TP3 : Graph Neural Networks (GNNs)

L'objectif de ce TP est de construire notre propre réseau de neurone de graphes (GNN) en utilisant la bibliothèque ***PyG***. Nous allons appliquer notre modèle sur deux ensembles de données d'Open Graph Benchmark (***OGB***). Ces deux ensembles de données seront utilisés pour évaluer notre modèle sur deux tâches d'apprentissage sur graphes. La première est la prédiction des propriétés de nœuds (classification de nœuds) et la deuxième est la prédiction des propriétés de graphes en entier (classifications de graphes).

Avant de commencer le TP, assurez-vous que vous avez installé tous les modules nécessaires (voir le fichier Requirements.txt). En cas de problème de mémoire insuffisante, vous pouvez augmenter la limite de taille mémoire utilisée par votre notebook (Jupyter) à l'aide de cette instruction :

```
jupyter notebook --NotebookApp.max_buffer_size=99999999999999
```

Télécharger le notebook *TP3_GNNs* pour répondre aux questions.

Partie 1 :

Nous commençons par voir comment ***PyG*** stocke les graphes en tenseurs. Ensuite, nous verrons un petit exemple d'utilisation du package ***OGB***.

1. PyG Dataset et Data :

PyG a deux classes pour stocker et transformer les graphes en tenseurs. La première est `torch_geometric.datasets` , qui contient une variété d'ensemble de données graphes. La deuxième classe `torch_geometric.data` qui permet de manipuler des graphes en tant que tenseurs.

1- Exécuter la première cellule pour télécharger l'ensemble de données ENZYMES.
2- Quel est le nombre de classes et de features de cet ensemble de données ?

Chaque ensemble de données ***PyG*** contient généralement une liste d'objets `torch_geometric.data.Data`. Chaque objet `torch_geometric.data.Data` représente généralement un graphe. Pour plus d'informations, veuillez-vous référer à la documentation :
https://pytorch-geometric.readthedocs.io/en/latest/modules/data.html#torch_geometric.data.Data

3- Quelle est la classe (label) du graphe d'index 50 dans l'ensemble de données ENZYMES ?
4- Quelle est le nombre d'arêtes du graphe d'index 100 ?

2. Open Graph Benchmark (OGB) :

L'***OGB*** est une collection des ensembles de données réels de grande échelle utilisés pour l'apprentissage automatique sur les graphes. Ces ensembles de données peuvent être téléchargés, traités, est divisés (entrainement, validation et test) automatiquement en utilisant l'***OGB*** Data Loader. L'***OGB*** permet également d'évaluer les performances des différents modèles d'apprentissage d'une manière unifiée. L'***OGB*** prend également en charge les ensembles de données et les données ***PyG***. Nous allons jeter un coup d'œil sur l'ensemble de données `ogbn-arxiv.`

5- Exécuter la cellule pour charger l'ensemble de données et le transformer en Tenseur.
6- Quel est le nombre de features des nœuds du graphe ogbn-arxiv ?

Partie 2 (Les réseaux de neurones de graphes) :

Dans cette partie, nous allons construire notre premier réseau de neurones de graphes (***GNN***) en utilisant ***PyG*** et nous allons l'appliquer à la tache de prédiction de propriété de nœuds (classification de nœuds). Notre modèle sera basé sur l'opérateur de convolution de graphes GCN (Kipf et al.2017)[1]. On ne vous demande pas d'implémenter cet opérateur. Vous devez utiliser directement la couche `GCNConv` de ***PyG***.

1- Exécuter les deux premières cellules de la partie 2 pour importer les bibliothèques nécessaires et charger le graphe `ogbn-arxiv`.

L'ensemble de données `ogbn-arxiv` est un graphe orienté, représentant le réseau de citations entre tous les articles indexés `arXiv` d'informatique. Chaque nœud est un article arXiv et chaque arête dirigée indique qu'un article en cite un autre. Chaque article est accompagné d'un vecteur de caractéristiques (features) à 128 dimensions obtenu en faisant la moyenne des plongements de mots dans son titre et son résumé. Les plongements des mots sont calculés en exécutant le modèle Skip-gram.
Notre objectif est de prédire les 40 domaines des articles d'informatique d'arXiv, par exemple intelligence artificielle, génie logiciel et système d'exploitation, qui sont déterminés manuellement (c'est-à-dire étiquetés) par les auteurs des articles et les modérateurs d'arXiv. Le volume des publications scientifiques ayant doublé tous les 12 ans au cours du siècle dernier, il est pratiquement important de classer automatiquement les domaines et les sujets de chaque publication. Formellement, la tâche consiste à prédire les catégories primaires des articles arXiv, ce qui est formulé comme un problème de classification à 40 classes.

Pour cela nous allons implémenter le réseau de neurones de graphes illustré par la figure 1. Ce modèle est constitué des couches suivantes :

[1] https://arxiv.org/pdf/1609.02907.pdf

- `num_layers` couches `GCNConv` de ***PyG***. La couche `GCNConv` est un opérateur de convolution de graphes proposé dans le papier [2].
- `num_layers-1` couches de normalisation `BatchNorm1d`.
- Une couche `LogSoftmax`.

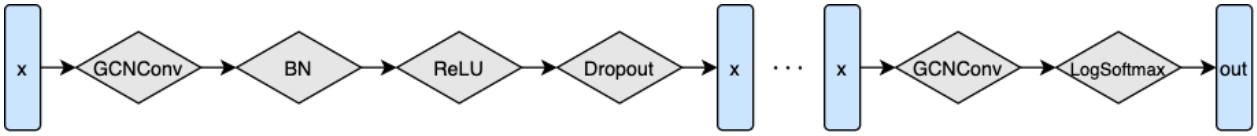

Figure 1. Le modèle GCN à implémenter

2- Compléter la classe `GCN`.
3- Compléter les fonctions `train`, `forward` et `test`.
4- Tester votre modèle, quelle est l'accuracy que vous avez obtenu ?

Partie 3 (GNN pour la classification de graphes) :

Dans cette partie, nous allons créer un réseau de neurones de graphes GNN pour la prédiction des propriétés des graphes (classification des graphes)

1- Exécuter les deux premières cellules de la partie 3 pour importer les bibliothèques nécessaires et charger le graphe `ogbn-molhiv`.

L'ensemble de données `ogbn-molhiv` est un ensemble de données de classification de graphes. Chaque graphe dans cet ensemble de données représente une molécule chimique. Les nœuds des graphes sont des atomes et les arêtes sont des liaisons chimiques. Chaque nœud est accompagné d'un vecteur de 9-cases représentant des propriétés physico-chimique.

Notre objectif est de prédire si une molécule inhibe ou non la réplication du virus VIH (classification binaire).

Nous allons réutiliser le modèle précèdent pour générer les plongements des nœuds et utiliser un pooling global sur les nœuds pour prédire les propriétés du graphe en entier (molécule).

2- Compléter la classe `GCN_Graph`.
3- Compléter les fonctions `train` et `forward`.
4- Tester votre modèle, quelle est l'accuracy que vous avez obtenu ?
5- Effectuer des tests avec des deux autres couches de global pooling.

[2] https://arxiv.org/pdf/1609.02907.pdf