# Comment réaliser des comparaisons rigoureuses entre différents GNN ?

Séminaire TALia du 11/03/2022

Article

OpenReview

publié dans
ICLR 2020

arXiv > cs > arXiv:1912.09893

Computer Science > Machine Learning

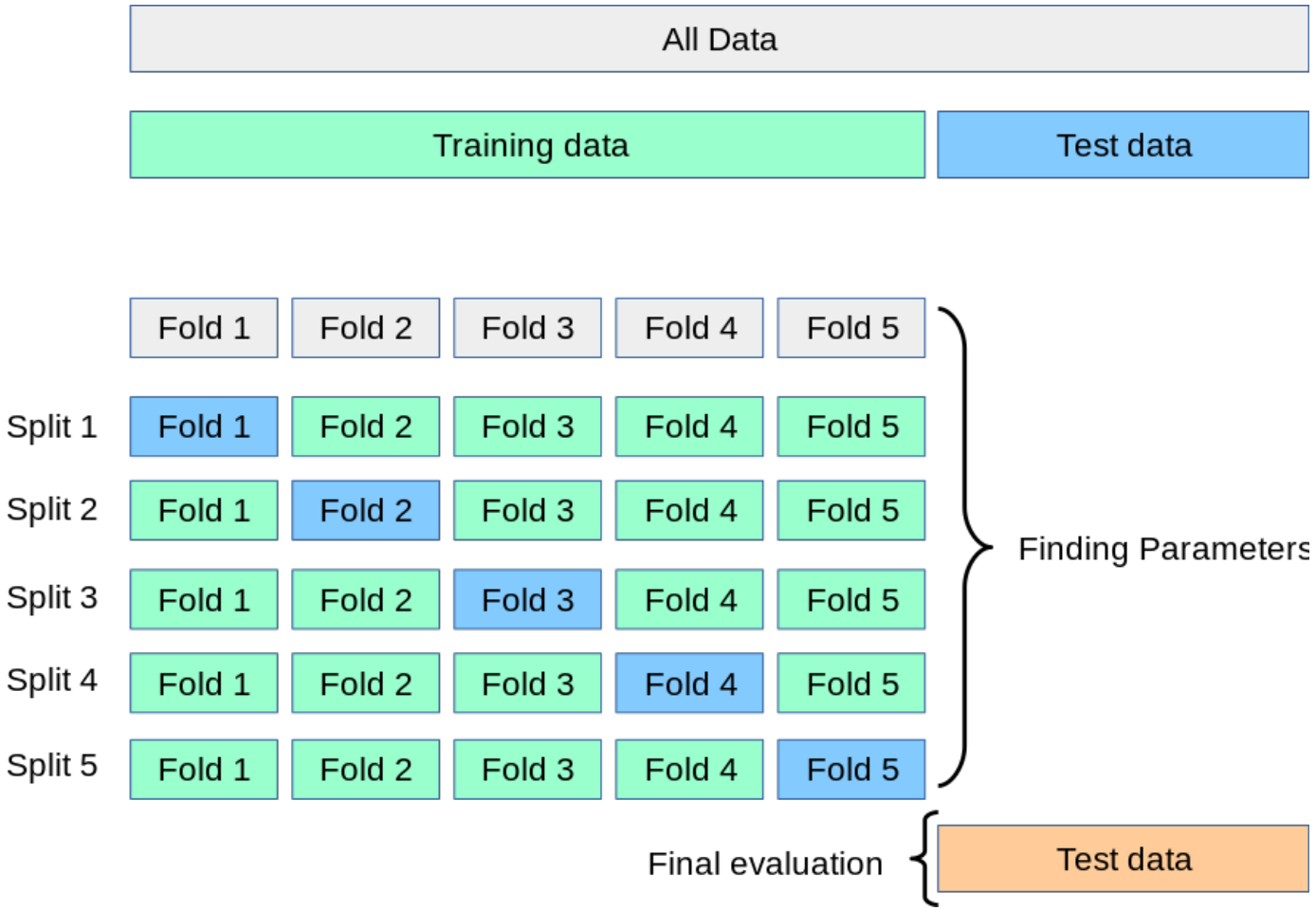
*[Submitted on 20 Dec 2019 (v1), last revised 17 Feb 2022 (this version, v3)]*

**A Fair Comparison of Graph Neural Networks for Graph Classification**

Federico Errica, Marco Podda, Davide Bacciu, Alessio Micheli

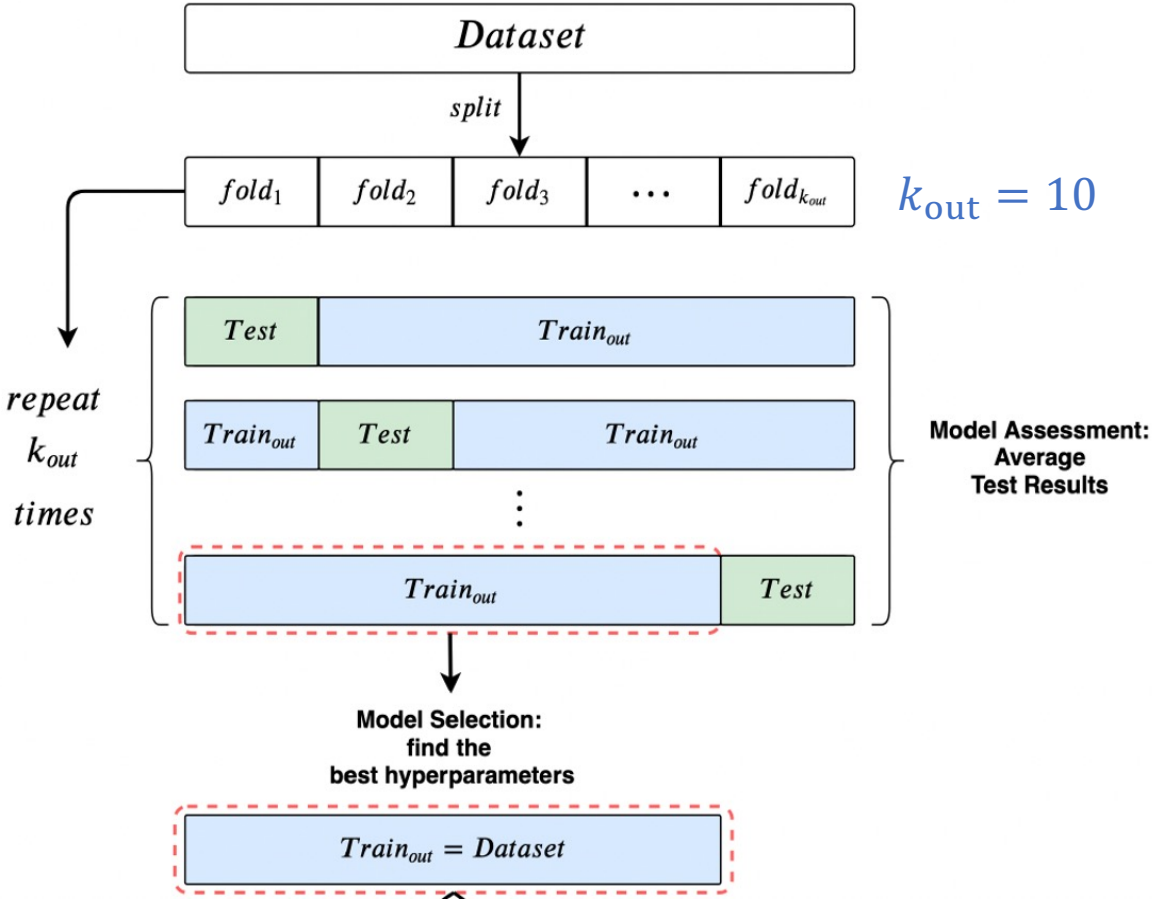
5 modèles
9 data sets
47000 entraînements

# La validation croisée pour la sélection d'un modèle

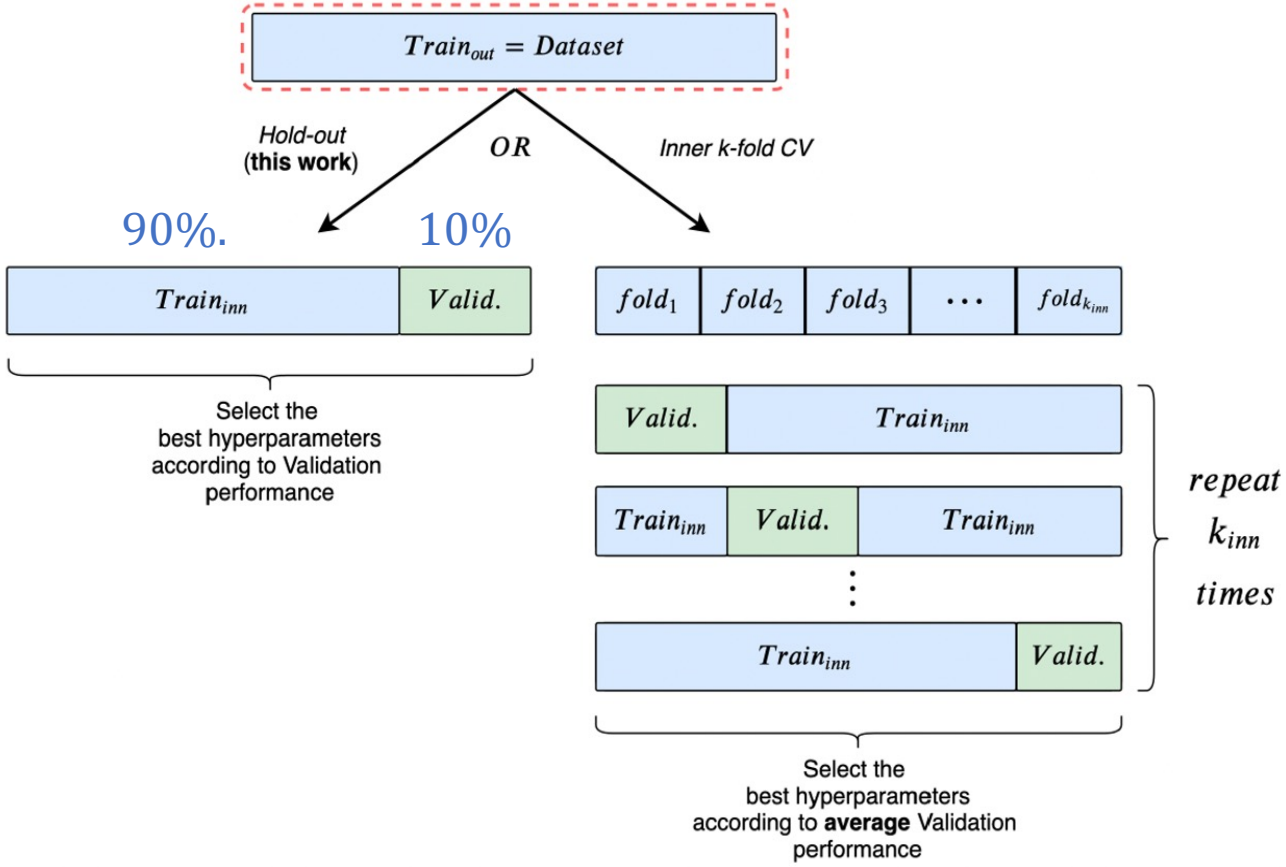


# La validation croisée pour l'évaluation d'un modèle

## Évaluation




## Sélection



# La validation croisée pour l'évaluation d'un modèle

## Évaluation des performances

**Algorithm 1** Model Assessment ($k$-fold CV)

Input: Dataset $\mathcal{D}$, set of configurations $\Theta$
Split $\mathcal{D}$ into $k$ folds $F_1, \ldots, F_k$
**for** $i \leftarrow 1, \ldots, k$ **do**
  $\text{train}_k, \text{test}_k \leftarrow \left(\bigcup_{j \neq i} F_j\right), F_i$
  $\boxed{\text{best}_k} \leftarrow \text{Select}(\text{train}_k, \Theta)$
  **for** $r \leftarrow 1, \ldots, R$ **do**
    $\text{model}_r \leftarrow \text{Train}(\text{train}_k, \boxed{\text{best}_k})$
    $\text{p}_r \leftarrow \text{Eval}(\text{model}_k, \text{test}_k)$
  **end for**
  $\text{perf}_k \leftarrow \sum_{r=1}^{R} \text{p}_r / R$
**end for**
**return** $\sum_{i=1}^{k} \text{perf}_i / k$

Initialisation aléatoire des poids

Early stopping avec hold out de 10% sur $\text{train}_k$

$k = 10, \qquad R = 3$

## Sélection du modèle

**Algorithm 2** Model Selection

Input: $\text{train}_k$, $\Theta$
Split $\text{train}_k$ into *train* and *valid*
$\text{p}_\theta = \emptyset$
**for each** $\theta \in \Theta$ **do**
  model $\leftarrow$ Train( *train* , $\theta$)
  $\text{p}_\theta \leftarrow \text{p}_\theta \cup \text{Eval}(\text{model}, \textit{valid})$
**end for**
$\text{best}_\theta \leftarrow \text{argmax}_\theta \, \text{p}_\theta$
**return** $\text{best}_\theta$

GridSearch

# Une check-list de bonnes pratiques pour la reproductibilité

- Publier le **code** de **pré-processing** des données
- Publier le **code** de **sélection** du meilleur modèle
- Publier le **code d'évaluation** du modèle
- Fournir les **partitions** train / validation / test
- Enumérer la **liste des hyper-paramètres** optimisés en CV
  (nb couches, learning rate, optimiseur, critère pour early stopping et…)
- Utiliser un échantillonnage **stratifié** pour créer les partitions (respect des proportions des classes)
- Reporter la **déviation standard** des $k$ évaluations obtenues par CV et non seulement le score moyen
- Utiliser un **germe aléatoire** pour la reproductibilité des opérations d'échantillonnage
- Reporter les **scores de test** et non pas les scores de validation, toujours biaisés et trop optimistes
- **Pré-calculer les partitions** train/validation/test et appliquer les mêmes à tous les modèles
- **Avoir une baseline agnostique** vis-à-vis de la structure du graphe

# Avoir une baseline agnostique de la structure du graphe !

- Permet de valider l'utilité de prendre en compte la structure du graphe
- Permet de valider la capacité du GNN à prendre en compte cette structure

- S'il n'y pas de différence avec la base line agnostique,
  - soit la structure du graphe n'intervient pas dans la prédiction
  - soit le GNN ne parvient pas à l'exploiter

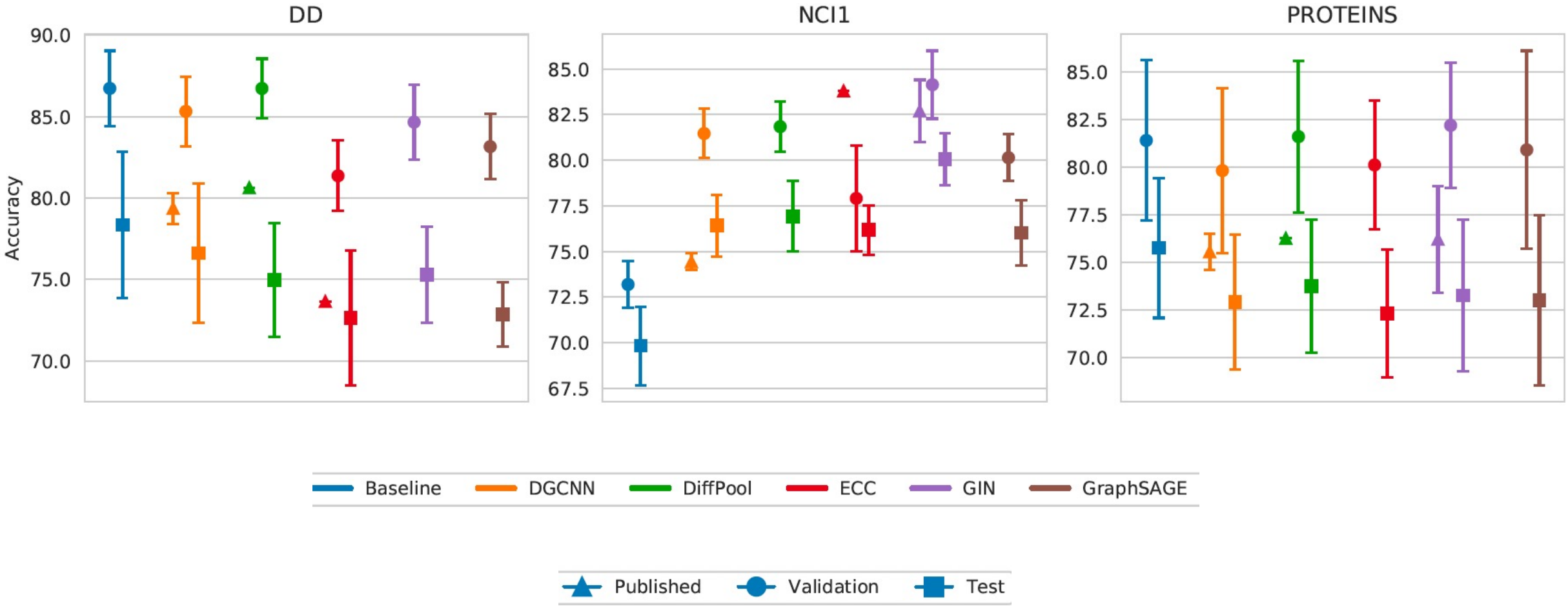
| | D&D | NCI1 | PROTEINS | ENZYMES |
|---|---|---|---|---|
| Baseline | **78.4** $\pm$ 4.5 | 69.8 $\pm$ 2.2 | **75.8** $\pm$ 3.7 | **65.2** $\pm$ 6.4 |
| DGCNN | 76.6 $\pm$ 4.3 | 76.4 $\pm$ 1.7 | 72.9 $\pm$ 3.5 | 38.9 $\pm$ 5.7 |
| DiffPool | 75.0 $\pm$ 3.5 | 76.9 $\pm$ 1.9 | 73.7 $\pm$ 3.5 | 59.5 $\pm$ 5.6 |
| ECC | 72.6 $\pm$ 4.1 | 76.2 $\pm$ 1.4 | 72.3 $\pm$ 3.4 | 29.5 $\pm$ 8.2 |
| GIN | 75.3 $\pm$ 2.9 | **80.0** $\pm$ 1.4 | 73.3 $\pm$ 4.0 | 59.6 $\pm$ 4.5 |
| GraphSAGE | 72.9 $\pm$ 2.0 | 76.0 $\pm$ 1.8 | 73.0 $\pm$ 4.5 | 58.2 $\pm$ 6.0 |

# Faut-il rajouter des features de graphe tel que les degrés ?

- Pour le baseline rajouter une feature de degré de chaque nœud peut augmenter significativement la précision.
- Pour les GNN c'est moins utile puisqu'ils sont sensés extraire cette information du graphe (… demande à être discuté…)
- Pour les GNN il semble cependant que rajouter le degré comme feature diminue le nombre de couches nécessaires d'une unité environ.
- Le ranking des modèles change après ajout d'une feature de degré.

| | | IMDB-B | IMDB-M | REDDIT-B | REDDIT-5K | COLLAB |
|---|---|---|---|---|---|---|
| No Features | Baseline | $50.7 \pm 2.4$ | $36.1 \pm 3.0$ | $72.1 \pm 7.8$ | $35.1 \pm 1.4$ | $55.0 \pm 1.9$ |
| | DGCNN | $53.3 \pm 5.0$ | $38.6 \pm 2.2$ | $77.1 \pm 2.9$ | $35.7 \pm 1.8$ | $57.4 \pm 1.9$ |
| | DiffPool | $68.3 \pm 6.1$ | $45.1 \pm 3.2$ | $76.6 \pm 2.4$ | $34.6 \pm 2.0$ | $67.7 \pm 1.9$ |
| | ECC | $67.8 \pm 4.8$ | $44.8 \pm 3.1$ | OOR | OOR | OOR |
| | GIN | $66.8 \pm 3.9$ | $42.2 \pm 4.6$ | $\mathbf{87.0} \pm 4.4$ | $\mathbf{53.8} \pm 5.9$ | $\mathbf{75.9} \pm 1.9$ |
| | GraphSAGE | $\mathbf{69.9} \pm 4.6$ | $\mathbf{47.2} \pm 3.6$ | $86.1 \pm 2.0$ | $49.9 \pm 1.7$ | $71.6 \pm 1.5$ |
| With Degree | Baseline | $70.8 \pm 5.0$ | $\mathbf{49.1} \pm 3.5$ | $82.2 \pm 3.0$ | $52.2 \pm 1.5$ | $70.2 \pm 1.5$ |
| | DGCNN | $69.2 \pm 3.0$ | $45.6 \pm 3.4$ | $87.8 \pm 2.5$ | $49.2 \pm 1.2$ | $71.2 \pm 1.9$ |
| | DiffPool | $68.4 \pm 3.3$ | $45.6 \pm 3.4$ | $89.1 \pm 1.6$ | $53.8 \pm 1.4$ | $68.9 \pm 2.0$ |
| | ECC | $67.7 \pm 2.8$ | $43.5 \pm 3.1$ | OOR | OOR | OOR |
| | GIN | $\mathbf{71.2} \pm 3.9$ | $48.5 \pm 3.3$ | $\mathbf{89.9} \pm 1.9$ | $\mathbf{56.1} \pm 1.7$ | $\mathbf{75.6} \pm 2.3$ |
| | GraphSAGE | $68.8 \pm 4.5$ | $47.6 \pm 3.5$ | $84.3 \pm 1.9$ | $50.0 \pm 1.3$ | $73.9 \pm 1.7$ |

# Ce qui est publié et ce que dit une évaluation rigoureuse



# Sur la classification de nœuds

- Importance de **randomiser le les partitions** train/validation/test
- Utiliser une même **procédure standardisée pour l'optimisation des hyper-paramètres**
- Le **GCN s'en sort très bien** si l'on utilise les mêmes méthodes d'entraînement et si l'on calcule une moyenne sur les partitions train/validation/test

| | CORA | CiteSeer | PubMed | CORA Full |
|---|---|---|---|---|
| **GCN** | $81.5 \pm 1.3$ | $\mathbf{71.9} \pm 1.9$ | $77.8 \pm 2.9$ | $\mathbf{62.2} \pm 0.6$ |
| **GAT** | $\mathbf{81.8} \pm 1.3$ | $71.4 \pm 1.9$ | $\mathbf{78.7} \pm 2.3$ | $51.9 \pm 1.5$ |
| **MoNet** | $81.3 \pm 1.3$ | $71.2 \pm 2.0$ | $78.6 \pm 2.3$ | $59.8 \pm 0.8$ |
| **GS-mean** | $79.2 \pm 7.7$ | $71.6 \pm 1.9$ | $77.4 \pm 2.2$ | $58.6 \pm 1.6$ |
| **GS-maxpool** | $76.6 \pm 1.9$ | $67.5 \pm 2.3$ | $76.1 \pm 2.3$ | $40.7 \pm 1.5$ |
| **GS-meanpool** | $77.9 \pm 2.4$ | $68.6 \pm 2.4$ | $76.5 \pm 2.4$ | $40.5 \pm 1.5$ |
| **MLP** | $58.2 \pm 2.1$ | $59.1 \pm 2.3$ | $70.0 \pm 2.1$ | $36.8 \pm 1.0$ |
| **LogReg** | $57.1 \pm 2.3$ | $61.0 \pm 2.2$ | $64.1 \pm 3.1$ | $40.5 \pm 0.8$ |
| **LabelProp** | $74.4 \pm 2.6$ | $67.8 \pm 2.1$ | $70.5 \pm 5.3$ | $50.5 \pm 1.5$ |
| **LabelProp NL** | $73.9 \pm 1.6$ | $66.7 \pm 2.2$ | $72.3 \pm 2.9$ | $51.0 \pm 1.0$ |

article

arXiv > cs > arXiv:1811.05868

**Computer Science > Machine Learning**

*[Submitted on 14 Nov 2018 (v1), last revised 18 Jun 2019 (this version, v2)]*

**Pitfalls of Graph Neural Network Evaluation**

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, Stephan Günnemann

# Conclusion

- Les **évaluations de performances** de GNN sont rarement menées de manière rigoureuse dans la littérature (du moins avant 2020)
- Il y a de nombreuses **ambiguïtés** dans les procédures d'évaluation et de sélection
- De nombreux articles ne publient pas leur code de prétraitement, de sélection et d'évaluation ce qui rend la **reproductibilité** quasi impossible
- Disposer d'une **baseline agnostique** vis-à-vis de la structure du graphe est essentiel
- L'apport de **features structurelles** autre que le degré est un problème ouvert.
- Une évaluation rigoureuse est **coûteuse** en temps de calcul