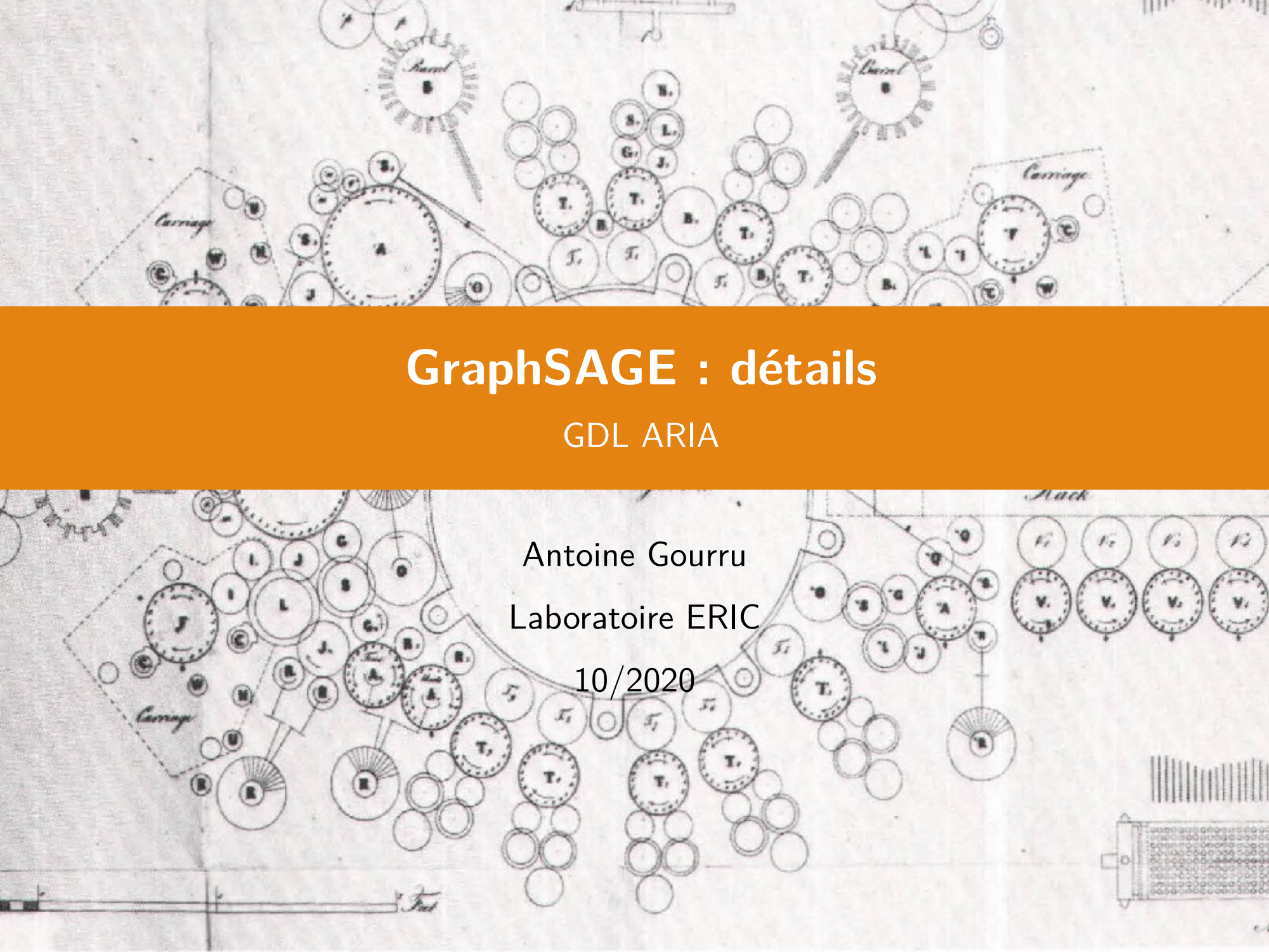


GraphSAGE : détails

GDL ARIA

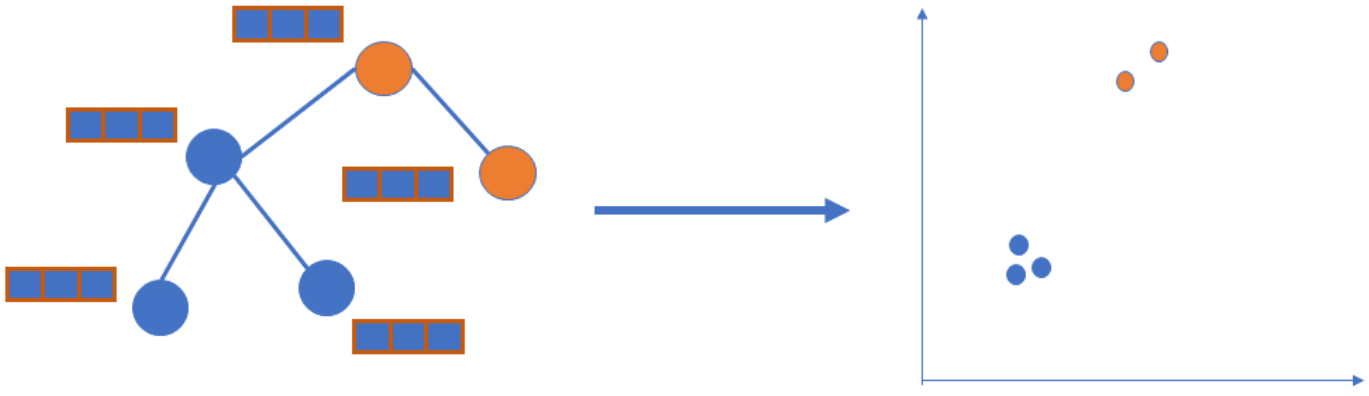
Antoine Gourru

Laboratoire ERIC

10/2020

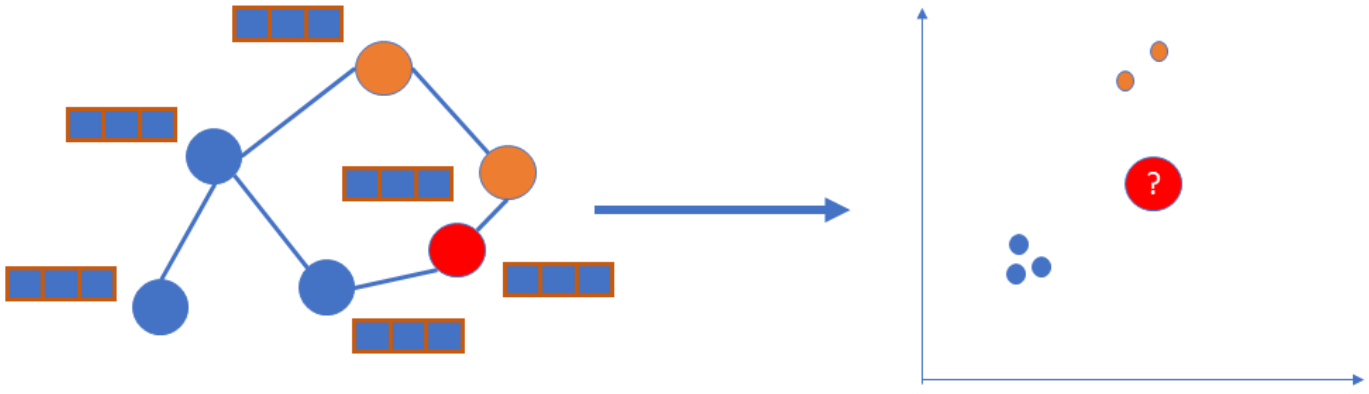
1 Induction VS Transduction

Apprentissage de représentation de graphes avec attributs



- **Réseaux de Documents**
- Réseaux de protéines
- Si pas d'attributs : mesures issues du graphe (dégrées, centralité)

Problème : la plupart des méthodes sont transductives.



On fait quoi si on observe un nouveau noeud ? Il faut tout ré-entraîner.

1 GraphSAGE

GraphSAGE [1] est une méthode inductive qui étend les GCN [2] (présentés par Adrien). Elle est basée sur une *fonction d'agrégation* dont les paramètres sont appris : l'embedding d'un nouveau noeud est fonction des attributs de son voisinage local.

L'entraînement est auto-supervisé (pas sur une tâche spécifique)

Cette méthode préfigure les Graph Attention Network [3], qui seront présentés par Julien. Pour spoiler un peu, les différences entre les deux modèles : GraphSAGE donne autant d'importance à tout le voisinage, contrairement au GAT.

Notations :

- Graph : $(\mathcal{V}, \mathcal{E})$, attributs : $\{x_v, \forall v \in \mathcal{V}\}$, $\mathcal{N}(v)$ voisinage de v
- K matrices de poids : $\{W_k, \forall k \in 1 : K\}$
- fonction d'agrégation : $\{f_k, \forall k \in 1 : K\}$, g fonction d'activation
- embedding : $\{z_v, \forall v \in \mathcal{V}\}$

Algorithm 1 Construction des embeddings

$h_v^0 = x_v, \forall v \in \mathcal{V}$
for $k = 1$ to K **do**
  for $v \in \mathcal{V}$ **do**
    $h = f_k(\{h_u^{k-1}, \forall u \in \mathcal{N}(v)\})$
    $h_v^k = normalize(g(W_k.concat(h_v^{k-1}, h)))$
  end for
end for
$z_v = h_v^K$

Fonctions d'agrégation :

- Moyenne
- LSTM
- Pooling (max après un MLP)
- extension du GCN

$h_v^k = \sigma(W \cdot moyenne(h_u^{k-1} \bigcup \{h_u^{k-1}, \forall u \in \mathcal{N}(v)\}))$
, où σ est la fonction sigmoïde.

Voisinage : construit par sous échantillonnage du voisinage complet pour réduire les temps de calcul. Tirage uniforme de s_k noeuds (différent selon la "couche")

Minimisation de

$$J(G) = \sum_{(u,v)\in C^+} \log \sigma(u^t.v) + r \cdot \mathbb{E}_{v_n \sim p_n(v)}[\log \sigma(-u^t.v_n)] \qquad (1)$$

C^+ est l'ensemble des pairs de noeuds tel que v a cooccuré dans une fenêtre autour de u dans des marches aléatoires, r le nombre de tirage aléatoires et $p_n(v)$ une distribution de tirage négatifs.

Paramètres testés

- Random walk : 50 marches de taille 5 partant de chaque noeud
- 20 échantillons négatifs, tirés selon la fréquence empirique puissance 0.75 [4]
- sgd avec adam
- activation : reLu
- K=2, $s_1 = 25$ et $s_2 = 10$

1 Évaluation

Deux datasets : Posts Reddit et extrait de Reuters (Citations)

Reuters : les auteurs utilisent la méthode d'Arora et al [5] avec word2Vec [4] pour construire les embeddings des abstracts et concatènent avec le degré du noeud.

Pour Reddit : les auteurs concatènent les embeddings du titre (moyenne de GLOVE [6]), du contenu, et ajoutent nb de commentaires et le score du post.

1 Résultats

Tâche d'évaluation : classification de document avec régression logistique (les descripteurs sont les embeddings de chaque noeud). Comparaison de deux cas : non supervisé, et supervisé. Dans le cas supervisé, ils remplacent la perte précédente avec une cross entropy sur les labels associés aux noeuds dans l'ensemble d'entraînement.

	Citation		Reddit	
Name	Unsup. F1	Sup. F1	Unsup. F1	Sup. F1
Random	0.206	0.206	0.043	0.042
Raw features	0.575	0.575	0.585	0.585
DeepWalk	0.565	0.565	0.324	0.324
DeepWalk + features	0.701	0.701	0.691	0.691
GraphSAGE-GCN	0.742	0.772	**0.908**	0.930
GraphSAGE-mean	0.778	0.820	0.897	0.950
GraphSAGE-LSTM	0.788	0.832	**0.907**	**0.954**
GraphSAGE-pool	**0.798**	**0.839**	0.892	0.948
% gain over feat.	39%	46%	55%	63%

1 Résultats : insights

- $K = 2$: boost de la précision de 10-15% en moyenne comparé à $K = 1$. $K > 2$: augmentation marginale (0-5%) et multiplie les temps de calcul par $\sim$10-100
- plateau de performance observé pour des s_k assez bas.
- LSTM et max-pooling marchent mieux

- Méthode inductive : hyper importante dans des contextes réels de recommandation.
- méthode évolutive : dans les xp, les auteurs utilisent des moyennes d'embeddings de mots. On peut utiliser une méthode de représentation vectorielle du document plus récente : e.g. SBERT [7], ou BERT [8].
- subsampling accélère beaucoup les calculs (surtout pour les graphes denses).

1 XP cadeau

Quelques XP en cadeau : test sur CORA, entraînement supervisé, dimension 50, $s_1 = s_2 = 10$, stellargraph [1].
Est-ce qu'on gagne beaucoup en utilisant des méthodes plus récentes d'embedding de documents ?

ratio train/test	10%	30%	50%
DeepWalk [9]	70.6 (2.0)	-	81.0 (0.7)
TADW [10]	81.9 (0.8)	-	87.4 (0.8)
GELD [11]	84.3 (1.1)	-	88.3 (0.4)
tf_idf	68.77 (2.87)	78.54 (0.57)	80.87 (0.74)
gs_idf	83.48 (0.48)	84.44 (0.38)	84.62 (0.82)
glove	63.34 (1.3)	69.64 (0.71)	72.03 (0.86)
gs_glove	82.09 (0.53)	82.78 (0.42)	83.24 (0.84)
use	61.26 (1.13)	68.26 (0.76)	70.52 (1.19)
gs_use	81.15 (0.88)	82.91 (0.32)	82.97 (0.94)
sbert	54.4 (1.49)	62.2 (1.43)	65.48 (0.98)
gs_sbert	78.27 (1.13)	80.26 (0.77)	80.61 (0.82)

1. https://stellargraph.readthedocs.io

1 Références I

William L Hamilton, Rex Ying, and Jure Leskovec.
Inductive representation learning on large graphs.
In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 1025–1035, 2017.

Thomas N Kipf and Max Welling.
Semi-supervised classification with graph convolutional networks.
arXiv preprint arXiv:1609.02907, 2016.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio.
Graph attention networks.
arXiv preprint arXiv:1710.10903, 2017.

1 Références II

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean.
Distributed representations of words and phrases and their compositionality.
In *Advances in neural information processing systems*, pages 3111–3119, 2013.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma.
A simple but tough-to-beat baseline for sentence embeddings.
In *International Conference on Learning Representations*, 2016.

Jeffrey Pennington, Richard Socher, and Christopher Manning.
Glove : Global vectors for word representation.
In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Nils Reimers and Iryna Gurevych.
Sentence-bert : Sentence embeddings using siamese bert-networks.
arXiv preprint arXiv:1908.10084, 2019.

1 Références III

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova.
Bert : Pre-training of deep bidirectional transformers for language understanding.
In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena.
Deepwalk : Online learning of social representations.
In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.

Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y Chang.
Network representation learning with rich text information.
In *IJCAI*, volume 2015, pages 2111–2117, 2015.

1 Références IV

Antoine Gourru, Julien Velcin, and Julien Jacques.
Gaussian embedding of linked documents from a pretrained semantic space.
In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, 2020.