# Node Classification using Graph Convolutional Networks

This node classification task uses CORA dataset from https://linqs.soe.ucsc.edu/data
The dataset consists of **2708** nodes which correspond to scientific publications.
The nodes are classified into **7** categories indicating the topics of each document.
The edges indicate whether a document is cited by the other or vice versa.
Each node has **1433** features which is described by a 0/1-valued vector, indicating the bag-of-words from the dictionary.

This is an undirected graph problem

In [ ]:
```
#importing dependencies

import numpy as np
import os
import networkx as nx
from keras.utils import to_categorical
from sklearn.preprocessing import LabelEncoder
from sklearn.utils import shuffle
from sklearn.metrics import classification_report

from spektral.layers import GraphConv

from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Dropout, Dense
from tensorflow.keras import Sequential
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import TensorBoard, EarlyStopping
import tensorflow as tf
from tensorflow.keras.regularizers import l2

from collections import Counter
from sklearn.manifold import TSNE
import matplotlib.pyplot as plt
```

## Data Loading and Preprocessing

We are going to use the edges connecting the (from file **cora.cites**).
The nodes are loaded from file **cora.content**.

In [ ]:
```
#Loading the data

all_data = []
all_edges = []

for root,dirs,files in os.walk('./cora'):
    for file in files:
        if '.content' in file:
            with open(os.path.join(root,file),'r') as f:
                all_data.extend(f.read().splitlines())
        elif 'cites' in file:
            with open(os.path.join(root,file),'r') as f:
                all_edges.extend(f.read().splitlines())


#Shuffle the data because the raw data is ordered based on the label
random_state = 77
all_data = shuffle(all_data,random_state=random_state)
```

In **cora.content** file:
The **first** element indicates the **node name**

The **second** until the last second elements indicate the **node features**
The **last** element indicates the **label of that particular node**

In **cora.cites** file:
Each line indicates the tuple of connected nodes

## Parsing the data

In [ ]:
```
#parse the data
labels = []
nodes = []
X = []

for i,data in enumerate(all_data):
    elements = data.split('\t')
    labels.append(elements[-1])
    X.append(elements[1:-1])
    nodes.append(elements[0])

X = np.array(X,dtype=int)
N = X.shape[0] #the number of nodes
F = X.shape[1] #the size of node features
print('X shape: ', X.shape)


#parse the edge
edge_list=[]
for edge in all_edges:
    e = edge.split('\t')
    edge_list.append((e[0],e[1]))

print('\nNumber of nodes (N): ', N)
print('\nNumber of features (F) of each node: ', F)
print('\nCategories: ', set(labels))

num_classes = len(set(labels))
print('\nNumber of classes: ', num_classes)
```

```
X shape:  (2708, 1433)

Number of nodes (N):  2708

Number of features (F) of each node:  1433

Categories:  {'Neural_Networks', 'Reinforcement_Learning', 'Probabilistic_Methods', 'Genetic_Algorithms', 'Ru
le_Learning', 'Theory', 'Case_Based'}

Number of classes:  7
```

## Select examples for training, validation, and test then set the mask

In [ ]:
```
def limit_data(labels,limit=20,val_num=500,test_num=1000):
    '''
    Get the index of train, validation, and test data
    '''
    label_counter = dict((l, 0) for l in labels)
    train_idx = []

    for i in range(len(labels)):
        label = labels[i]
        if label_counter[label]<limit:
            #add the example to the training data
            train_idx.append(i)
            label_counter[label]+=1

        #exit the loop once we found 20 examples for each class
        if all(count == limit for count in label_counter.values()):
            break
```

```
    #get the indices that do not go to traning data
    rest_idx = [x for x in range(len(labels)) if x not in train_idx]
    #get the first val_num
    val_idx = rest_idx[:val_num]
    test_idx = rest_idx[val_num:(val_num+test_num)]
    return train_idx, val_idx,test_idx


train_idx,val_idx,test_idx = limit_data(labels)
```

In [ ]:
```
#set the mask
train_mask = np.zeros((N,),dtype=bool)
train_mask[train_idx] = True

val_mask = np.zeros((N,),dtype=bool)
val_mask[val_idx] = True

test_mask = np.zeros((N,),dtype=bool)
test_mask[test_idx] = True
```

## Show Data Distribution

In [ ]:
```
print("All Data Distribution: \n{}".format(Counter(labels)))
```

```
All Data Distribution:
Counter({'Neural_Networks': 818, 'Probabilistic_Methods': 426, 'Genetic_Algorithms': 418, 'Theory': 351, 'Cas
e_Based': 298, 'Reinforcement_Learning': 217, 'Rule_Learning': 180})
```

In [ ]:
```
print("Training Data Distribution: \n{}".format(Counter([labels[i] for i in train_idx])))
```

```
Training Data Distribution:
Counter({'Reinforcement_Learning': 20, 'Probabilistic_Methods': 20, 'Neural_Networks': 20, 'Case_Based': 20,
'Theory': 20, 'Genetic_Algorithms': 20, 'Rule_Learning': 20})
```

In [ ]:
```
print("Validation Data Distribution: \n{}".format(Counter([labels[i] for i in val_idx])))
```

```
Validation Data Distribution:
Counter({'Neural_Networks': 172, 'Genetic_Algorithms': 78, 'Probabilistic_Methods': 72, 'Theory': 63, 'Case_B
ased': 58, 'Reinforcement_Learning': 35, 'Rule_Learning': 22})
```

## Convert the labels to one hot encoding

In [ ]:
```
def encode_label(labels):
    label_encoder = LabelEncoder()
    labels = label_encoder.fit_transform(labels)
    labels = to_categorical(labels)
    return labels, label_encoder.classes_

labels_encoded, classes = encode_label(labels)
```

## Build a graph on NetworkX using the obtained nodes and edges list

In [ ]:
```
#build the graph
G = nx.Graph()
G.add_nodes_from(nodes)
G.add_edges_from(edge_list)

#obtain the adjacency matrix (A)
A = nx.adjacency_matrix(G)
print('Graph info: ', nx.info(G))
```

```
Graph info:  Name:
Type: Graph
Number of nodes: 2708
Number of edges: 5278
Average degree:   3.8981
```

# Building and Training Graph Convolutional Networks

In [ ]:
```
# Parameters
channels = 16           # Number of channels in the first layer
dropout = 0.5           # Dropout rate for the features
l2_reg = 5e-4           # L2 regularization rate
learning_rate = 1e-2    # Learning rate
epochs = 100            # Number of training epochs
es_patience = 50        # Patience for early stopping

# Preprocessing operations
A = GraphConv.preprocess(A).astype('f4')

# Model definition
X_in = Input(shape=(F, ))
fltr_in = Input((N, ), sparse=True)

dropout_1 = Dropout(dropout)(X_in)
graph_conv_1 = GraphConv(channels,
                         activation='relu',
                         kernel_regularizer=l2(l2_reg),
                         use_bias=False)([dropout_1, fltr_in])

dropout_2 = Dropout(dropout)(graph_conv_1)
graph_conv_2 = GraphConv(num_classes,
                         activation='softmax',
                         use_bias=False)([dropout_2, fltr_in])

# Build model
model = Model(inputs=[X_in, fltr_in], outputs=graph_conv_2)
optimizer = Adam(lr=learning_rate)
model.compile(optimizer=optimizer,
              loss='categorical_crossentropy',
              weighted_metrics=['acc'])
model.summary()

tbCallBack_GCN = tf.keras.callbacks.TensorBoard(
    log_dir='./Tensorboard_GCN_cora',
)
callback_GCN = [tbCallBack_GCN]
```

```
Model: "model"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            [(None, 1433)]       0

__________________________________________________________________________________________________
dropout (Dropout)               (None, 1433)         0           input_1[0][0]
__________________________________________________________________________________________________
input_2 (InputLayer)            [(None, 2708)]       0
__________________________________________________________________________________________________
graph_conv (GraphConv)          (None, 16)           22928       dropout[0][0]
                                                                 input_2[0][0]
__________________________________________________________________________________________________
dropout_1 (Dropout)             (None, 16)           0           graph_conv[0][0]
__________________________________________________________________________________________________
graph_conv_1 (GraphConv)        (None, 7)            112         dropout_1[0][0]
                                                                 input_2[0][0]
==================================================================================================
Total params: 23,040
Trainable params: 23,040
Non-trainable params: 0
__________________________________________________________________________________________________
```

In [ ]:
```
# Train model
validation_data = ([X, A], labels_encoded, val_mask)
model.fit([X, A],
          labels_encoded,
          sample_weight=train_mask,
          epochs=epochs,
```

```
        batch_size=N,
        validation_data=validation_data,
        shuffle=False,
        callbacks=[
            EarlyStopping(patience=es_patience,  restore_best_weights=True),
            tbCallBack_GCN
        ])
```

```
Epoch 1/100
1/1 [==============================] - 0s 434ms/step - loss: 0.1162 - acc: 0.1429 - val_loss: 0.3662 - val_ac
c: 0.2420
Epoch 2/100
1/1 [==============================] - ETA: 0s - loss: 0.1094 - acc: 0.2929WARNING:tensorflow:Method (on_trai
n_batch_end) is slow compared to the batch update (0.195233). Check your callbacks.
1/1 [==============================] - 0s 187ms/step - loss: 0.1094 - acc: 0.2929 - val_loss: 0.3570 - val_ac
c: 0.3440
Epoch 3/100
1/1 [==============================] - 0s 193ms/step - loss: 0.1030 - acc: 0.4143 - val_loss: 0.3469 - val_ac
c: 0.3960
Epoch 4/100
1/1 [==============================] - 0s 300ms/step - loss: 0.0976 - acc: 0.5571 - val_loss: 0.3371 - val_ac
c: 0.4100
Epoch 5/100
1/1 [==============================] - 0s 290ms/step - loss: 0.0918 - acc: 0.5357 - val_loss: 0.3280 - val_ac
c: 0.4220
Epoch 6/100
1/1 [==============================] - 0s 266ms/step - loss: 0.0872 - acc: 0.5929 - val_loss: 0.3197 - val_ac
c: 0.4280
Epoch 7/100
1/1 [==============================] - 0s 249ms/step - loss: 0.0831 - acc: 0.5714 - val_loss: 0.3115 - val_ac
c: 0.4420
Epoch 8/100
1/1 [==============================] - 0s 256ms/step - loss: 0.0805 - acc: 0.6071 - val_loss: 0.3032 - val_ac
c: 0.4760
Epoch 9/100
1/1 [==============================] - 0s 256ms/step - loss: 0.0769 - acc: 0.6643 - val_loss: 0.2949 - val_ac
c: 0.5280
Epoch 10/100
1/1 [==============================] - 0s 232ms/step - loss: 0.0754 - acc: 0.7571 - val_loss: 0.2871 - val_ac
c: 0.5740
Epoch 11/100
1/1 [==============================] - 0s 256ms/step - loss: 0.0743 - acc: 0.7286 - val_loss: 0.2799 - val_ac
c: 0.5980
Epoch 12/100
1/1 [==============================] - 0s 263ms/step - loss: 0.0732 - acc: 0.7429 - val_loss: 0.2731 - val_ac
c: 0.6360
Epoch 13/100
1/1 [==============================] - 0s 408ms/step - loss: 0.0708 - acc: 0.7429 - val_loss: 0.2664 - val_ac
c: 0.6600
Epoch 14/100
1/1 [==============================] - 0s 240ms/step - loss: 0.0694 - acc: 0.8143 - val_loss: 0.2597 - val_ac
c: 0.6900
Epoch 15/100
1/1 [==============================] - 0s 242ms/step - loss: 0.0648 - acc: 0.8571 - val_loss: 0.2531 - val_ac
c: 0.7140
Epoch 16/100
1/1 [==============================] - 0s 218ms/step - loss: 0.0650 - acc: 0.8429 - val_loss: 0.2467 - val_ac
c: 0.7340
Epoch 17/100
1/1 [==============================] - 0s 247ms/step - loss: 0.0636 - acc: 0.8857 - val_loss: 0.2404 - val_ac
c: 0.7540
Epoch 18/100
1/1 [==============================] - 0s 244ms/step - loss: 0.0640 - acc: 0.8643 - val_loss: 0.2342 - val_ac
c: 0.7640
Epoch 19/100
1/1 [==============================] - 0s 257ms/step - loss: 0.0618 - acc: 0.9143 - val_loss: 0.2288 - val_ac
c: 0.7760
Epoch 20/100
1/1 [==============================] - 0s 302ms/step - loss: 0.0610 - acc: 0.9071 - val_loss: 0.2237 - val_ac
c: 0.7820
Epoch 21/100
1/1 [==============================] - 0s 236ms/step - loss: 0.0588 - acc: 0.9143 - val_loss: 0.2191 - val_ac
c: 0.7820
Epoch 22/100
1/1 [==============================] - 0s 206ms/step - loss: 0.0571 - acc: 0.9000 - val_loss: 0.2149 - val_ac
c: 0.7860
Epoch 23/100
1/1 [==============================] - 0s 243ms/step - loss: 0.0564 - acc: 0.9000 - val_loss: 0.2111 - val_ac
c: 0.7820
```

```
Epoch 24/100
1/1 [==============================] - 0s 237ms/step - loss: 0.0565 - acc: 0.8929 - val_loss: 0.2076 - val_ac
c: 0.7880
Epoch 25/100
1/1 [==============================] - 0s 254ms/step - loss: 0.0551 - acc: 0.9286 - val_loss: 0.2049 - val_ac
c: 0.7840
Epoch 26/100
1/1 [==============================] - 0s 244ms/step - loss: 0.0543 - acc: 0.9214 - val_loss: 0.2023 - val_ac
c: 0.7880
Epoch 27/100
1/1 [==============================] - 0s 254ms/step - loss: 0.0539 - acc: 0.9214 - val_loss: 0.1998 - val_ac
c: 0.7900
Epoch 28/100
1/1 [==============================] - 0s 247ms/step - loss: 0.0530 - acc: 0.9357 - val_loss: 0.1978 - val_ac
c: 0.7940
Epoch 29/100
1/1 [==============================] - 0s 228ms/step - loss: 0.0515 - acc: 0.9214 - val_loss: 0.1960 - val_ac
c: 0.7920
Epoch 30/100
1/1 [==============================] - 0s 200ms/step - loss: 0.0511 - acc: 0.9214 - val_loss: 0.1941 - val_ac
c: 0.7880
Epoch 31/100
1/1 [==============================] - 0s 187ms/step - loss: 0.0499 - acc: 0.9214 - val_loss: 0.1920 - val_ac
c: 0.7960
Epoch 32/100
1/1 [==============================] - 0s 186ms/step - loss: 0.0497 - acc: 0.9357 - val_loss: 0.1898 - val_ac
c: 0.7960
Epoch 33/100
1/1 [==============================] - 0s 183ms/step - loss: 0.0479 - acc: 0.9214 - val_loss: 0.1876 - val_ac
c: 0.7940
Epoch 34/100
1/1 [==============================] - 0s 183ms/step - loss: 0.0466 - acc: 0.9643 - val_loss: 0.1859 - val_ac
c: 0.7960
Epoch 35/100
1/1 [==============================] - 0s 195ms/step - loss: 0.0466 - acc: 0.9143 - val_loss: 0.1846 - val_ac
c: 0.7960
Epoch 36/100
1/1 [==============================] - 0s 192ms/step - loss: 0.0464 - acc: 0.9143 - val_loss: 0.1836 - val_ac
c: 0.7940
Epoch 37/100
1/1 [==============================] - 0s 173ms/step - loss: 0.0433 - acc: 0.9571 - val_loss: 0.1826 - val_ac
c: 0.7940
Epoch 38/100
1/1 [==============================] - 0s 180ms/step - loss: 0.0446 - acc: 0.9571 - val_loss: 0.1814 - val_ac
c: 0.7940
Epoch 39/100
1/1 [==============================] - 0s 163ms/step - loss: 0.0447 - acc: 0.9143 - val_loss: 0.1802 - val_ac
c: 0.7980
Epoch 40/100
1/1 [==============================] - 0s 191ms/step - loss: 0.0467 - acc: 0.9357 - val_loss: 0.1791 - val_ac
c: 0.7960
Epoch 41/100
1/1 [==============================] - 0s 170ms/step - loss: 0.0442 - acc: 0.9286 - val_loss: 0.1778 - val_ac
c: 0.7940
Epoch 42/100
1/1 [==============================] - 0s 190ms/step - loss: 0.0436 - acc: 0.9500 - val_loss: 0.1768 - val_ac
c: 0.7960
Epoch 43/100
1/1 [==============================] - 0s 178ms/step - loss: 0.0448 - acc: 0.9214 - val_loss: 0.1758 - val_ac
c: 0.7960
Epoch 44/100
1/1 [==============================] - 0s 176ms/step - loss: 0.0419 - acc: 0.9357 - val_loss: 0.1745 - val_ac
c: 0.7900
Epoch 45/100
1/1 [==============================] - 0s 161ms/step - loss: 0.0422 - acc: 0.9286 - val_loss: 0.1736 - val_ac
c: 0.7900
Epoch 46/100
1/1 [==============================] - 0s 173ms/step - loss: 0.0419 - acc: 0.9500 - val_loss: 0.1725 - val_ac
c: 0.7840
Epoch 47/100
1/1 [==============================] - 0s 169ms/step - loss: 0.0406 - acc: 0.9786 - val_loss: 0.1712 - val_ac
```

```
c: 0.7860
Epoch 48/100
1/1 [==============================] - 0s 170ms/step - loss: 0.0411 - acc: 0.9429 - val_loss: 0.1705 - val_ac
c: 0.7900
Epoch 49/100
1/1 [==============================] - 0s 166ms/step - loss: 0.0418 - acc: 0.9429 - val_loss: 0.1692 - val_ac
c: 0.7840
Epoch 50/100
1/1 [==============================] - 0s 170ms/step - loss: 0.0403 - acc: 0.9500 - val_loss: 0.1679 - val_ac
c: 0.7800
Epoch 51/100
1/1 [==============================] - 0s 186ms/step - loss: 0.0380 - acc: 0.9500 - val_loss: 0.1672 - val_ac
c: 0.7800
Epoch 52/100
1/1 [==============================] - 0s 174ms/step - loss: 0.0393 - acc: 0.9500 - val_loss: 0.1663 - val_ac
c: 0.7820
Epoch 53/100
1/1 [==============================] - 0s 169ms/step - loss: 0.0396 - acc: 0.9500 - val_loss: 0.1657 - val_ac
c: 0.7860
Epoch 54/100
1/1 [==============================] - 0s 178ms/step - loss: 0.0384 - acc: 0.9286 - val_loss: 0.1653 - val_ac
c: 0.7820
Epoch 55/100
1/1 [==============================] - 0s 171ms/step - loss: 0.0369 - acc: 0.9714 - val_loss: 0.1651 - val_ac
c: 0.7820
Epoch 56/100
1/1 [==============================] - 0s 182ms/step - loss: 0.0391 - acc: 0.9214 - val_loss: 0.1651 - val_ac
c: 0.7820
Epoch 57/100
1/1 [==============================] - 0s 188ms/step - loss: 0.0373 - acc: 0.9571 - val_loss: 0.1646 - val_ac
c: 0.7840
Epoch 58/100
1/1 [==============================] - 0s 180ms/step - loss: 0.0368 - acc: 0.9571 - val_loss: 0.1637 - val_ac
c: 0.7780
Epoch 59/100
1/1 [==============================] - 0s 180ms/step - loss: 0.0374 - acc: 0.9643 - val_loss: 0.1621 - val_ac
c: 0.7780
Epoch 60/100
1/1 [==============================] - 0s 180ms/step - loss: 0.0360 - acc: 0.9714 - val_loss: 0.1609 - val_ac
c: 0.7800
Epoch 61/100
1/1 [==============================] - 0s 177ms/step - loss: 0.0379 - acc: 0.9786 - val_loss: 0.1602 - val_ac
c: 0.7880
Epoch 62/100
1/1 [==============================] - 0s 176ms/step - loss: 0.0362 - acc: 0.9429 - val_loss: 0.1603 - val_ac
c: 0.7860
Epoch 63/100
1/1 [==============================] - 0s 184ms/step - loss: 0.0363 - acc: 0.9500 - val_loss: 0.1611 - val_ac
c: 0.7840
Epoch 64/100
1/1 [==============================] - 0s 216ms/step - loss: 0.0340 - acc: 0.9714 - val_loss: 0.1615 - val_ac
c: 0.7860
Epoch 65/100
1/1 [==============================] - 0s 270ms/step - loss: 0.0354 - acc: 0.9357 - val_loss: 0.1605 - val_ac
c: 0.7920
Epoch 66/100
1/1 [==============================] - 0s 200ms/step - loss: 0.0350 - acc: 0.9500 - val_loss: 0.1594 - val_ac
c: 0.8020
Epoch 67/100
1/1 [==============================] - 0s 185ms/step - loss: 0.0345 - acc: 0.9714 - val_loss: 0.1592 - val_ac
c: 0.7960
Epoch 68/100
1/1 [==============================] - 0s 188ms/step - loss: 0.0353 - acc: 0.9429 - val_loss: 0.1592 - val_ac
c: 0.7960
Epoch 69/100
1/1 [==============================] - 0s 178ms/step - loss: 0.0364 - acc: 0.9286 - val_loss: 0.1594 - val_ac
c: 0.7920
Epoch 70/100
1/1 [==============================] - 0s 180ms/step - loss: 0.0346 - acc: 0.9857 - val_loss: 0.1597 - val_ac
c: 0.7800
Epoch 71/100
```

```
1/1 [==============================] - 0s 181ms/step - loss: 0.0354 - acc: 0.9429 - val_loss: 0.1603 - val_ac
c: 0.7700
Epoch 72/100
1/1 [==============================] - 0s 195ms/step - loss: 0.0334 - acc: 0.9714 - val_loss: 0.1599 - val_ac
c: 0.7700
Epoch 73/100
1/1 [==============================] - 0s 174ms/step - loss: 0.0311 - acc: 0.9714 - val_loss: 0.1587 - val_ac
c: 0.7720
Epoch 74/100
1/1 [==============================] - 0s 191ms/step - loss: 0.0367 - acc: 0.9429 - val_loss: 0.1571 - val_ac
c: 0.7760
Epoch 75/100
1/1 [==============================] - 0s 183ms/step - loss: 0.0342 - acc: 0.9429 - val_loss: 0.1555 - val_ac
c: 0.7820
Epoch 76/100
1/1 [==============================] - 0s 183ms/step - loss: 0.0337 - acc: 0.9500 - val_loss: 0.1541 - val_ac
c: 0.7880
Epoch 77/100
1/1 [==============================] - 0s 190ms/step - loss: 0.0353 - acc: 0.9714 - val_loss: 0.1531 - val_ac
c: 0.7940
Epoch 78/100
1/1 [==============================] - 0s 178ms/step - loss: 0.0311 - acc: 0.9714 - val_loss: 0.1528 - val_ac
c: 0.8040
Epoch 79/100
1/1 [==============================] - 0s 183ms/step - loss: 0.0337 - acc: 0.9357 - val_loss: 0.1534 - val_ac
c: 0.8040
Epoch 80/100
1/1 [==============================] - 0s 178ms/step - loss: 0.0323 - acc: 0.9714 - val_loss: 0.1546 - val_ac
c: 0.7960
Epoch 81/100
1/1 [==============================] - 0s 181ms/step - loss: 0.0341 - acc: 0.9500 - val_loss: 0.1557 - val_ac
c: 0.7840
Epoch 82/100
1/1 [==============================] - 0s 190ms/step - loss: 0.0353 - acc: 0.9214 - val_loss: 0.1560 - val_ac
c: 0.7840
Epoch 83/100
1/1 [==============================] - 0s 189ms/step - loss: 0.0316 - acc: 0.9714 - val_loss: 0.1551 - val_ac
c: 0.7820
Epoch 84/100
1/1 [==============================] - 0s 184ms/step - loss: 0.0337 - acc: 0.9429 - val_loss: 0.1542 - val_ac
c: 0.7880
Epoch 85/100
1/1 [==============================] - 0s 185ms/step - loss: 0.0323 - acc: 0.9500 - val_loss: 0.1537 - val_ac
c: 0.7820
Epoch 86/100
1/1 [==============================] - 0s 182ms/step - loss: 0.0302 - acc: 0.9714 - val_loss: 0.1541 - val_ac
c: 0.7860
Epoch 87/100
1/1 [==============================] - 0s 179ms/step - loss: 0.0319 - acc: 0.9571 - val_loss: 0.1540 - val_ac
c: 0.7740
Epoch 88/100
1/1 [==============================] - 0s 182ms/step - loss: 0.0335 - acc: 0.9357 - val_loss: 0.1546 - val_ac
c: 0.7660
Epoch 89/100
1/1 [==============================] - 0s 184ms/step - loss: 0.0313 - acc: 0.9571 - val_loss: 0.1565 - val_ac
c: 0.7700
Epoch 90/100
1/1 [==============================] - 0s 180ms/step - loss: 0.0310 - acc: 0.9714 - val_loss: 0.1569 - val_ac
c: 0.7660
Epoch 91/100
1/1 [==============================] - 0s 181ms/step - loss: 0.0321 - acc: 0.9571 - val_loss: 0.1559 - val_ac
c: 0.7740
Epoch 92/100
1/1 [==============================] - 0s 187ms/step - loss: 0.0290 - acc: 0.9571 - val_loss: 0.1554 - val_ac
c: 0.7820
Epoch 93/100
1/1 [==============================] - 0s 187ms/step - loss: 0.0330 - acc: 0.9500 - val_loss: 0.1542 - val_ac
c: 0.7940
Epoch 94/100
1/1 [==============================] - 0s 178ms/step - loss: 0.0303 - acc: 0.9714 - val_loss: 0.1532 - val_ac
c: 0.8000
```

```
Epoch 95/100
1/1 [==============================] - 0s 183ms/step - loss: 0.0295 - acc: 0.9857 - val_loss: 0.1520 - val_ac
c: 0.7980
Epoch 96/100
1/1 [==============================] - 0s 191ms/step - loss: 0.0297 - acc: 0.9500 - val_loss: 0.1509 - val_ac
c: 0.7920
Epoch 97/100
1/1 [==============================] - 0s 185ms/step - loss: 0.0280 - acc: 0.9857 - val_loss: 0.1499 - val_ac
c: 0.7800
Epoch 98/100
1/1 [==============================] - 0s 185ms/step - loss: 0.0293 - acc: 0.9714 - val_loss: 0.1486 - val_ac
c: 0.7800
Epoch 99/100
1/1 [==============================] - 0s 187ms/step - loss: 0.0296 - acc: 0.9786 - val_loss: 0.1482 - val_ac
c: 0.7780
Epoch 100/100
1/1 [==============================] - 0s 194ms/step - loss: 0.0298 - acc: 0.9500 - val_loss: 0.1482 - val_ac
c: 0.7780
```

Out[ ]: `<tensorflow.python.keras.callbacks.History at 0x1309f37b648>`

In [ ]:
```python
# Evaluate model
X_te = X[test_mask]
A_te = A[test_mask,:][:,test_mask]
y_te = labels_encoded[test_mask]

y_pred = model.predict([X_te, A_te], batch_size=N)
report = classification_report(np.argmax(y_te,axis=1), np.argmax(y_pred,axis=1), target_names=classes)
print('GCN Classification Report: \n {}'.format(report))
```

```
GCN Classification Report:
                         precision    recall  f1-score   support

            Case_Based       0.68      0.72      0.70       114
    Genetic_Algorithms       0.86      0.87      0.87       156
       Neural_Networks       0.80      0.68      0.74       290
 Probabilistic_Methods       0.78      0.74      0.76       172
Reinforcement_Learning       0.70      0.79      0.74        85
         Rule_Learning       0.56      0.73      0.64        60
                Theory       0.61      0.68      0.64       123

              accuracy                           0.74      1000
             macro avg       0.71      0.74      0.73      1000
          weighted avg       0.75      0.74      0.74      1000
```

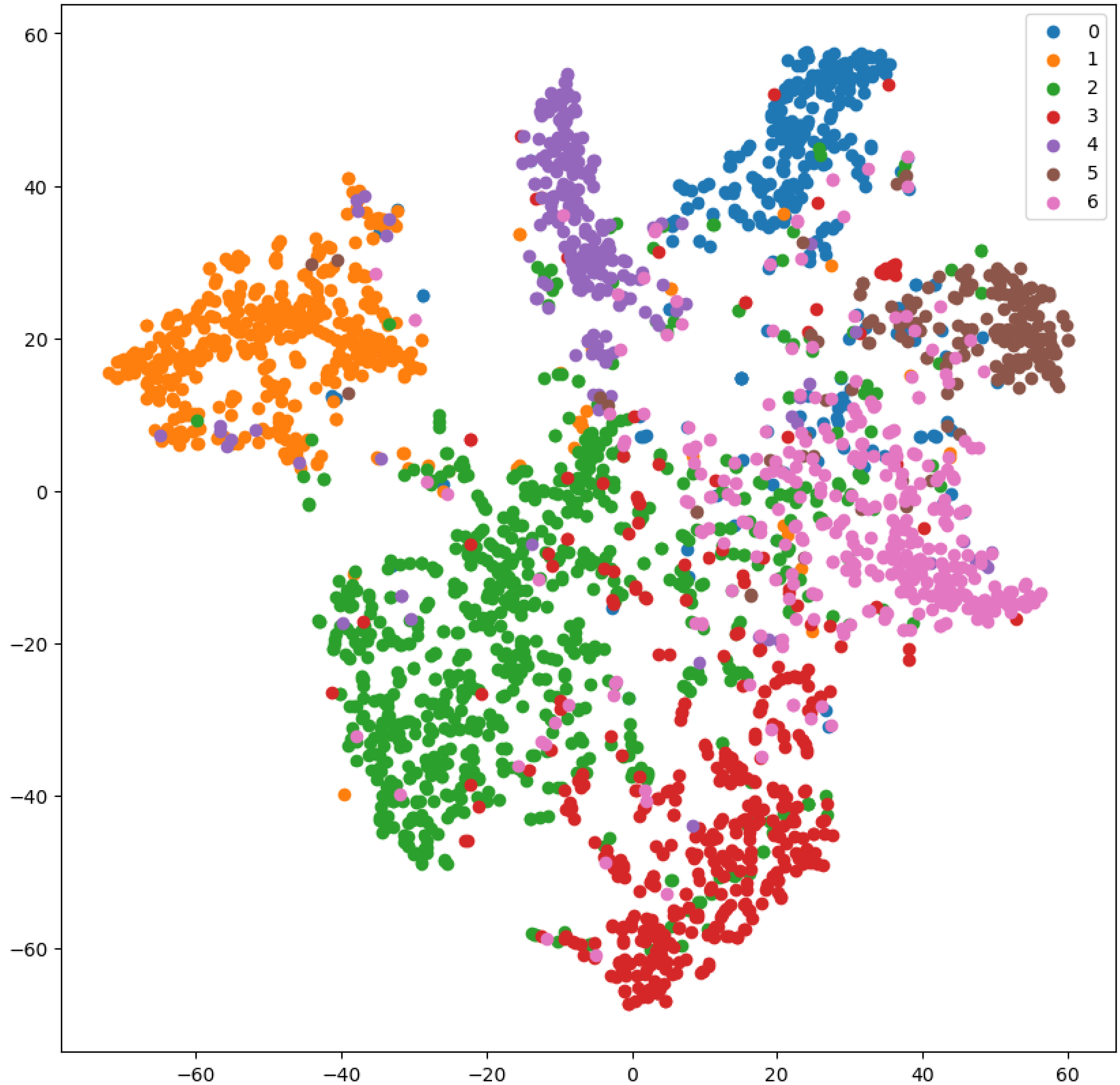
## Get hidden layer representation for GCN

In [ ]:
```python
layer_outputs = [layer.output for layer in model.layers]
activation_model = Model(inputs=model.input, outputs=layer_outputs)
activations = activation_model.predict([X,A],batch_size=N)

#Get t-SNE Representation
#get the hidden layer representation after the first GCN layer
x_tsne = TSNE(n_components=2).fit_transform(activations[3])
```

In [ ]:
```python
def plot_tSNE(labels_encoded,x_tsne):
    color_map = np.argmax(labels_encoded, axis=1)
    plt.figure(figsize=(10,10))
    for cl in range(num_classes):
        indices = np.where(color_map==cl)
        indices = indices[0]
        plt.scatter(x_tsne[indices,0], x_tsne[indices, 1], label=cl)
    plt.legend()
    plt.show()

plot_tSNE(labels_encoded,x_tsne)
```




## Comparison to Fully-Connected Neural Networks

### Building and Training FNN

In [ ]:
```
es_patience = 50
optimizer = Adam(lr=1e-2)
l2_reg = 5e-4
epochs = 100

#Compare with FNN
#Construct the model
model_fnn = Sequential()
model_fnn.add(Dense(
                    128,
                    input_dim=X.shape[1],
                    activation=tf.nn.relu,
                    kernel_regularizer=tf.keras.regularizers.l2(l2_reg))
              )
model_fnn.add(Dropout(0.5))
model_fnn.add(Dense(256, activation=tf.nn.relu))
model_fnn.add(Dropout(0.5))
model_fnn.add(Dense(num_classes, activation=tf.keras.activations.softmax))
```

```
model_fnn.compile(optimizer=optimizer,
              loss='categorical_crossentropy',
              weighted_metrics=['acc'])


#define TensorBoard
tbCallBack_FNN = TensorBoard(
    log_dir='./Tensorboard_FNN_cora',
)

#Train model
validation_data_fnn = (X, labels_encoded, val_mask)
model_fnn.fit(
                X,labels_encoded,
                sample_weight=train_mask,
                epochs=epochs,
                batch_size=N,
                validation_data=validation_data_fnn,
                shuffle=False,
                callbacks=[
                  EarlyStopping(patience=es_patience,  restore_best_weights=True),
                  tbCallBack_FNN
          ])
```

```
Epoch 1/100
1/1 [==============================] - 0s 302ms/step - loss: 0.2181 - acc: 0.1786 - val_loss: 0.4333 - val_ac
c: 0.3040
Epoch 2/100
1/1 [==============================] - ETA: 0s - loss: 0.1741 - acc: 0.3929WARNING:tensorflow:Method (on_trai
n_batch_end) is slow compared to the batch update (0.168763). Check your callbacks.
1/1 [==============================] - 0s 190ms/step - loss: 0.1741 - acc: 0.3929 - val_loss: 0.3964 - val_ac
c: 0.4340
Epoch 3/100
1/1 [==============================] - 0s 183ms/step - loss: 0.1381 - acc: 0.6643 - val_loss: 0.3627 - val_ac
c: 0.5120
Epoch 4/100
1/1 [==============================] - 0s 185ms/step - loss: 0.1091 - acc: 0.7714 - val_loss: 0.3299 - val_ac
c: 0.5480
Epoch 5/100
1/1 [==============================] - 0s 184ms/step - loss: 0.0850 - acc: 0.8286 - val_loss: 0.2965 - val_ac
c: 0.5880
Epoch 6/100
1/1 [==============================] - 0s 190ms/step - loss: 0.0630 - acc: 0.8857 - val_loss: 0.2728 - val_ac
c: 0.5760
Epoch 7/100
1/1 [==============================] - 0s 187ms/step - loss: 0.0523 - acc: 0.9143 - val_loss: 0.2673 - val_ac
c: 0.5560
Epoch 8/100
1/1 [==============================] - 0s 246ms/step - loss: 0.0442 - acc: 0.9643 - val_loss: 0.2756 - val_ac
c: 0.5640
Epoch 9/100
1/1 [==============================] - 0s 292ms/step - loss: 0.0410 - acc: 0.9714 - val_loss: 0.2885 - val_ac
c: 0.5600
Epoch 10/100
1/1 [==============================] - 0s 378ms/step - loss: 0.0408 - acc: 0.9929 - val_loss: 0.3009 - val_ac
c: 0.5760
Epoch 11/100
1/1 [==============================] - 0s 266ms/step - loss: 0.0436 - acc: 0.9714 - val_loss: 0.3309 - val_ac
c: 0.5460
Epoch 12/100
1/1 [==============================] - 0s 271ms/step - loss: 0.0414 - acc: 0.9857 - val_loss: 0.3530 - val_ac
c: 0.5300
Epoch 13/100
1/1 [==============================] - 0s 382ms/step - loss: 0.0400 - acc: 0.9857 - val_loss: 0.3726 - val_ac
c: 0.5100
Epoch 14/100
1/1 [==============================] - 0s 284ms/step - loss: 0.0378 - acc: 0.9929 - val_loss: 0.3877 - val_ac
c: 0.5200
Epoch 15/100
1/1 [==============================] - 0s 261ms/step - loss: 0.0364 - acc: 0.9929 - val_loss: 0.4196 - val_ac
c: 0.4940
Epoch 16/100
1/1 [==============================] - 0s 261ms/step - loss: 0.0342 - acc: 0.9929 - val_loss: 0.4306 - val_ac
c: 0.4960
Epoch 17/100
1/1 [==============================] - 0s 250ms/step - loss: 0.0300 - acc: 1.0000 - val_loss: 0.4248 - val_ac
c: 0.5000
Epoch 18/100
1/1 [==============================] - 0s 264ms/step - loss: 0.0288 - acc: 0.9857 - val_loss: 0.4351 - val_ac
c: 0.5120
Epoch 19/100
1/1 [==============================] - 0s 273ms/step - loss: 0.0263 - acc: 0.9786 - val_loss: 0.4404 - val_ac
c: 0.5240
Epoch 20/100
1/1 [==============================] - 0s 298ms/step - loss: 0.0228 - acc: 1.0000 - val_loss: 0.4463 - val_ac
c: 0.5380
Epoch 21/100
1/1 [==============================] - 0s 272ms/step - loss: 0.0252 - acc: 0.9786 - val_loss: 0.4559 - val_ac
c: 0.5160
Epoch 22/100
1/1 [==============================] - 0s 269ms/step - loss: 0.0246 - acc: 0.9714 - val_loss: 0.4378 - val_ac
c: 0.4940
Epoch 23/100
1/1 [==============================] - 0s 257ms/step - loss: 0.0247 - acc: 0.9571 - val_loss: 0.4218 - val_ac
c: 0.4980
```

```
Epoch 24/100
1/1 [==============================] - 0s 297ms/step - loss: 0.0195 - acc: 0.9929 - val_loss: 0.3978 - val_ac
c: 0.5220
Epoch 25/100
1/1 [==============================] - 0s 278ms/step - loss: 0.0247 - acc: 0.9571 - val_loss: 0.3813 - val_ac
c: 0.5020
Epoch 26/100
1/1 [==============================] - 0s 254ms/step - loss: 0.0214 - acc: 0.9857 - val_loss: 0.3844 - val_ac
c: 0.4860
Epoch 27/100
1/1 [==============================] - 0s 286ms/step - loss: 0.0241 - acc: 0.9500 - val_loss: 0.3702 - val_ac
c: 0.4980
Epoch 28/100
1/1 [==============================] - 0s 306ms/step - loss: 0.0279 - acc: 0.9500 - val_loss: 0.3631 - val_ac
c: 0.4920
Epoch 29/100
1/1 [==============================] - 0s 273ms/step - loss: 0.0223 - acc: 0.9571 - val_loss: 0.3744 - val_ac
c: 0.4660
Epoch 30/100
1/1 [==============================] - 0s 270ms/step - loss: 0.0210 - acc: 0.9929 - val_loss: 0.3902 - val_ac
c: 0.4660
Epoch 31/100
1/1 [==============================] - 0s 253ms/step - loss: 0.0251 - acc: 0.9714 - val_loss: 0.3881 - val_ac
c: 0.4780
Epoch 32/100
1/1 [==============================] - 0s 244ms/step - loss: 0.0243 - acc: 0.9714 - val_loss: 0.3721 - val_ac
c: 0.4960
Epoch 33/100
1/1 [==============================] - 0s 236ms/step - loss: 0.0240 - acc: 0.9929 - val_loss: 0.3567 - val_ac
c: 0.5040
Epoch 34/100
1/1 [==============================] - 0s 248ms/step - loss: 0.0246 - acc: 0.9786 - val_loss: 0.3403 - val_ac
c: 0.5260
Epoch 35/100
1/1 [==============================] - 0s 345ms/step - loss: 0.0255 - acc: 0.9857 - val_loss: 0.3351 - val_ac
c: 0.5160
Epoch 36/100
1/1 [==============================] - 0s 226ms/step - loss: 0.0250 - acc: 0.9857 - val_loss: 0.3379 - val_ac
c: 0.5040
Epoch 37/100
1/1 [==============================] - 0s 270ms/step - loss: 0.0250 - acc: 1.0000 - val_loss: 0.3394 - val_ac
c: 0.4960
Epoch 38/100
1/1 [==============================] - 0s 278ms/step - loss: 0.0248 - acc: 1.0000 - val_loss: 0.3366 - val_ac
c: 0.5000
Epoch 39/100
1/1 [==============================] - 0s 238ms/step - loss: 0.0248 - acc: 1.0000 - val_loss: 0.3361 - val_ac
c: 0.5040
Epoch 40/100
1/1 [==============================] - 0s 247ms/step - loss: 0.0276 - acc: 0.9714 - val_loss: 0.3291 - val_ac
c: 0.5240
Epoch 41/100
1/1 [==============================] - 0s 236ms/step - loss: 0.0241 - acc: 1.0000 - val_loss: 0.3213 - val_ac
c: 0.5380
Epoch 42/100
1/1 [==============================] - 0s 215ms/step - loss: 0.0238 - acc: 1.0000 - val_loss: 0.3158 - val_ac
c: 0.5500
Epoch 43/100
1/1 [==============================] - 0s 195ms/step - loss: 0.0241 - acc: 0.9929 - val_loss: 0.3117 - val_ac
c: 0.5540
Epoch 44/100
1/1 [==============================] - 0s 177ms/step - loss: 0.0230 - acc: 0.9929 - val_loss: 0.3077 - val_ac
c: 0.5680
Epoch 45/100
1/1 [==============================] - 0s 177ms/step - loss: 0.0222 - acc: 1.0000 - val_loss: 0.3065 - val_ac
c: 0.5820
Epoch 46/100
1/1 [==============================] - 0s 197ms/step - loss: 0.0214 - acc: 1.0000 - val_loss: 0.3055 - val_ac
c: 0.5800
Epoch 47/100
1/1 [==============================] - 0s 194ms/step - loss: 0.0209 - acc: 0.9929 - val_loss: 0.3061 - val_ac
```

```
c: 0.5740
Epoch 48/100
1/1 [==============================] - 0s 206ms/step - loss: 0.0198 - acc: 1.0000 - val_loss: 0.3068 - val_ac
c: 0.5680
Epoch 49/100
1/1 [==============================] - 0s 188ms/step - loss: 0.0201 - acc: 0.9929 - val_loss: 0.3084 - val_ac
c: 0.5460
Epoch 50/100
1/1 [==============================] - 0s 182ms/step - loss: 0.0190 - acc: 1.0000 - val_loss: 0.3150 - val_ac
c: 0.5300
Epoch 51/100
1/1 [==============================] - 0s 187ms/step - loss: 0.0187 - acc: 0.9929 - val_loss: 0.3239 - val_ac
c: 0.5200
Epoch 52/100
1/1 [==============================] - 0s 179ms/step - loss: 0.0169 - acc: 1.0000 - val_loss: 0.3308 - val_ac
c: 0.5080
Epoch 53/100
1/1 [==============================] - 0s 196ms/step - loss: 0.0183 - acc: 0.9714 - val_loss: 0.3216 - val_ac
c: 0.5200
Epoch 54/100
1/1 [==============================] - 0s 199ms/step - loss: 0.0159 - acc: 1.0000 - val_loss: 0.3115 - val_ac
c: 0.5320
Epoch 55/100
1/1 [==============================] - 0s 184ms/step - loss: 0.0162 - acc: 0.9929 - val_loss: 0.3083 - val_ac
c: 0.5320
Epoch 56/100
1/1 [==============================] - 0s 173ms/step - loss: 0.0156 - acc: 0.9929 - val_loss: 0.3087 - val_ac
c: 0.5340
Epoch 57/100
1/1 [==============================] - 0s 188ms/step - loss: 0.0150 - acc: 0.9929 - val_loss: 0.3097 - val_ac
c: 0.5280
```

Out[ ]:

```
<tensorflow.python.keras.callbacks.History at 0x130ad831f48>
```

In [ ]:

```
# Evaluate model
y_pred = model_fnn.predict(X_te)
report = classification_report(np.argmax(y_te,axis=1), np.argmax(y_pred,axis=1), target_names=classes)
print('FCNN Classification Report: \n {}'.format(report))
```

```
FCNN Classification Report: 
                         precision    recall  f1-score   support

            Case_Based       0.53      0.48      0.50       114
    Genetic_Algorithms       0.59      0.83      0.69       156
       Neural_Networks       0.72      0.47      0.57       290
 Probabilistic_Methods       0.76      0.47      0.58       172
Reinforcement_Learning       0.43      0.59      0.50        85
         Rule_Learning       0.38      0.72      0.49        60
                Theory       0.41      0.50      0.45       123

              accuracy                           0.56      1000
             macro avg       0.54      0.58      0.54      1000
          weighted avg       0.60      0.56      0.56      1000
```

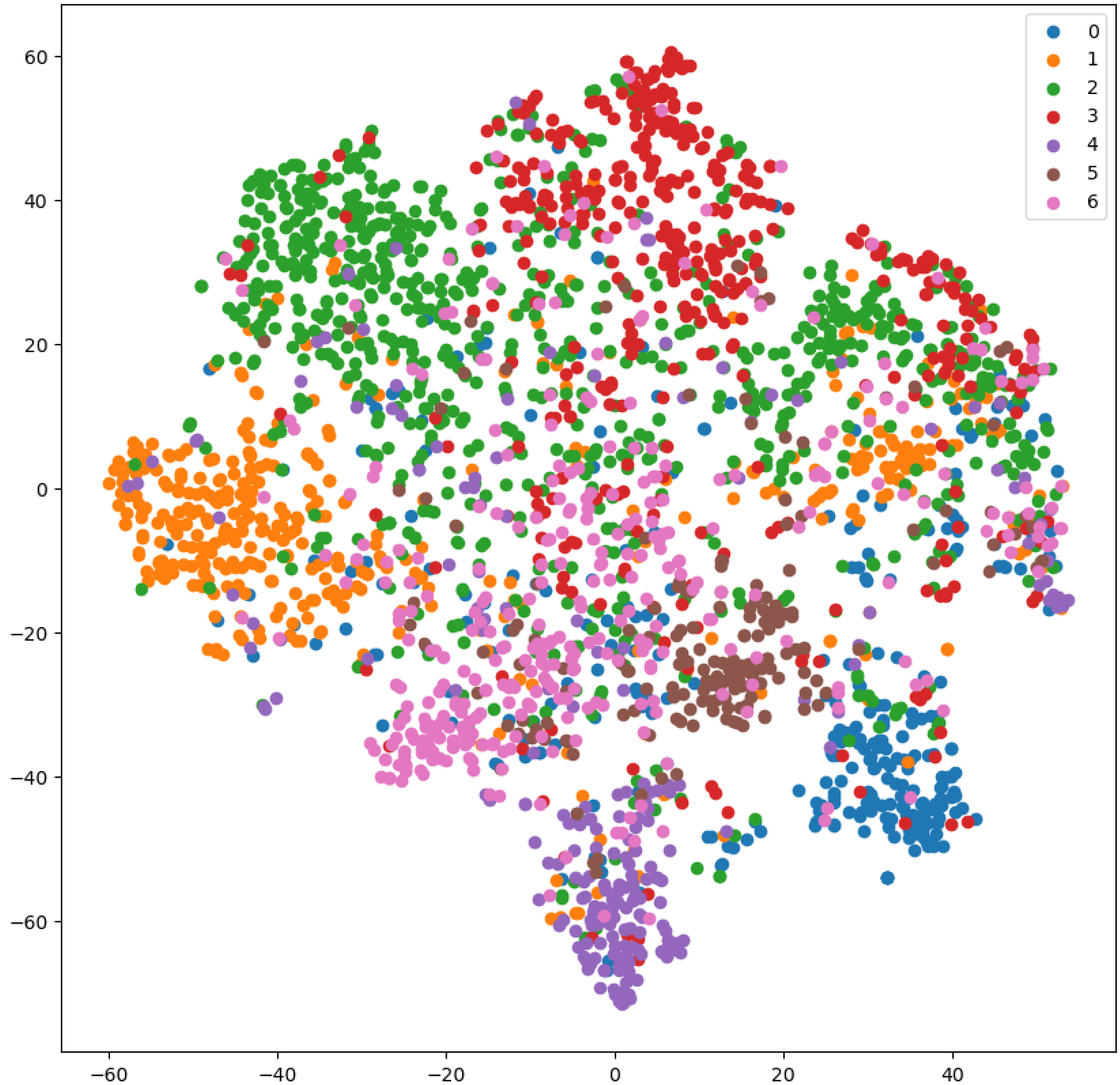
## Get hidden layer representation for FNN

In [ ]:

```
layer_outputs = [layer.output for layer in model_fnn.layers]
activation_model = Model(inputs=model_fnn.input, outputs=layer_outputs)
activations = activation_model.predict([X])
```

In [ ]:

```
x_tsne = TSNE(n_components=2).fit_transform(activations[3])
plot_tSNE(labels_encoded,x_tsne)
```




In [ ]:
```
### END OF NOTEBOOK ###
```