Amélioration de l'architecture GAT par la prise en compte de la courbure du graphe

Hugo Attali *, Adrien Guille *
Stephane Chretien *

* Université de Lyon, Lyon 2, ERIC UR 3083
prénom.nom@univ-lyon2.fr,

Résumé. Bien que les réseaux de neurones opérant sur des graphes comme le GCN ou le GAT, soient très utilisés, il est établi qu'ils souffrent d'un de goulot d'étranglement qui limite leur efficacité. Récemment, il a été montré que le phénomène de goulot d'étranglement provient de certaines zones des graphes, que l'on peut identifier par une mesure de courbure des arêtes. Tandis qu'une solution consiste à modifier le graphe dans ces zones en ajoutant et en supprimant des arêtes nous proposons de modifier le mécanisme d'attention du GAT pour moduler les poids d'attention selon la courbure des arêtes. Les expériences menées sur différents jeux de données montrent que notre méthode s'avère plus efficace, moins couteuse et améliore la méthode originale GAT dans la classification des nœuds.

1 Introduction

La recherche d'information nécessite une bonne représentation des documents. En plus des informations textuelles, représenter un corpus par un graphe permet de prendre en compte les liens entre documents et ainsi d'améliorer la qualité des représentations. Récemment, l'architecture en graphe est ancrée dans le domaine de l'apprentissage profond et résout efficacement de nombreuses tâches du Traitement Automatique de la Langue comme la classification de document (Guille et Attali, 2022a,b, 2023) et de classification de nœuds (Brochier et al., 2019, 2020). La majorité des réseaux de neurones opérant sur des graphes (GNNs) est basée sur le passage de messages, où l'information entre les nœuds voisins est propagée à travers le graphe comme GCN (Kipf et Welling, 2017) ou GAT (Veličković et al., 2018). Les GNNs peuvent être confrontés à un certain nombre de problèmes en particulier de moins bonnes performances en environnement hétérophile (Zhu et al., 2020), lorsque les nœuds voisins ne sont pas similaires ou du moins donnent des informations très différentes. Le phénomène de surlissage arrive lorsque le passage de message est effectué de manière excessive, dans ce cas toutes les représentations des sommets du graphe ont tendance à être similaires (Oono et Suzuki, 2020)(Cai et Wang, 2020). Ce phénomène provoque une dégradation conséquente des résultats(Kipf et Welling, 2017)(Qimai Li, 2018) . Un autre problème est le phénomène de goulot d'étranglement dans un graphe qui se produit lorsque deux parties d'un graphe sont connectées par très peu

d'arêtes. Ainsi, lors du passage d'un message, l'information devra être condensée, entraînant une perte d'information. Ce dernier problème est dû à la topologie du graphe et plus précisément à la courbure négative (cf figure 1) de certaines arêtes du graphe (Topping et al., 2022) . Dans cet article, nous proposons de modifier le mécanisme d'attention du GAT afin de prendre en compte la courbure des arêtes pour atténuer le problème de goulot d'étranglement.

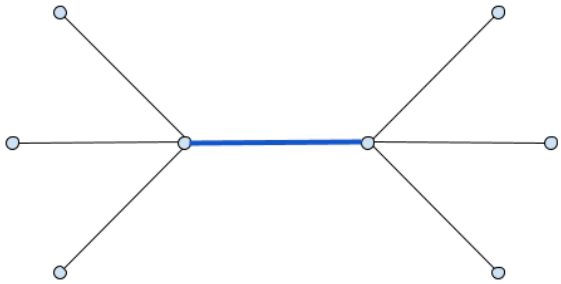

FIG. 1 – *En bleu un exemple d'arête à courbure négative.*

Reproductibilité. Le code permettant de reproduire les résultats de l'article est disponible. [1]

2 Travaux connexes

Réseau de neurones opérant sur des graphes Le succès de l'apprentissage profond dans le domaine euclidien a suscité un grand intérêt pour la généralisation des réseaux de neurones aux domaines non euclidiens, comme les graphes. L'idée principale des GNNs est d'agréger les attributs des nœuds voisins pour enrichir les représentations des nœuds. Les GNNs mettent à jour leurs représentations de différentes manières. Le GCN (Kipf et Welling, 2017) met à jour la représentation des nœuds en utilisant les degrés des nœuds voisins.

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

où $\mathbf{D}$ est la matrice de degrés, $\tilde{\mathbf{D}} = \mathbf{D} + \mathbf{I}$, $\mathbf{A}$ la matrice d'adjacence, $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, $\mathbf{W}^{(l+1)}$ représente une matrice de poids à la l+1-éme couche, σ est une fonction d'activation non linéaire.

L'architecture GAT (Veličković et al., 2018) pondère l'importance des attributs des sommets voisins avec un mécanisme d'attention α où α_{ij} est l'importance attribuée au nœud j par le nœud i. Si le nœud j n'est pas un voisin du nœud i $\alpha_{ij} = 0$, sinon il est calculé comme suit :

$$\alpha_{ij}^{(l+1)} = \frac{\exp\left(\text{LeakyReLU}(z^{(l+1)} \cdot [h_i^{(l)}\mathbf{W}^{(l+1)} || h_j^{(l)}\mathbf{W}^{(l+1)}])\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\text{LeakyReLU}(z^{(l+1)} \cdot [h_i^{(l)}\mathbf{W}^{(l+1)} || h_k^{(l)}\mathbf{W}^{(l+1)}])\right)}, \tag{2}$$

1. Code disponible sur : `https://github.com/Hugo-Attali/Curvature-GAT`

où $||$ représente la concaténation et $\mathcal{N}_i$ est l'ensemble des voisins du sommet i. Ce score est paramétré par $z^{(l+1)}$ et $\mathbf{W}^{(l+1)}$, respectivement un vecteur de poids et une transformation linéaire. L'activation LeakyReLU est calculée avec une pente négative de 0,2. Enfin, les mises à jour de la représentation sont calculées en fonction des scores d'attention.

$$\mathbf{H}^{(l+1)} = \sigma\left(\boldsymbol{\alpha}\mathbf{H}^{(l)}\mathbf{W}^{(l+1)}\right). \tag{3}$$

Ces différentes méthodes n'exploitent pas de manière optimale la structure du graphe. (Wang et al., 2021) montre que la prise en compte des degrés du graphe dans le mécanisme d'attention améliore les résultats dans la tâche de classification de nœuds. Les auteurs modifient la matrice des poids d'attention telle que la mise à jour de la représentation est :

$$\mathbf{H}^{(l+1)} = \sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{\alpha}_{\boldsymbol{D}}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}^{(l+1)}), \tag{4}$$

où $\boldsymbol{\alpha_D}$ = $\boldsymbol{\alpha}$ * D, avec $\boldsymbol{\alpha}$ la matrice de poids d'attention originale (Veličković et al., 2018).

Courbure dans un graphe Les principales publications sur la courbure des arêtes dans un graphe sont les travaux de (Forman, 2003) et (Ollivier, 2007). Dans cet article, nous nous concentrons sur la courbure d'Ollivier, plus précise, basée sur la théorie du transport optimal. Nous définissons une distribution de probabilité μ_i sur le graphe de maniere à appliquer à chaque nœud i une mesure de probabilité π_i :

$$\mu_i = \begin{cases} \pi_i & \text{si } j = i \\ (1-\pi_i)/k & \text{si } j \in \mathcal{N}(i) \\ 0 & \text{sinon} \end{cases}, \tag{5}$$

Où $k$représente le degré du nœud i. En suivant les travaux précédents (Ni et al., 2015) (Ni et al., 2018) (Ye et al., 2019) nous choisissons π_i = 0,5. Nous introduisons alors la distance de Wasserstain de l'ordre 1, $W_1(i,j)$, correspondant au transport optimal des masses de probabilités des voisins de i vers les voisins de j :

$$W_1(\mu_i, \mu_j) = \min_{\pi \in \Pi(\mu_i, \mu_j)} \int_{(u,v)} d(u,v) d\pi(u,v). \tag{6}$$

La courbure d'Ollivier c_{ij} d'une arête e_{ij} peut être définie comme :

$$C_{ij} = 1 - \frac{W_1(\mu_i, \mu_j)}{\text{dist}(i,j)}, \tag{7}$$

où dist(i,j) est la longueur du plus court chemin entre le nœud i et le nœud j.

(Topping et al., 2022) montre que la courbure du graphe, notamment les arêtes à courbure négative, joue un rôle important dans le phénomène de goulot d'étranglement et perturbe la bonne transmission du passage de message. Il propose donc de modifier la topologie du graphe pour atténuer ce problème (cf figure 2).

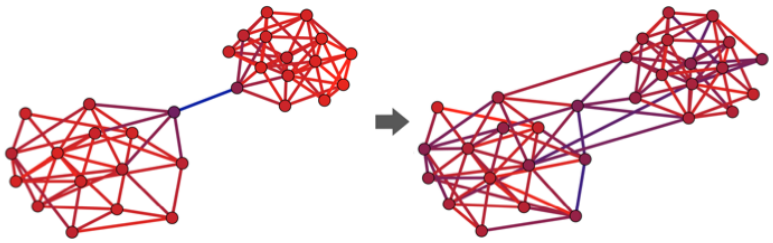

FIG. 2 – *Changement de la topologie du graphe pour éviter le phénomène de goulot d'étranglement. Figure tirée de (Topping et al., 2022)*

3 Proposition

Nous proposons de modifier le mécanisme d'attention du GAT (Veličković et al., 2018) en intégrant la courbure du graphe pour atténuer le phénomène de goulot d'étranglement. Dans cet objectif, nous travaillons avec la courbure d'Ollivier. L'interet de travailler avec la courbure d'Ollivier est que nous sommes capables de la calculer en amont, et donc de l'introduire comme une entrée de notre réseau, cela ne rajoute donc aucun paramètre à notre modèle. Nous introduisons une matrice $C \in \mathbb{R}^{N \times N}$ où N représente le nombre de nœuds du graphe. Ainsi, C_{ij} est la courbure de l'arête entre les nœuds i et j. Inspirés par (Topping et al., 2022), nous appliquons une fonction softmax de manière à accorder plus d'importance aux arêtes à courbure négative responsables du phénomène de goulot d'étranglement. Par conséquent, nous définissons un nouveau mécanisme d'attention α' tel que :

$$\alpha'_{ij} = \frac{(\alpha_{ij} + \text{softmax}(C_{ij}) \times Q)}{Q + 1} \tag{8}$$

où α_{ij} est le mécanisme d'attention original défini dans l'équation 2 et Q $\in \mathbb{R}^+$, initialisé à $\frac{1}{4}$, est appris par notre réseau qui attribue l'importance de la courbure dans le mécanisme d'attention. Enfin, nous mettons à jour la représentation $H'^{(l)}$ en remplaçant l'équation 3 par :

$$\mathbf{H}^{(l+1)} = \sigma\left(\boldsymbol{\alpha'}\mathbf{H}^{(l)}\mathbf{W}^{(l+1)}\right), \tag{9}$$

4 Expériences

Nous réalisons des expériences sur 5 jeux de données divers sur une tâche de classification de nœuds.

— **Réseaux de publication scientifique** [2] : Les jeux de données Cora, Citeseer et Pubmed (Sen et al., 2008) décrivent les citations de publications scientifiques. Chaque publication est décrite par un sac de mots du résumé de la publication. Les classes sont les catégories de publications.

2. https://github.com/calciver/Graph-Attention-Networks/tree/master/data

— **Amazon** [3] : Amazon computers et Amazon photos sont un extrait du graphe d'achat Amazon (McAuley et al., 2015), où les nœuds représentent les biens, les arêtes indiquent si deux biens sont fréquemment achetés ensemble. Les attributs sont les sacs de mots des descriptions des produits. Les classes sont les catégories de produits.

Pour les expériences, nous appliquons la même stratégie d'apprentissage que celle présentée dans (Shchur et al., 2018)

TAB. 1 – *Description des jeux de données.*

	Cora	**Citeseer**	**Pubmed**	**Amazon photo**	**Amazon computers**
# Nodes	2 708	3 312	19 717	7 650	13,752
# Edges	5 429	4 715	44 324	143 663	287 209
# Classes	7	6	3	8	10
# Train Nodes	140	120	60	160	200

Nous comparons notre méthode avec 4 méthodes basées sur les GNNs.

— **GCN** : Deux couches avec une taille cachée de 64, un pas d'apprentissage de 0,01 comme présenté dans (Shchur et al., 2018).

— **GAT** [4] (Veličković et al., 2018) : Deux couches GAT. Pour les jeux de données Cora et Citeseer, la première couche comporte 8 têtes d'attention et une tête pour la seconde couche. Pendant l'apprentissage, nous utilisons un taux d'apprentissage de 0.005 et nous appliquons une couche de régularisation L_2 avec λ = 0.0005 avec un dropout de 0.6 comme dans l'article original. Pour les jeux de données Pubmed, Amazon photos et Amazon computers, la première couche a 8 têtes d'attention et la seconde cinq. Pendant l'apprentissage, nous utilisons un taux d'apprentissage de 0,01 et nous appliquons une couche de régularisation L_2 avec λ = 0,0005 et un dropout de 0,5.

— **APPN** [5] (Klicpera et al., 2018) : Deux couches de GCN avec une taille cachée de 64, un pas d'apprentissage de 0,01. Pour la propagation du Pagerank (Page et al., 1999), nous utilisons une probabilité de téléportation de $\gamma = 0,1$ comme dans l'article original.

— **Gat + norm.adj** [6] (Wang et al., 2021) : Même configuration que GAT pour une comparaison juste, avec la matrice d'attention normalisée comme dans l'article original.

— **SRDF** [7] (Topping et al., 2022) : Résultats issus du papier original. Les résultats sur les jeux de données Amazon sont issus de (Sun et al., 2022).

— **Gat + Curvature (notre méthode)** Pour une comparaison juste, nous utilisons notre méthode avec la même configuration que GAT.

Comme (Shchur et al., 2018) nous expérimentons toutes les méthodes avec une patience de 100. Nous exécutons chaque méthode 10 fois sur chaque jeu de données et nous rapportons le taux de réussite moyen.

3. `https://github.com/EdisonLeeeee/GraphData/tree/master/datasets`
4. Code : `https://github.com/PetarV-/GAT`
5. Code : `https://github.com/gasteigerjo/ppnp`
6. Code : `https://github.com/espylapiza/BoT`
7. Code : `https://github.com/jctops/understanding-oversquashing`

TAB. 2 – *Moyenne des scores en test. Meilleur score en gras et deuxième meilleur score en italique.*

	Cora	Citeseer	Pubmed	Photos	Computers
GCN(Kipf et Welling, 2017)	81.5(1.3)	70.9 (1.9)	77.8 (2.9)	91.2 (1.2)	82.6 (2.4)
GAT(Veličković et al., 2018)	*82.9* (0.7)	72.2 (0.7)	78.0 (0.5)	84.6 (17.8)	80.7 (1.2)
APPN(Klicpera et al., 2018)	82.7 (0.7)	**72.9** (1.5)	*78.8* (1.1)	*91.6* (0.7)	80.8 (0.7)
Gat + norm.adj(Wang et al., 2021)	**83.2** (0.8)	72.2 (0.6)	77.8 (1.7)	90.9 (0.6)	*84.2* (1.7)
SRDF(Topping et al., 2022)	82.8 (0.2)	*72.6* (0.2)	**79.1** (0.1)	> 5 jours	> 5 jours
GAT + Curvature	*82.9* (0.8)	*72.6* (0.8)	78.3 (0.4)	**91.7** (1.1)	**86.2** (1.6)
$\Delta_{\mathbf{GAT}}$	**+0.0**	**+0.4**	**+0.3**	**+5.1**	**+5.5**

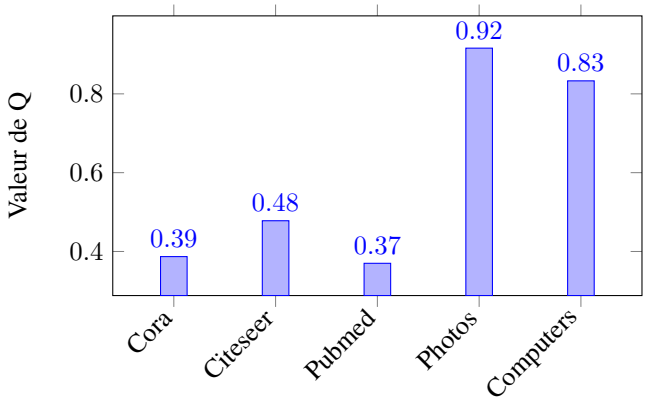

FIG. 3 – *Sur la gauche, la comparaison des résultats entre notre méthode et le GAT original. Sur la droite, la valeur de Q dans l'équation 8 en fonction des jeux de données.*

Nous constatons que notre méthode est meilleure que GAT sur tous les jeux de données et obtient des résultats significativement meilleurs sur les graphes larges et complexes. Pour ces jeux de données le réseau donne beaucoup d'importance à la courbure (cf figure ▯▯) dans le mécanisme d'attention et les améliorations de nos résultats sont plus importants.

5 Conclusion

Les structures en graphe sont omniprésentes pour représenter différents réseaux de documents. Dans cet article, nous montrons que la prise en compte de la structure locale et plus précisément de la courbure d'un graphe dans le mécanisme d'attention apporte des améliorations dans la tâche de classification des nœuds. Apporter une plus grande importance aux arêtes négatives atténue le problème de goulot d'étranglement et évite la perte d'information, d'autant plus que les graphes sont grands et complexes. Dans des travaux futurs, nous aimerions explorer l'utilité de ce type d'architecture pour résoudre des tâches de prédictions de liens et de classifications de documents.

Références

Brochier, R., A. Guille, et J. Velcin (2019). Global vectors for node representations. In *Proceedings of the ACM WWW World Wide Web Conference*.

Brochier, R., A. Guille, et J. Velcin (2020). Inductive document network embedding with topic-word attention. In *Proceddings of the European Conference on Information Retrieval*.

Cai, C. et Y. Wang (2020). A note on over-smoothing for graph neural networks. *Graph Representation Learning*.

Forman, R. (2003). Bochner's method for cell complexes and combinatorial ricci curvature.

Guille, A. et H. Attali (2022a). Classification interprétable de documents à l'aide d'un réseau de neurones opérant sur des graphes. TextMine @ EGC.

Guille, A. et H. Attali (2022b). Document classification with hierarchical graph neural networks. MLG @ ECML-PKDD.

Guille, A. et H. Attali (2023). Classification de documents par un réseau de neurones opérant sur des graphes dans l'espace hyperbolique. In *Actes de la conférence sur l'Extraction et la Gestion de Connaissance à partir des Données*.

Kipf, T. N. et M. Welling (2017). Semi-Supervised Classification with Graph Convolutional Networks. In *Proceedings of the International Conference on Learning Representations*, ICLR.

Klicpera, J., A. Bojchevski, et S. Günnemann (2018). Predict then propagate : Graph neural networks meet personalized pagerank. *arXiv preprint arXiv :1810.05997*.

McAuley, J., C. Targett, Q. Shi, et A. Van Den Hengel (2015). Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pp. 43–52.

Ni, C.-C., Y.-Y. Lin, J. Gao, X. David Gu, et E. Saucan (2015). Ricci curvature of the internet topology. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pp. 2758–2766.

Ni, C.-C., Y.-Y. Lin, J. Gao, et X. Gu (2018). Network alignment by discrete ollivier-ricci flow. In *International Symposium on Graph Drawing and Network Visualization*, pp. 447–462. Springer.

Ollivier, Y. (2007). Ricci curvature of metric spaces. *Comptes Rendus Mathematique 345*(11), 643–646.

Oono, K. et T. Suzuki (2020). Graph neural networks exponentially lose expressive power for node classification. *Proceedings of the International Conference on Learning Representations*.

Page, L., S. Brin, R. Motwani, et T. Winograd (1999). The pagerank citation ranking : Bringing order to the web. Technical report, Stanford InfoLab.

Qimai Li, Zhichao Han, X.-M. W. (2018). Deeper insights into graph convolutional networks for semi-supervised learning.

Sen, P., G. Namata, M. Bilgic, L. Getoor, B. Galligher, et T. Eliassi-Rad (2008). Collective classification in network data. *AI magazine 29*(3), 93–93.

Shchur, O., M. Mumme, A. Bojchevski, et S. Günnemann (2018). Pitfalls of graph neural network evaluation. *arXiv preprint arXiv :1811.05868*.

Sun, Q., J. Li, H. Yuan, X. Fu, H. Peng, C. Ji, Q. Li, et P. S. Yu (2022). Position-aware structure learning for graph topology-imbalance by relieving under-reaching and over-squashing. *CIKM '22 : Proceedings of the 31st ACM International Conference on Information Knowledge Management*.

Topping, J., F. Di Giovanni, B. P. Chamberlain, X. Dong, et M. M. Bronstein (2022). Understanding over-squashing and bottlenecks on graphs via curvature. *Proceedings of the International Conference on Learning Representations*.

Veličković, P., G. Cucurull, A. Casanova, A. Romero, P. Liò, et Y. Bengio (2018). Graph Attention Networks. ICLR.

Wang, Y., J. Jin, W. Zhang, Y. Yu, Z. Zhang, et D. Wipf (2021). Bag of tricks for node classification with graph neural networks. *arXiv preprint arXiv :2103.13355*.

Ye, Z., K. S. Liu, T. Ma, J. Gao, et C. Chen (2019). Curvature graph network. In *International Conference on Learning Representations*.

Zhu, J., Y. Yan, L. Zhao, M. Heimann, L. Akoglu, et D. Koutra (2020). Beyond homophily in graph neural networks : Current limitations and effective designs. *Advances in Neural Information Processing Systems 33*, 7793–7804.

Summary

Over the past few years, graph structures have proven their effectiveness to represent interaction between textual information on many natural language tasks. The new GAT architecture has significantly improved the results in node classification tasks thanks to their attention mechanism based on the features of the vertices. In parallel, recent publications have shown that taking into account the topological aspect of the graph can attenuate some problems such as over-smoothing and the bottleneck phenomenon. In this paper we propose a way to improve GAT by considering the graph curvature in the attention mechanism. Experiments conducted on various datasets show that our method improves the original method GAT and outperforms recent GNNs specialized in node classification.