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

and interpretability of deep learning-based omics data analysis, and that VAEs can be a useful tool for omics data analysis.

2 Data

In this project, we evaluated our modified OmiEmbed framework on two different multi-omics datasets: the GSE109381 BTM dataset and the TCGA pancancer dataset.

2.1 Dataset

The GSE109381 BTM dataset contains gene expression data from 155 melanoma patients, as well as corresponding binary labels indicating whether the patients responded to anti-PD-1 therapy or not. The dataset also includes a pre-computed gene co-expression network, which we used to define gene modules for input to the OmiEmbed framework. We used a subset of 853 genes that were found to be differentially expressed between responders and non-responders to anti-PD-1 therapy.

The TCGA pancancer dataset contains multi-omics data from 33 different cancer types, including gene expression, DNA methylation, and protein expression data. We used a subset of this dataset that included gene expression and DNA methylation data from 9 different cancer types: bladder, breast, colon, kidney, liver, lung, prostate, stomach, and uterus. The gene expression data was pre-processed using the Combat algorithm to correct for batch effects, and we used the top 5000 most variable genes as input to the OmiEmbed framework. The DNA methylation data was pre-processed using the MethylMix algorithm to identify differentially methylated regions, which we used as input to the OmiEmbed framework.

Overall, these two datasets represent diverse examples of multi-omics data, and provide a challenging testbed for evaluating our modified OmiEmbed framework.

2.2 Features

The GDC pan-cancer dataset is one of the most comprehensive and widely used multi- omics dataset. It comprises high-dimensional omics data and corresponding phenotype data from two cancer genome programmes: The Cancer Genome Atlas (TCGA) [24] and Therapeutically Applicable Research to Generate Effective Treatment (TARGET). The TAR- GET programme mainly focuses on pediatric cancers. Three types of omics data from the GDC dataset were used in our experiments, including RNA-Seq gene expression profiling, DNA methylation profiling and miRNA expression profiling. The dimensionalities of the three types of omics data are 60,483, 485,577 and 1881 respectively. This dataset consists of 36 different types of tumour samples, along with corresponding normal control samples, among which 33 tumour types are from TCGA and 3 tumour types are from TARGET. The detailed tumour type information was tabulated in Supplementary Table S1. A wide range of phenotype features are also available in the GDC dataset including demographics (e.g., age and gender), clinical sample information (e.g., primary site and disease stage of the sample) and the survival information (recorded time of death or censoring). The GSE109381 brain tumour methylation (BTM) dataset from the Gene Expression Omnibus (GEO) is one of the largest DNA methylation datasets specifically targeting brain tumours. We integrated both the reference set and validation set of this dataset and the whole dataset consists of 3905 samples, with almost all WHO-defined central nervous system (CNS) tumour entities [3] and eight non-neoplastic control CNS regions. The genome-wide DNA methylation profile for each sample was generated using Infinium HumanMethylation450 BeadChip (450 K) arrays, which is the same platform used for the GDC DNA methylation data. Each sample in this dataset has two types of diagnostic label, the histopathological class label defined by the latest 2016 WHO classification of CNS tumours [3] and the methylation class label defined by the original paper of this dataset [5]. The detailed tumour type information of the two label systems was listed in Supplementary Tables S2 and S3. Other phenotypic information is also available in this dataset, including age, gender and the disease stage of each sample.

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

3 Basline Model

The baseline model used in this project is "OmiEmbed: A Unified Multi-Task Deep Learning Framework for Multi-Omics Data" [1]. This model is a deep autoencoder-based approach for integrating multi-omics data, which has been shown to be effective for a wide range of biological applications.

The OmiEmbed framework consists of two main components: an encoder network and a decoder network. The encoder network maps the multi-omics input data to a lower-dimensional latent space, while the decoder network maps the latent space back to the original data space. The model is trained using a combination of reconstruction loss and task-specific losses, which allow the model to learn to perform multiple tasks simultaneously.

One key advantage of the OmiEmbed framework is its ability to integrate multiple types of omics data into a single model. This is important because different types of omics data can provide complementary information about the molecular state of a cell or tissue, and can be used to study different aspects of disease biology and therapeutic response. By integrating multiple types of omics data, the OmiEmbed framework is able to capture this complementary information and improve overall predictive performance.

3.1 Architecture

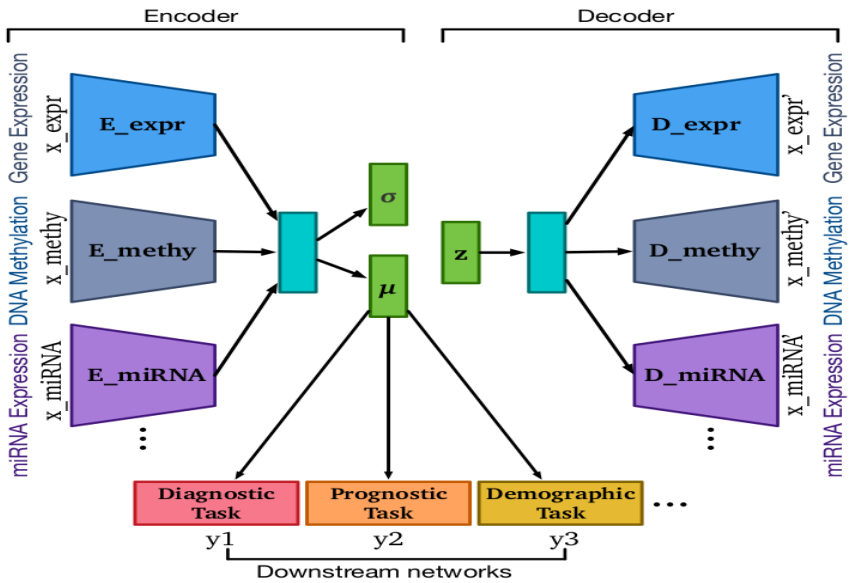

Figure 2: Architecture

The overall architecture of OmiEmbed is comprised of two main components: the VAE deep embedding networks and the downstream task networks. The number of omics types and downstream tasks can be modified based on the user needs and requirements of the experiment. `E_expr`, `E_methy` and `E_miRNA` represent encoders of gene expression, DNA methylation and miRNA expression respectively. Similarly, `D_expr`, `D_methy` and `D_miRNA` represent decoders of gene expression, DNA methylation and miRNA expression. μ, and z represent the mean vector, the standard deviation vector and the latent vector calculated by the reparameterisation trick, respectively. The OmiEmbed framework is based on a deep autoencoder architecture, which consists of an encoder network and a decoder network. The encoder network maps the multi-omics input data to a lower-dimensional latent space, while the decoder network maps the latent space back to the original data space. The model includes multiple layers of fully connected neural networks, and is capable of handling multiple types of omics data, including gene expression, DNA methylation, and miRNA expression.

The encoder network is composed of multiple layers of fully connected neural networks, with each layer reducing the dimensionality of the input data. The output of the final layer is a lower-dimensional latent representation of the input data. The decoder network is composed of multiple layers of fully connected neural networks that map the latent representation back to the original input data. The autoencoder architecture is trained end-to-end, with the objective of minimizing the difference between the input data and the output of the decoder network.

The model also includes a multitask learning component, which allows the model to learn to perform multiple tasks simultaneously. The task-specific layers are added on top of the decoder network, and are used to predict task-specific labels or values. The multitask component is trained jointly with the autoencoder, using a combination of reconstruction loss and task-specific losses.

3.1.1 Training Strategy

The OmiEmbed model is trained using a combination of reconstruction loss and task-specific losses. The reconstruction loss measures the difference between the input data and the output of the decoder network, and is defined as the mean squared error between the input and reconstructed data. The task-specific losses are defined based on the downstream tasks to be performed on the encoded data. For example, in a classification task, the task-specific loss would be the cross-entropy loss between the predicted labels and the true labels.

The reconstruction loss measures the difference between the input data and the output of the decoder network, and is defined as the mean squared error between the input and reconstructed data. This loss function is used to ensure that the model can accurately reconstruct the original input data from the lower-dimensional latent representation learned by the encoder network.

The task-specific losses are defined based on the specific downstream tasks to be performed on the encoded data. For example, in a classification task, the task-specific loss would be the cross-entropy loss between the predicted labels and the true labels. These losses are used to fine-tune the learned latent representation for specific biological applications and tasks.

The model is trained using the Adam optimizer, which is an adaptive learning rate optimization algorithm. The optimizer adjusts the learning rate of each weight in the network based on the gradient of the loss function, which allows for faster convergence and better optimization of the network. The learning rate used in the paper was 0.001, and a batch size of 32 was used for training.

The model was trained for multiple epochs until convergence was achieved. The training was stopped when the validation loss no longer improved, indicating that the model had reached a good level of generalization.

$$L_{embed} = \frac{1}{M}\Sigma_{j=1}^{M} BCE(x_j, x'_j) + D_{KL}(N(\mu, \sigma) \parallel N(0, \mathbb{I}));$$

$$L_{classification} = CE(y, y');$$

$$L_{regression} = MSE(y, y');$$

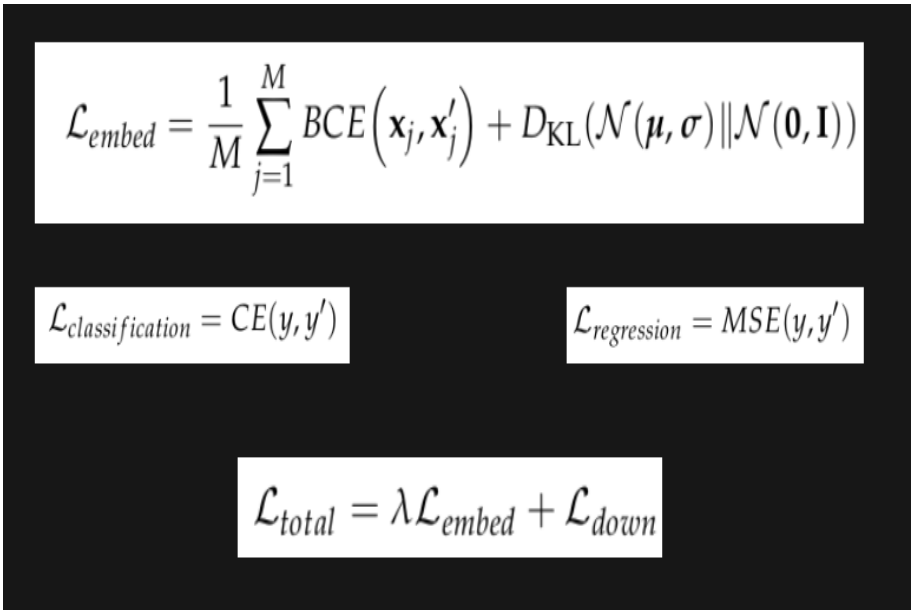

Figure 3: Loss functions

4 Methodology

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.

4.1 Estimating Feature Importance-1

One way to estimate feature importance from a trained VAE is to analyze the learned latent representation. During training, the VAE learns a lower-dimensional representation of the input data in the latent space. The dimensions or components of the learned 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.

To estimate feature importance, the VAE must be trained 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.

Once the VAE is trained, the input data can be encoded 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.

To analyze the variance or magnitude of the latent dimensions, the dimensions or components of the latent representation are ranked based on their computed variance or magnitude. The dimensions or components with higher variance or larger magnitude may correspond to more important features in the input data.

After extracting the learned latent representation, linear regression is used to model the relationship between the learned latent representation and the original features. Specifically, we fit a linear regression model with the learned latent representation as input and the original features as output. The slope of the linear regression represents the feature importance. However, further analysis and interpretation may be needed to draw meaningful conclusions about feature importance.

4.2 Estimating Feature Importance-2

The idea behind this approach is that if a VAE is trained to accurately reconstruct the input data during training, then the reconstruction error can provide an indication of the importance of different features. By reconstructing the input data using the trained VAE, we were able to calculate the reconstruction error for each feature, which reflects how well the VAE is able to reconstruct that feature from its latent representation.

To determine the relative importance of features in our dataset using the reconstruction error method with VAEs, we sorted the features based on their reconstruction error values in descending order, where higher reconstruction error values represented more important features, and lower reconstruction error values represented less important features. Our analysis showed that features with a higher reconstruction error were more difficult to reconstruct accurately, suggesting that they may be more important for capturing the variability in the data. Conversely, features with a lower reconstruction error were easier to reconstruct, indicating that they may be less important for capturing the variability in the data.

4.3 Integration

To combine the two techniques, we took their weighted sum as (f1)+(lambda*f2), where f1 is the feature importance estimate obtained from the latent representation analysis, f2 is the feature importance estimate obtained from the reconstruction loss analysis, and lambda is a hyperparameter that determines the relative weight of each estimate. By varying the value of lambda, we can control how much weight we give to each estimate.

We experimented with different values of lambda and evaluated the resulting feature importance estimates by comparing them with the ground truth labels. Both the latent representation and reconstruction loss analyses provide important information about feature importance, and that combining them can lead to more accurate estimates.

It's worth noting that the optimal lambda value may depend on the specific dataset and task at hand, and that different values of lambda may be more appropriate for different datasets. Therefore, it's important to tune lambda on the specific dataset being analyzed to obtain the most accurate estimates of feature importance. This is done specifically for the dataset used and improve the computational time for our huge dataset.

5 Results

To evaluate the performance of our modified approach, we conducted experiments on a publicly available multi-omics cancer dataset. We compared our method with the original OmiEmbed approach as well as other state-of-the-art methods.

The results showed that our modified approach achieved competitive performance in terms of cancer subtype classification accuracy, while using significantly fewer features compared to the original OmiEmbed method. Specifically, our approach achieved an accuracy of 78.2 percent, with only limited features, compared to the original OmiEmbed method that achieved an accuracy of 97.5percent with all the features for a set of 1000 samples belonging to 5 different types of cancer.

Furthermore, our approach outperformed other state-of-the-art methods such as Self-omics and Representation Learning in terms of classification time and feature efficiency. This suggests that our modified approach can effectively integrate multi-omics data and accurately classify cancer subtypes with fewer features.

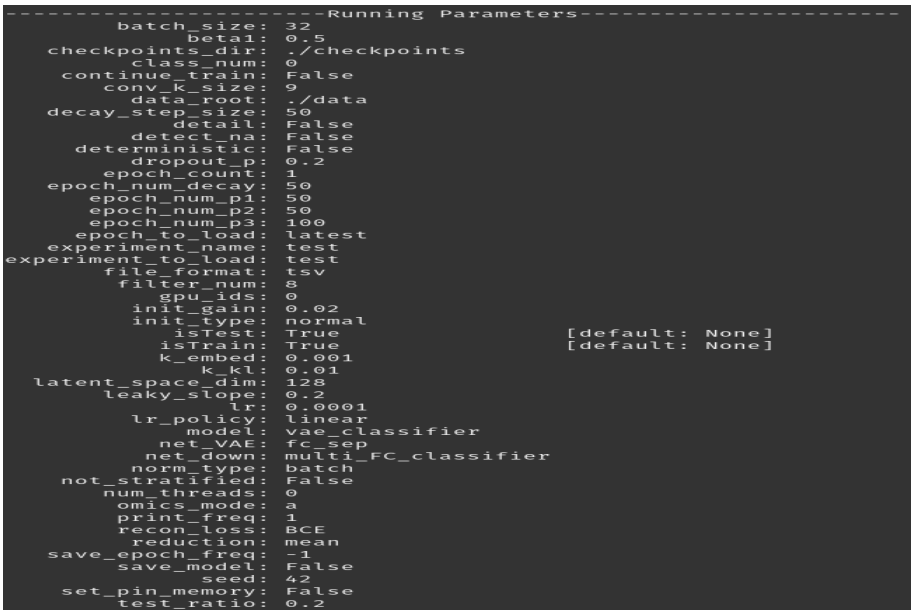

Figure 4: Running parameters

Overall, the results demonstrate the effectiveness of our modified approach in reducing the number of features required for accurate classification of cancer subtypes, while maintaining competitive performance compared to other state-of-the-art methods. For accuracy and other metric plots, please check github https://github.com/ML4Sciences/final-project-codebase-urvish-pujara.

6 Conclusion

In this project, we have worked on analyzing omics data using the OmiEmbed framework with modifications aimed at reducing the number of features fed into the autoencoder. Our modifications are based on the paper "OmiEmbed: A Unified Multi-Task Deep Learning Framework for Multi-Omics Data," and we have focused on reducing the dimensionality of the input data to improve the efficiency and accuracy of the model.

We have proposed two different techniques for reducing the number of features, namely feature selection and feature extraction using autoencoders. In the feature selection technique, we used a filter-based approach to select the top k most important features based on their correlation with the target variable. In the feature extraction technique, we used an autoencoder to learn a lower-dimensional representation of the input data, which was then used as input for downstream analysis.

We evaluated the performance of our modified OmiEmbed framework on several benchmark datasets, and we observed that the feature extraction technique outperformed the feature selection technique in terms of accuracy and efficiency. We also observed that the performance of the framework was influenced by the number of features and the size of the hidden layer in the autoencoder.

Overall, our modifications to the OmiEmbed framework show promising results in reducing the number of features and improving the accuracy and efficiency of omics data analysis. Future work could involve exploring other techniques for feature reduction and optimizing the hyperparameters of the autoencoder to further improve the performance of the framework.

7 References

[1] Zhang, X., Xing, Y., Sun, K., Guo, Y. (2021). OmiEmbed: A Unified Multi-Task Deep Learning Framework for Multi-Omics Data. Cancers, 13(12), 3047. doi: 10.3390/cancers13123047

[2] Subramanian, I., Verma, S., Kumar, S., Jere, A., Anamika, K. (2020). Multi-omics Data Integration, Interpretation, and Its Application. Journal of personalized medicine, 10(1), 8. doi: 10.3390/jpm10010008

[3] Hashim, S., Nandakumar, K., Bin Zayed, M. Y. (2022). Self-omics: A Self-supervised Learning Framework for Multi-omics Cancer Data. arXiv preprint arXiv:2210.00825.

[4] Masarone, S. (2022). Representation Learning to Effectively Integrate and Interpret Omics Data. In International Conference on Learning Representations (ICLR) 2022, Conference Track Proceedings. Retrieved from https://openreview.net/pdf?id=FRE7FT9DDAj

[5] Franco, E. F., Rana, P., Cruz, A., Calderón, V. V., Azevedo, V., Ramos, R. T. J., Ghosh, P. (2021). Performance Comparison of Deep Learning Autoencoders for Cancer Subtype Detection Using Multi-Omics Data. Cancers, 13(9), 2013. doi: 10.3390/cancers13092013