Etat de l'art des réseaux de convolutions de graphes et leurs nuances.

Cléa HAN - Léa TRIQUET - Adrien ZABBAN

27 octobre 2023

1 Introduction

Les méthodes que l'ont va aborder :
— Graph Convolusional Networks [1] (GCN)
— Simplifying Graph Convolutional Networks [2] (SGC)
— Graph Isomorphism Network [3] (GIN)

2 Les différentes méthodes

2.1 Graph Convolusional Networks (GCN)

Un Graph Convolution Network (GCN) est un type de réseau de neurones artificiels conçu pour traiter des données structurées sous forme de graphes. Ils utilisent des opérations de convolution spécialement adaptées aux graphes pour propager l'information entre les nœuds du graphe, ce qui leur permet de capturer les relations et les dépendances entre les entités interconnectées. L'équation 1 représente la règle de propagation de la couche l à la couche $l+1$.

$$H^{(l-1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}) \tag{1}$$

où $\tilde{A} = A + I_n$ est la matrice d'adjacence au quelle on ajoute des connections sur les noeuds vers eux-mêmes. $\tilde{D_{ii}} = \sum_j \tilde{A_{ij}}$ qui est une matrice diagonale des degrés des noeuds du graphe. $W^{(l)}$ sont les poids de la couche l et $H^{(l)}$ est la sortie de la couche $l-1$, $H^{(0)} = X$. σ est une fonction d'activation.

Par exemple, un petit réseaux de convolutions de graphes pourait suivre l'équation 2

$$Z_{GCN} = \text{softmax}(\hat{A}\,\text{ReLU}(\hat{A}XW^{(0)})W^{(1)}) \tag{2}$$

où $\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$

2.2 Simplifying Graph Convolutional Networks (SGC)

Les modèles SGCs sont construits à partir d'une linéarisation des GCNs en faisant l'hypothèse que les non linéarités principalement apportées par les ReLU entre les différentes couches d'un GCN n'apportent pas d'informations significatives. Ainsi, les modèles SGCs retirent ces non linéarités et conservent

Modèles	GCN	SGC	GIN
Cross Entropy	2.86	1.85	1.75
Accuracy	0.20	0.16	0.38

TABLE 1 – Résultat sur la base de données de teste

uniquement un Softmax en dernière couche, afin de pouvoir obtenir des résultats sous forme d'une distribution de probabilités.

Nous pouvons modéliser le SGC sous la forme de l'expression simplifiée suivante l'équiation 3

$$Z_{SGC} = \text{softmax}(\hat{A}^K XW) \tag{3}$$

où $\hat{A}^K$ représentant les K multiplications successives avec $\hat{A}$ notre matrice d'adjacence normalisée, définie dans la section 2.1, X est les entrées et W est la matrice réunissant l'ensemble des poids utilisés dans le modèle tel que : $W = W^{(1)}W^{(2)}\dots W^{(K)}$. La SGC équivaut à une une composante d'extraction de caractéristiques : $\tilde{X} = \hat{A}^K X$, ce qui nécessite aucun poids. Puis, cela est suivi d'une régression logistique $Z_{SGC} = \text{softmax}(\tilde{X}W)$. Ainsi la première étape équivaut à du prétraitement des caractéristiques de nos données X, car aucun poids n'est appliqué sur nos données. Ainsi, l'entraînement du SGC s'apparente en réalité à une régression logistique multi classe sur des données prétraités. D'autre part, l'entraînement d'une régression logistique est un problème d'optimisation convexe bien documenté, ainsi ses performances seront relativement meilleures que celles du GCN qui contient des non linéarités. De plus, l'entraînement du SGC sera naturellement plus rapide. D'un point de vue "graphe", la SGC correspond à un filtre fixe sur le domaine du graphe spectral. Si nous ajoutons des boucles dans le graphe original, l'application du SGC permet de réduire la taille du spectre graphique grâce à l'astuce de renormalisation présentée par Kipf & Welling [1]. La SGC agit comme un filtre passe bas qui lasse les caractéristiques du graphes, ainsi les nœuds voisins vont avoir tendance à partager des représentations et donc des prédictions similaires.

2.3 Graph Isomorphism Network (GIN)

3 Entraînements

Nous avons entraîné des petits réseaux de neurones utilisant les GCN, SGC et GIN. La Figure 1 représente respectivement l'entrainement des modèles GCN, SGC et GIN.

4 Résultats et conclusion

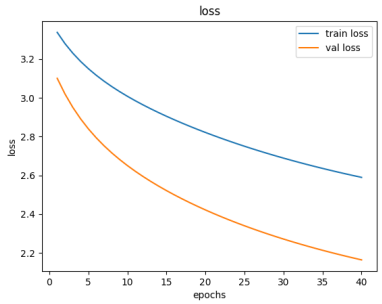
Nous voyons sur la Table 1, les résultats des modèles sur la base de données de teste.

Références

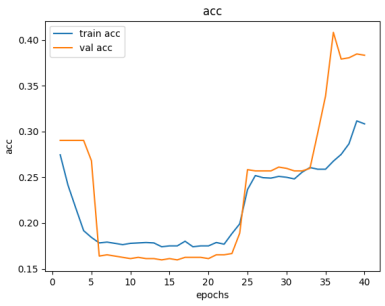
[1] T. N. Kipf et M. Welling, « Semi-Supervised Classification with Graph Convolutional Networks, » *CoRR*, t. abs/1609.02907, 2016. arXiv : 1609.02907. adresse : http://arxiv.org/abs/1609.02907.

[2] F. Wu, T. Zhang, A. H. S. Jr., C. Fifty, T. Yu et K. Q. Weinberger, « Simplifying Graph Convolutional Networks, » *CoRR*, t. abs/1902.07153, 2019. arXiv : 1902.07153. adresse : http://arxiv.org/abs/1902.07153.

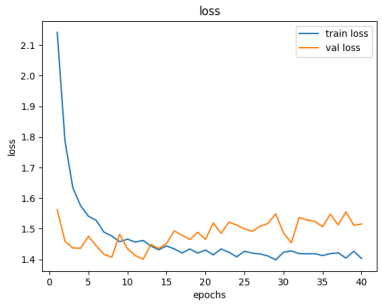
[3] K. Xu, W. Hu, J. Leskovec et S. Jegelka, « How Powerful are Graph Neural Networks ? » *CoRR*, t. abs/1810.00826, 2018. arXiv : 1810.00826. adresse : http://arxiv.org/abs/1810.00826.



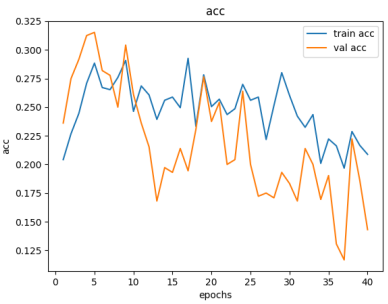
(a) GCN : Cross Entropy Loss



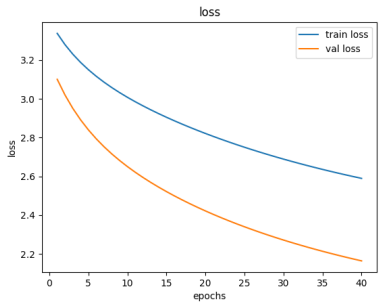
(b) GCN : accuracy



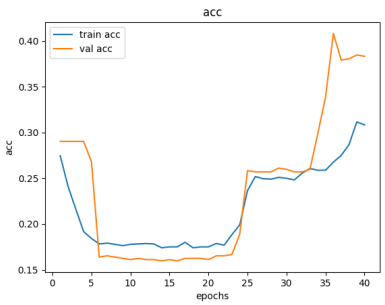
(c) SGC : Cross Entropy Loss



(d) SGC : accuracy



(e) GIN : Cross Entropy Loss



(f) GIN : accuracy

FIGURE 1 – Entraînement des modèles GCN, SGC et GIN