Deep Learning-Based Classification and Biomarker Discovery for Neurodegenerative Disorders Using Multi-Omics Data

SAUMYA
saumyaa.2310@gmail.com

April 20, 2025

Abstract

Neurodegenerative disorders are a group of incurable, debilitating conditions involving progressive neuronal degeneration. This project uses deep learning models to classify the main disorders, Alzheimer's, Parkinson's, and ALS, integrating transcriptomic and methylation data. An autoencoder was used for dimensionality reduction, followed by a neural network classifier. The project demonstrates how multi-omics and AI can assist in early diagnosis and biomarker discovery.

1 Introduction

Neurodegenerative diseases (NDs) are complex and heterogeneous, necessitating computational methods that can extract insights from large-scale omics data. Deep learning offers the ability to learn intricate, non-linear representations and has shown promise in biomedical applications.

2 Objectives

- To preprocess and integrate gene expression and methylation data.
- Apply deep learning for disease classification.
- To identify molecular features indicative of disease.

3 Materials and Methods

3.1 Data Acquisition

Datasets were obtained from public repositories like GEO and ADNI. RNA-Seq and methylation data sets were merged and filtered on the basis of quality and biological relevance.

3.2 Preprocessing and Feature Engineering

Normalization, scaling, label encoding, and splitting of trainswere applied using Python's `pandas`, `sklearn`, and `tensorflow` libraries.

3.3 Model Architecture

An autoencoder was used to denoise and compress high-dimensional omics data, followed by a dense neural network classifier.

3.4 Code Snippet

Listing 1: Deep Learning Pipeline for Classification

```
import pandas as pd
import numpy as np
import tensorflow as tf
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.metrics import classification_report, confusion_matrix
from keras.models import Model
from keras.layers import Input, Dense, Dropout
from keras.optimizers import Adam

# Load and merge datasets
gene_expr = pd.read_csv('gene_expression_data.csv', index_col=0)
methylation = pd.read_csv('methylation_data.csv', index_col=0)
labels = pd.read_csv('labels.csv', index_col=0)
merged_data = gene_expr.join(methylation)

# Preprocess
scaler = StandardScaler()
X = scaler.fit_transform(merged_data)
label_encoder = LabelEncoder()
y = label_encoder.fit_transform(labels['diagnosis'])

X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y)
y_train_ohe = tf.keras.utils.to_categorical(y_train)
y_test_ohe = tf.keras.utils.to_categorical(y_test)

# Autoencoder
input_layer = Input(shape=(X_train.shape[1],))
encoded = Dense(512, activation='relu')(input_layer)
encoded = Dropout(0.3)(encoded)
encoded = Dense(128, activation='relu')(encoded)
decoded = Dense(512, activation='relu')(encoded)
decoded = Dropout(0.3)(decoded)
decoded = Dense(X_train.shape[1], activation='linear')(decoded)

autoencoder = Model(inputs=input_layer, outputs=decoded)
autoencoder.compile(optimizer='adam', loss='mse')
autoencoder.fit(X_train, X_train, epochs=50)

# Classifier
encoder = Model(inputs=input_layer, outputs=encoded)
X_train_encoded = encoder.predict(X_train)
```

```
X_test_encoded = encoder.predict(X_test)

clf_input = Input(shape=(X_train_encoded.shape[1],))
x = Dense(64, activation='relu')(clf_input)
x = Dropout(0.4)(x)
x = Dense(32, activation='relu')(x)
x = Dense(y_train_ohe.shape[1], activation='softmax')(x)

classifier = Model(inputs=clf_input, outputs=x)
classifier.compile(optimizer=Adam(1e-3), loss='categorical_crossentropy
    ', metrics=['accuracy'])
classifier.fit(X_train_encoded, y_train_ohe, epochs=50)

# Evaluation
y_pred = classifier.predict(X_test_encoded)
y_pred_classes = np.argmax(y_pred, axis=1)
y_true_classes = np.argmax(y_test_ohe, axis=1)
print(classification_report(y_true_classes, y_pred_classes,
    target_names=label_encoder.classes_))
```

4 Results and Discussion

The model achieved promising accuracy across disease classes. Dimensionality reduction via autoencoders helped manage noise and overfitting. SHAP analysis (future extension) will help in biomarker interpretation.

5 Conclusion

Deep learning combined with omics integration is effective in classifying complex diseases like NDs. The pipeline is extensible for larger datasets and additional omics layers.

Future Work

- Apply SHAP or LIME for interpretability.
- Integrate imaging or proteomics data.
- Deploy the model via a web interface.

References

- LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015.
- Zhang B et al. Integration of multi-omics data for classification of neurodegenerative diseases. Brief Bioinform. 2020.