SECB2103-01

PROGRAMMING FOR BIOINFORMATICS 1

PROJECT PROGRESS 4

Title : Comparison of Deep Learning Algorithms on Drug-drug Interaction Prediction

GROUP MEMBERS :

1. AINA NAJWA BINTI MOHD ROZI (A21EC0010)
2. SOFIA ANAK HENRY (A21EC0133)

SECTION : 01

LECTURER'S NAME : DR NIES HUI WEN

1.0 INTRODUCTION

In this research, we focus on our domain which is drug-drug interaction (DDIs) prediction. Drug-drug interactions (DDIs) occur when two or more drugs are taken simultaneously or successively. Drug-drug interactions (DDIs) prediction is the process of assessing and predicting the potential interactions between two or more drugs when taken concurrently. Drug-Drug Interaction (DDI) prediction is one of the most critical issues in drug development and health (Rohani & Eslahchi, 2019).

1.1 PROBLEM BACKGROUND

When two or more medications interact in ways that can harm patients or limit the effectiveness of therapies, this is referred to as a drug-drug interaction (DDIs). These interactions endanger both patient safety and treatment outcomes (Zheng et al., 2018). Many hospital admissions of elderly patients for drug toxicity occur after administration of a drug known to cause drug-drug interactions (Juurlink, 2003). Elderly patients and those with multiple medical conditions are often prescribed numerous medications, making them more susceptible to DDIs. The correct prediction of DDIs has become critical in healthcare due to the increasing complexity of modern treatment regimens (Palleria et al., 2013). The complexity of DDIs data, the requirement for advanced machine learning models, scalability in healthcare. Real-world data from clinical trials and observational studies involving human subjects can provide insights into potential DDIs and their effects in clinical settings. However, these studies are time-consuming, resource-intensive, and may not always be conclusive due to various confounding factors (Juurlink et al., 2003). Addressing these issues through deep learning algorithm comparison is crucial for enhancing patient safety and healthcare decision-making (Aldoseri et al., 2023).

1.2 PROBLEM STATEMENT

The critical challenge under consideration centres on the imperative to enhance the precision and efficiency of drug-drug interaction (DDI) prediction, with a primary focus on elevating patient safety within the healthcare domain. It necessitates an exhaustive evaluation and comparative analysis of deep learning algorithms designed for DDI prediction, aimed at identifying the most efficacious methodologies for practical implementation within healthcare settings. Of paramount concern is the potential harm to patients and the constraints imposed on treatment outcomes resulting from DDIs, highlighting the pressing need for the advancement of DDI prediction algorithms. This research endeavours to provide a pathway towards the development and selection of optimal deep learning models, ultimately bolstering patient safety and the quality of healthcare delivery.

1.3 OBJECTIVE

i To identify the best deep learning algorithms for predicting drug-drug interactions (DDIs).

ii To evaluate the performance of selected methods in predicting drug-drug interactions by assessing their accuracy in comparison.

1.4 SCOPES

This research is focused on drug-drug interaction prediction using different types of algorithms in deep learning. The algorithms of deep learning that we will be comparing in this research are Deep Neural Network (DANN) and Neural Network (NN). The Programming Language that will be used is Python. The dataset about DDIs prediction using DANN and NN will be obtained from Drugbank, Github, and any internet research that is related to our topic.

1.5 CONCLUSION

In this study, we decided to do a comparison between deep learning algorithms such as Deep Neural Network (DNN) and Attention Neural Network to determine which deep learning algorithm is the more effective method for drug-drug interaction prediction.

2.0 SPECIFIC REQUIREMENT

Hardware Requirement :

A computer with :

CPU : Modern Multi-core CPU sufficient for training models and experimenting with deep learning concepts.

Processor : Intel Core i3

8GB of RAM : Sufficient to handle large dataset

Storage: A solid state drive (SSD)

Internet Connection: A stable and reasonably fast internet connection

Backup and Version Control: Git for tracking changes to our code.

Software Requirement :

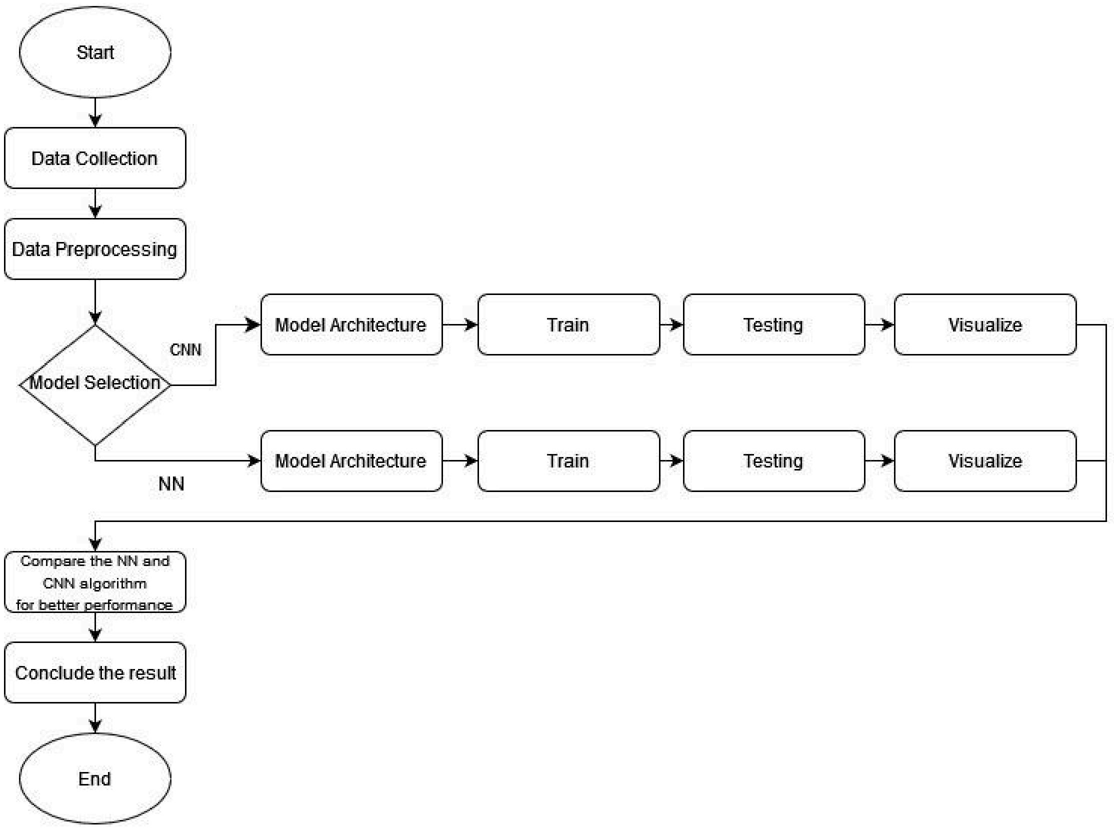
Deep Learning Framework : TensorFlow

Programming language : Python

IDE : Visual Studio Code

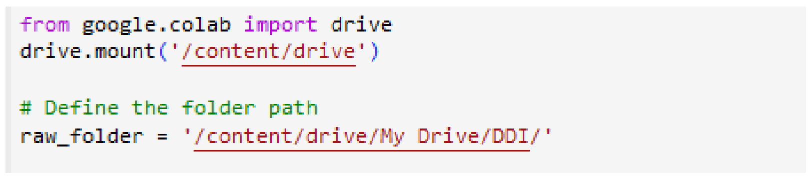
Operating System : Windows 11

3.0 FLOWCHART OF THE PROPOSED APPROACH



4.0 Dataset

4.1 Importing the Dataset

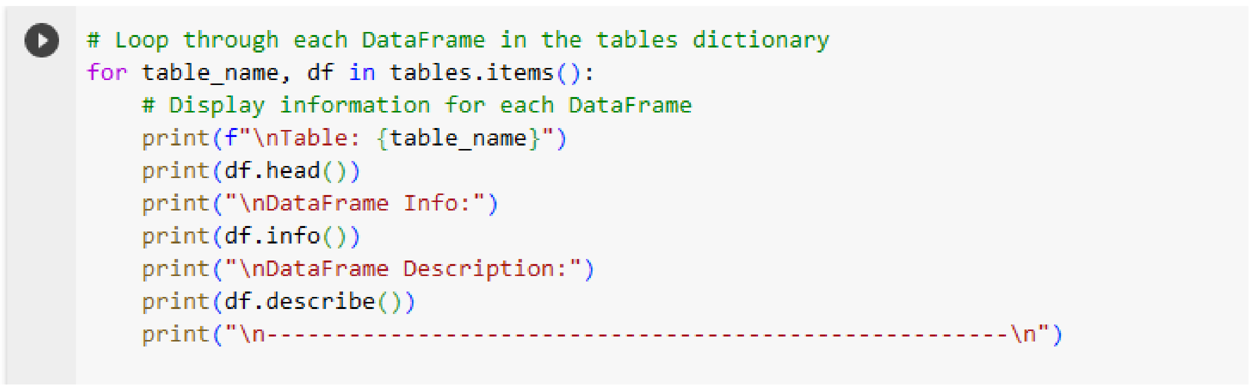


```
from google.colab import drive
drive.mount('/content/drive')

# Define the folder path
raw_folder = '/content/drive/My Drive/DDI/'
```

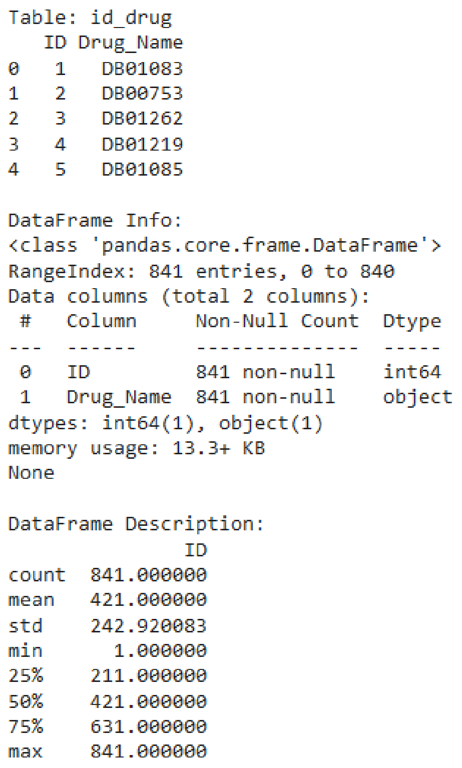
The initial line imports the `drive` module from the `google.colab` package, providing functionality for interaction with Google Drive in the Colab environment. The `drive.mount()` function is then utilized to mount Google Drive to the Colab notebook. As this cell executes, users are prompted to open a link, generate an authentication code, and subsequently paste it back into the cell. Upon the completion of this process, Google Drive becomes mounted, granting access to its contents.

The subsequent line establishes the variable `raw_folder` with the path '/content/drive/My Drive/DDI/'. This line specifies the path to a particular folder in Google Drive, assumed to contain data pertinent to Drug-Drug Interaction (DDI).

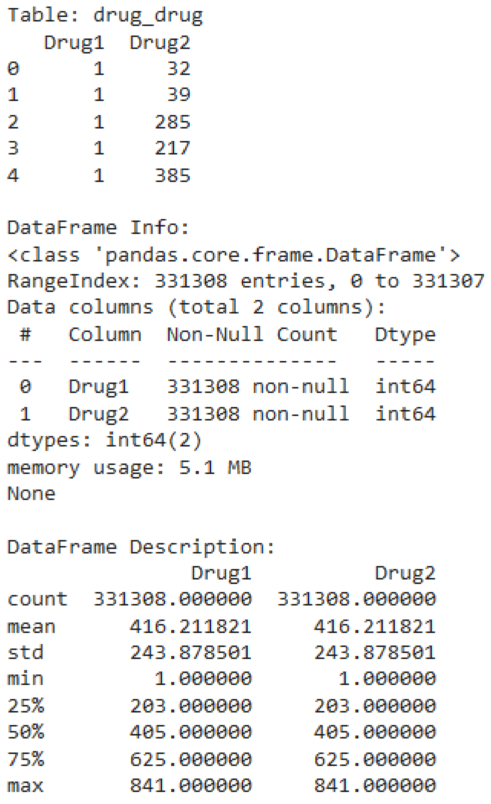
4.1.1 Understanding the data



```
# Loop through each DataFrame in the tables dictionary
for table_name, df in tables.items():
    # Display information for each DataFrame
    print(f"\nTable: {table_name}")
    print(df.head())
    print("\nDataFrame Info:")
    print(df.info())
    print("\nDataFrame Description:")
    print(df.describe())
    print("\n--------------------------------------------------------\n")
```



```
Table: id_drug
   ID Drug_Name
0   1   DB01083
1   2   DB00753
2   3   DB01262
3   4   DB01219
4   5   DB01085

DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 841 entries, 0 to 840
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   ID         841 non-null    int64 
 1   Drug_Name  841 non-null    object
dtypes: int64(1), object(1)
memory usage: 13.3+ KB
None

DataFrame Description:
               ID
count  841.000000
mean   421.000000
std    242.920083
min      1.000000
25%    211.000000
50%    421.000000
75%    631.000000
max    841.000000
```



```
Table: drug_drug
   Drug1  Drug2
0      1     32
1      1     39
2      1    285
3      1    217
4      1    385

DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 331308 entries, 0 to 331307
Data columns (total 2 columns):
 #   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   Drug1   331308 non-null  int64
 1   Drug2   331308 non-null  int64
dtypes: int64(2)
memory usage: 5.1 MB
None

DataFrame Description:
               Drug1          Drug2
count  331308.000000  331308.000000
mean      416.211821     416.211821
std       243.878501     243.878501
min         1.000000       1.000000
25%       203.000000     203.000000
50%       405.000000     405.000000
75%       625.000000     625.000000
max       841.000000     841.000000
```

```
Table: drug_enzyme
   Drug  Enzyme
0     1    3005
1     1    2958
2     2    2937
3     2    2916
4     2    2844

DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5992 entries, 0 to 5991
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Drug    5992 non-null   int64
 1   Enzyme  5992 non-null   int64
dtypes: int64(2)
memory usage: 93.8 KB
None

DataFrame Description:
              Drug       Enzyme
count  5992.000000  5992.000000
mean   1662.866155  1662.866155
std    1269.325560  1269.325560
min       1.000000     1.000000
25%     404.000000   404.000000
50%    1817.500000  1817.500000
75%    2932.000000  2932.000000
max    3007.000000  3007.000000
```

```
Table: drug_path
   Drug  Path
0     1  3069
1     1  3274
2     1  3049
3     1  3097
4     1  3115

DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23492 entries, 0 to 23491
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Drug    23492 non-null  int64
 1   Path    23492 non-null  int64
dtypes: int64(2)
memory usage: 367.2 KB
None

DataFrame Description:
               Drug          Path
count  23492.000000  23492.000000
mean    1794.592287   1794.592287
std     1381.053900   1381.053900
min        1.000000      1.000000
25%      433.000000    433.000000
50%     1924.500000   1924.500000
75%     3159.000000   3159.000000
max     3314.000000   3314.000000
```

```
Table: drug_structure
   Drug  Structure
0     1       1395
1     1       1297
2     1       1267
3     1       1394
4     1        843

DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209798 entries, 0 to 209797
Data columns (total 2 columns):
 #   Column     Non-Null Count   Dtype
---  ------     --------------   -----
 0   Drug       209798 non-null  int64
 1   Structure  209798 non-null  int64
dtypes: int64(2)
memory usage: 3.2 MB
None

DataFrame Description:
                Drug      Structure
count  209798.000000  209798.000000
mean      788.503894     788.503894
std       425.687700     425.687700
min         1.000000       1.000000
25%       413.000000     413.000000
50%       841.500000     841.500000
75%      1161.000000    1161.000000
max      1460.000000    1460.000000
```

```
Table: drug_target
   Drug  Target
0     1    2371
1     1    1659
2     1    2229
3     2    2263
4     2    2714

DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8630 entries, 0 to 8629
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Drug    8630 non-null   int64
 1   Target  8630 non-null   int64
dtypes: int64(2)
memory usage: 135.0 KB
None

DataFrame Description:
              Drug       Target
count  8630.000000  8630.000000
mean   1282.483082  1282.483082
std     923.510933   923.510933
min       1.000000     1.000000
25%     433.000000   433.000000
50%    1151.000000  1151.000000
75%    2128.000000  2128.000000
max    3006.000000  3006.000000
```

```
Table: drug_enzyme
   Drug  Enzyme
0     1    3005
1     1    2958
2     2    2937
3     2    2916
4     2    2844

DataFrame Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5992 entries, 0 to 5991
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Drug    5992 non-null   int64
 1   Enzyme  5992 non-null   int64
dtypes: int64(2)
memory usage: 93.8 KB
None

DataFrame Description:
              Drug       Enzyme
count  5992.000000  5992.000000
mean   1662.866155  1662.866155
std    1269.325560  1269.325560
min       1.000000     1.000000
25%     404.000000   404.000000
50%    1817.500000  1817.500000
75%    2932.000000  2932.000000
max    3007.000000  3007.000000
```

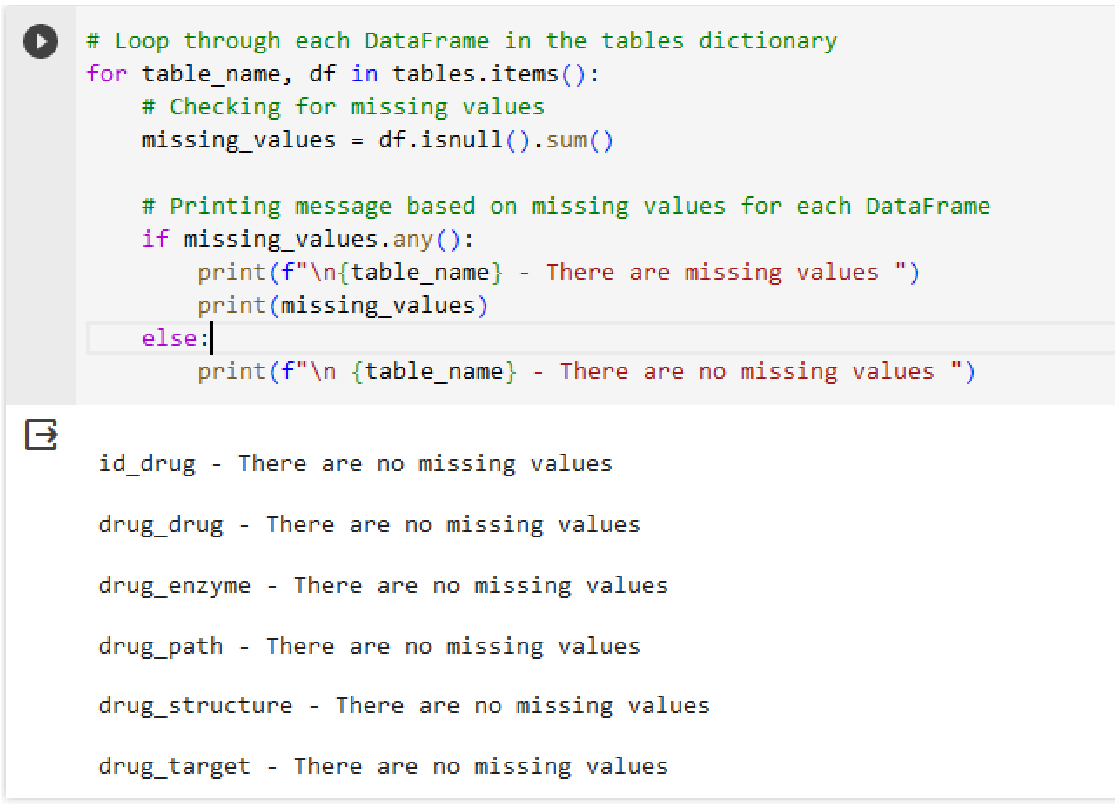
4.1.2 Python package for Data Science

```
import pandas as pd
import numpy as np
```

The line ‘import pandas as pd’ imports the Pandas library, a powerful tool for data manipulation and analysis in Python meanwhile ‘import numpy as np’ imports the NumPy library, a fundamental library for numerical computing in Python.

4.2 Data Wrangling

4.2.1 Identify missing values in the data

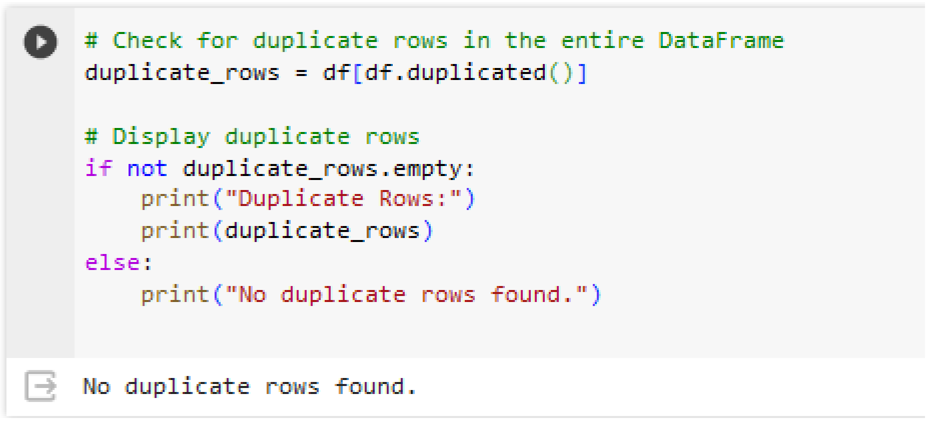


```python
# Loop through each DataFrame in the tables dictionary
for table_name, df in tables.items():
    # Checking for missing values
    missing_values = df.isnull().sum()

    # Printing message based on missing values for each DataFrame
    if missing_values.any():
        print(f"\n{table_name} - There are missing values ")
        print(missing_values)
    else:
        print(f"\n {table_name} - There are no missing values ")
```

```
 id_drug - There are no missing values

 drug_drug - There are no missing values

 drug_enzyme - There are no missing values

 drug_path - There are no missing values

 drug_structure - There are no missing values

 drug_target - There are no missing values
```

This code iterates through each DataFrame in the tables dictionary, checks for missing values, and prints a message for each DataFrame indicating whether it has missing values or not. If missing values are found, it prints the count of missing values for each column; otherwise, it prints a message stating that the data has no missing values.

4.2.2 Identify any duplicate row

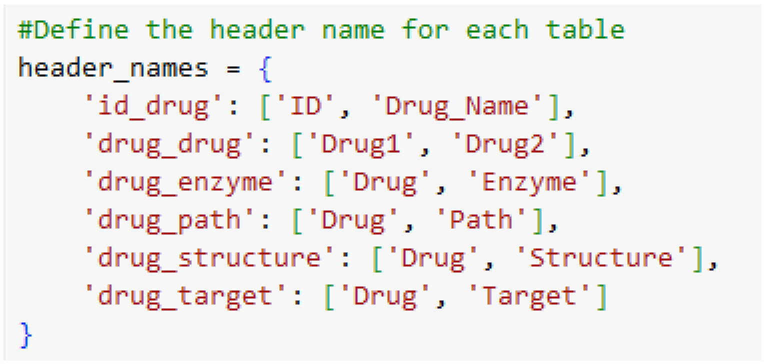


```python
# Check for duplicate rows in the entire DataFrame
duplicate_rows = df[df.duplicated()]

# Display duplicate rows
if not duplicate_rows.empty:
    print("Duplicate Rows:")
    print(duplicate_rows)
else:
    print("No duplicate rows found.")
```

```
No duplicate rows found.
```

We checked any duplicate row in the data and we found that no duplicate row was found.

4.3 Data Formatting

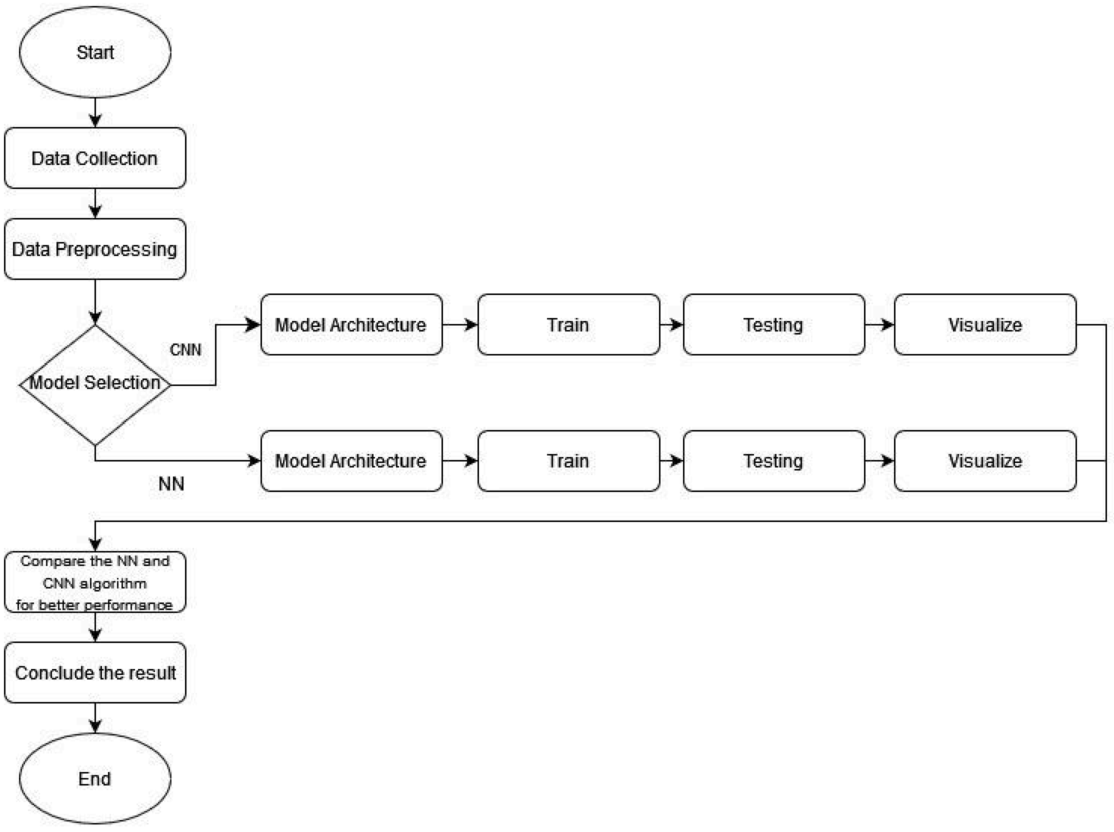


```python
#Define the header name for each table
header_names = {
    'id_drug': ['ID', 'Drug_Name'],
    'drug_drug': ['Drug1', 'Drug2'],
    'drug_enzyme': ['Drug', 'Enzyme'],
    'drug_path': ['Drug', 'Path'],
    'drug_structure': ['Drug', 'Structure'],
    'drug_target': ['Drug', 'Target']
}
```

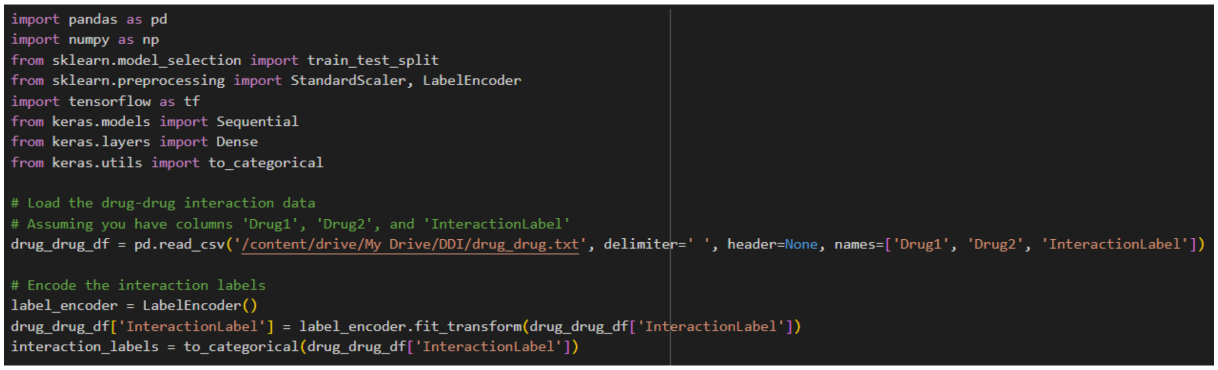
Headers are added in the data for each table because the data are headerless. Headers provide descriptive names for each column, making it easier for us to understand what each column represents.

5.0 Model Development and Evaluation

Flowchart for Model Development

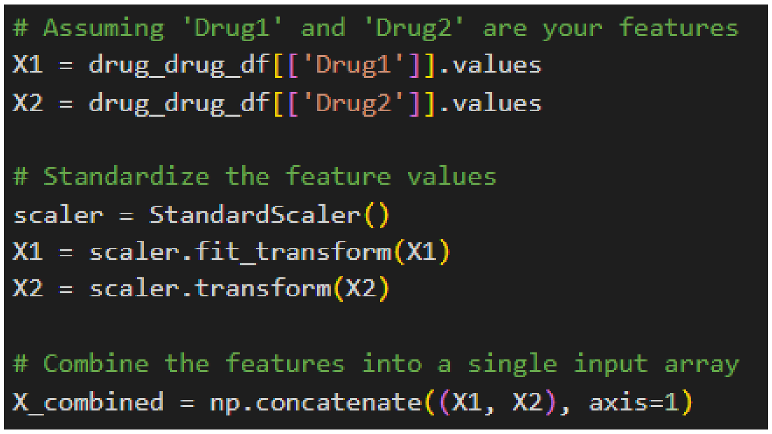


The drug-drug interaction data is loaded from a file ('/content/drive/My Drive/DDI/drug_drug.txt') using Pandas. The data has columns 'Drug1', 'Drug2', and 'InteractionLabel'. Interaction labels are encoded using LabelEncoder() and converted to categorical using to_categorical().



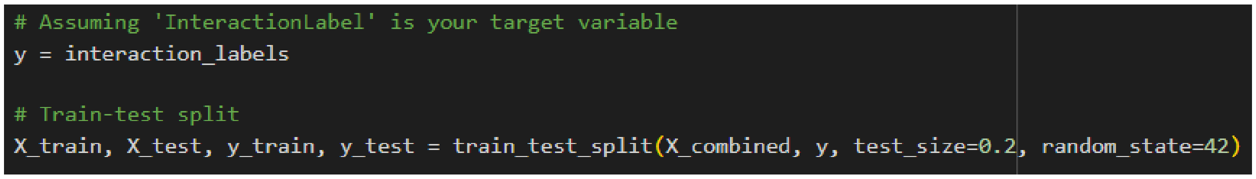
```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
import tensorflow as tf
from keras.models import Sequential
from keras.layers import Dense
from keras.utils import to_categorical

# Load the drug-drug interaction data
# Assuming you have columns 'Drug1', 'Drug2', and 'InteractionLabel'
drug_drug_df = pd.read_csv('/content/drive/My Drive/DDI/drug_drug.txt', delimiter=' ', header=None, names=['Drug1', 'Drug2', 'InteractionLabel'])

# Encode the interaction labels
label_encoder = LabelEncoder()
drug_drug_df['InteractionLabel'] = label_encoder.fit_transform(drug_drug_df['InteractionLabel'])
interaction_labels = to_categorical(drug_drug_df['InteractionLabel'])
```

The features ('Drug1' and 'Drug2') are standardized using StandardScaler(). The features are concatenated into a single input array.



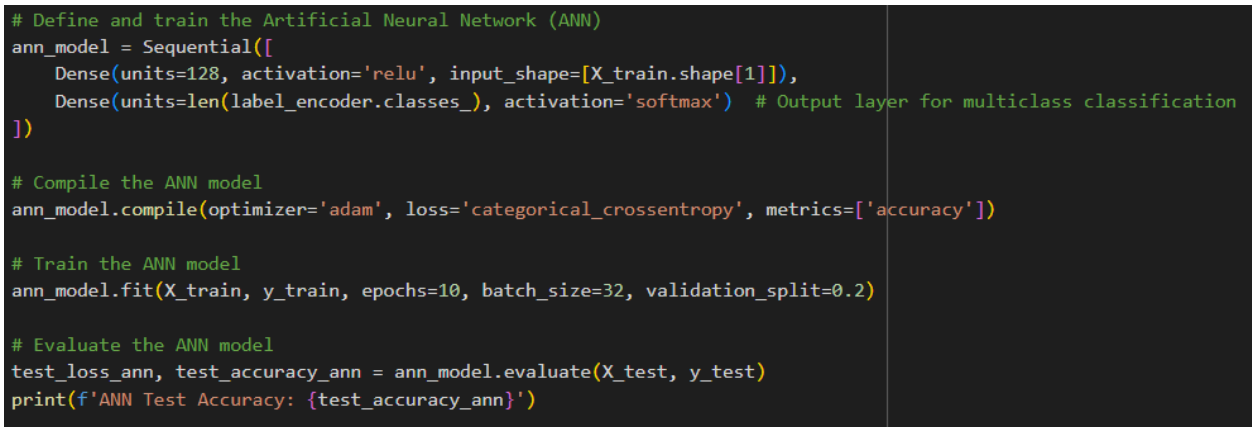
```python
# Assuming 'Drug1' and 'Drug2' are your features
X1 = drug_drug_df[['Drug1']].values
X2 = drug_drug_df[['Drug2']].values

# Standardize the feature values
scaler = StandardScaler()
X1 = scaler.fit_transform(X1)
X2 = scaler.transform(X2)

# Combine the features into a single input array
X_combined = np.concatenate((X1, X2), axis=1)
```

The data is split into training and testing sets.



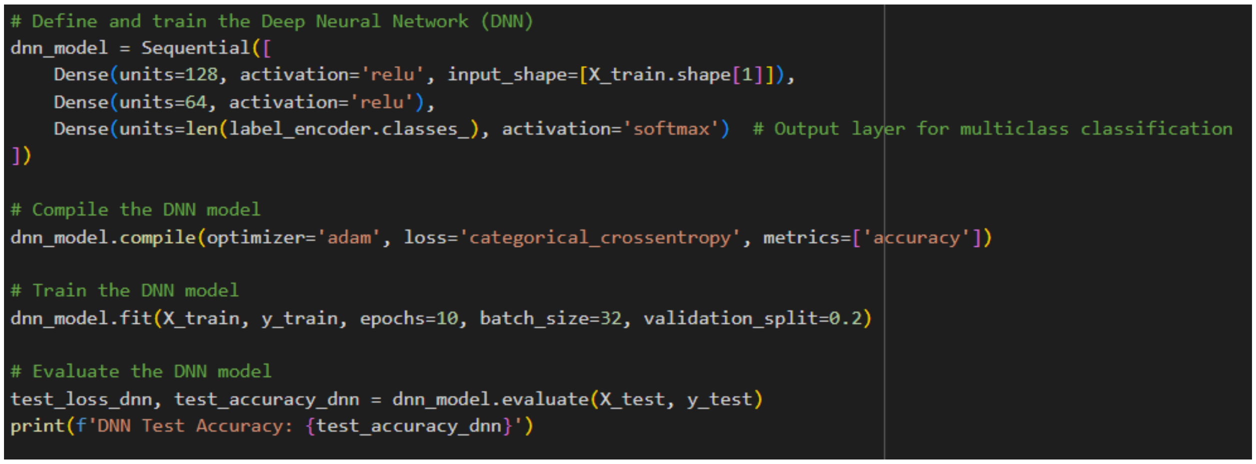
```python
# Assuming 'InteractionLabel' is your target variable
y = interaction_labels

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_combined, y, test_size=0.2, random_state=42)
```

A simple ANN model is defined with an input layer, a hidden layer with ReLU activation, and an output layer for multiclass classification.



```python
# Define and train the Artificial Neural Network (ANN)
ann_model = Sequential([
    Dense(units=128, activation='relu', input_shape=[X_train.shape[1]]),
    Dense(units=len(label_encoder.classes_), activation='softmax')  # Output layer for multiclass classification
])

# Compile the ANN model
ann_model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Train the ANN model
ann_model.fit(X_train, y_train, epochs=10, batch_size=32, validation_split=0.2)

# Evaluate the ANN model
test_loss_ann, test_accuracy_ann = ann_model.evaluate(X_test, y_test)
print(f'ANN Test Accuracy: {test_accuracy_ann}')
```

A DNN model is defined similarly with an additional hidden layer.



```
# Define and train the Deep Neural Network (DNN)
dnn_model = Sequential([
    Dense(units=128, activation='relu', input_shape=[X_train.shape[1]]),
    Dense(units=64, activation='relu'),
    Dense(units=len(label_encoder.classes_), activation='softmax')  # Output layer for multiclass classification
])

# Compile the DNN model
dnn_model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Train the DNN model
dnn_model.fit(X_train, y_train, epochs=10, batch_size=32, validation_split=0.2)

# Evaluate the DNN model
test_loss_dnn, test_accuracy_dnn = dnn_model.evaluate(X_test, y_test)
print(f'DNN Test Accuracy: {test_accuracy_dnn}')
```

REFERENCE

[1] Research, C. F. D. E. A. (2013, September 25). *Drug interactions: What you should know*. U.S. Food And Drug Administration. https://www.fda.gov/drugs/resources-drugs/drug-interactions-what-you-should-know

[2] Aldoseri, A., Al-Khalifa, K. N., & Hamouda, A. M. (2023). Re-Thinking Data Strategy and Integration for Artificial Intelligence: Concepts, Opportunities, and Challenges. *NATO Advanced Science Institutes Series E: Applied Sciences*, *13*(12), 7082. https://www.mdpi.com/2076-3417/13/12/7082

[3] Palleria, C., Di Paolo, A., Giofrè, C., Caglioti, C., Leuzzi, G., Siniscalchi, A., De Sarro, G., & Gallelli, L. (2013). Pharmacokinetic drug-drug interaction and their implication in clinical management. *Journal of Research in Medical Sciences: The Official Journal of Isfahan University of Medical Sciences*, *18*(7), 601. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3897029/

[4] Zheng, W. Y., Richardson, L. C., Li, L., Day, R. O., Westbrook, J. I., & Baysari, M. T. (2018). Drug-drug interactions and their harmful effects in hospitalised patients: a systematic review and meta-analysis. *European Journal of Clinical Pharmacology*, *74*(1). https://link.springer.com/article/10.1007/s00228-017-2357-5

[5] Zhang, C., Lu, Y., & Zang, T. (2022). CNN-DDI: a learning-based method for predicting drug–drug interactions using convolution neural networks. *BMC Bioinformatics*, *23*(S1). https://doi.org/10.1186/s12859-022-04612-2

[6] Rohani, N., & Eslahchi, C. (2019). Drug-Drug interaction predicting by neural network using integrated similarity. *Scientific Reports*, *9*(1). https://doi.org/10.1038/s41598-019-50121-3

[7] Juurlink, D. N. (2003). Drug-Drug interactions among elderly patients hospitalized for drug toxicity. *JAMA*, *289*(13), 1652. https://doi.org/10.1001/jama.289.13.1652