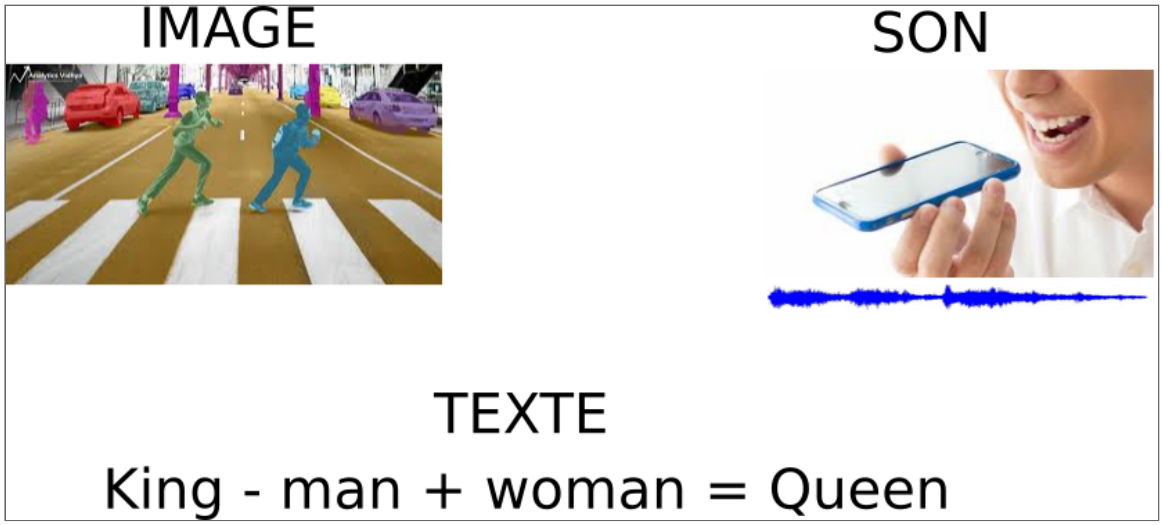
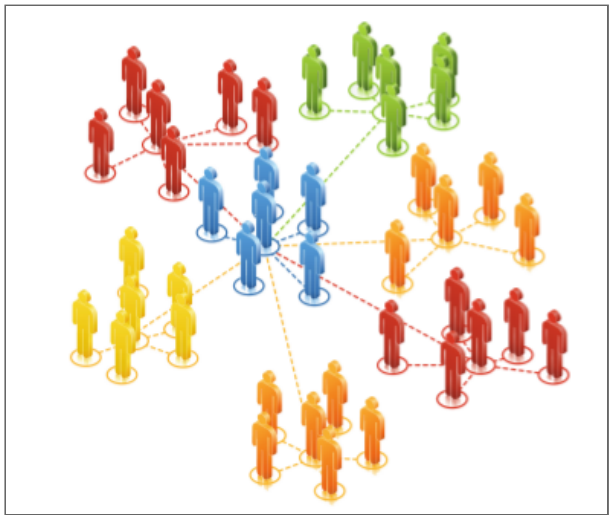
APPRENDRE SUR DES GRAPHES

PREMIERS SUCCÉS DU DEEP LEARNING (2010)



Depuis 2016, on entend de plus en plus parler de "Graph neural networks"

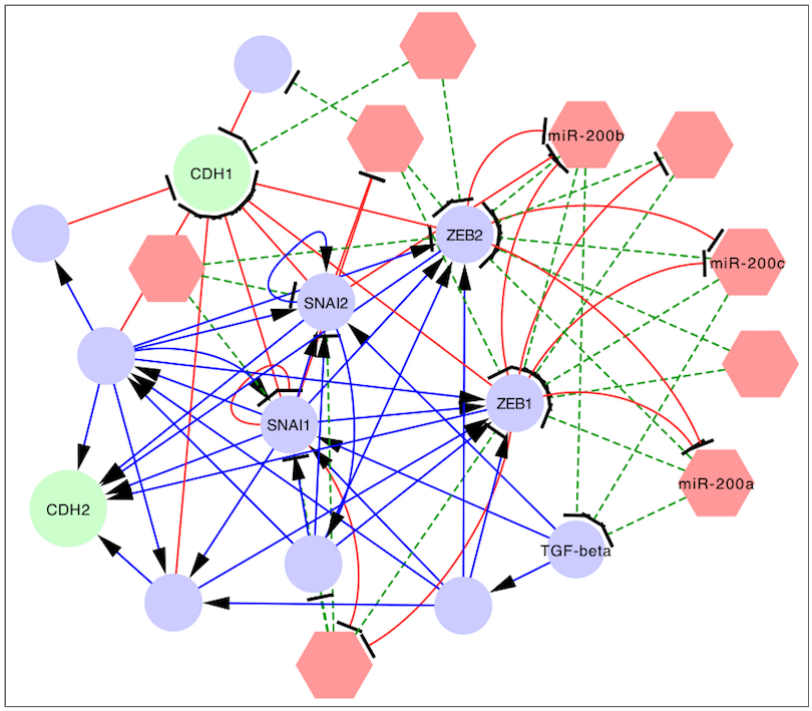
POURQUOI LES GRAPHES?



Analyse des réseaux sociaux

- Détection de communautés
- Reconnaissance de profils de clients (ou de terroriste)

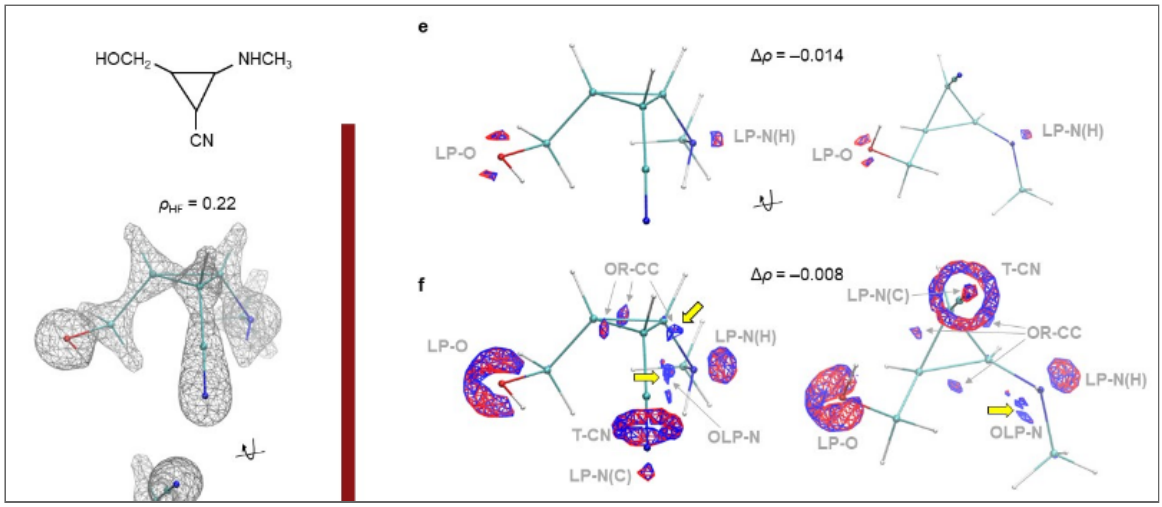
POURQUOI LES GRAPHES?



Réseau de co-expression de gènes

- Prédiction de caractéristiques génétiques

POURQUOI LES GRAPHES?

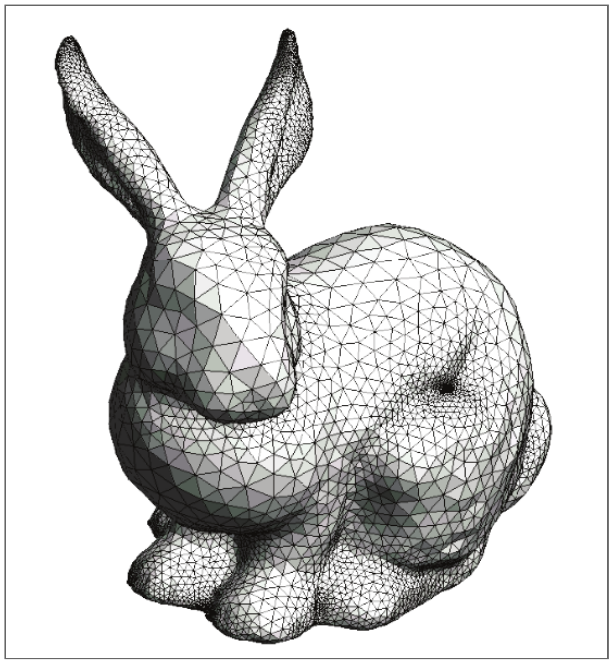


Chimie quantique

- Prédiction des propriétés des molécules

Gimler et al. Neural Message Passing for Quantum Chemistry, JMLR 2017

POURQUOI LES GRAPHES?

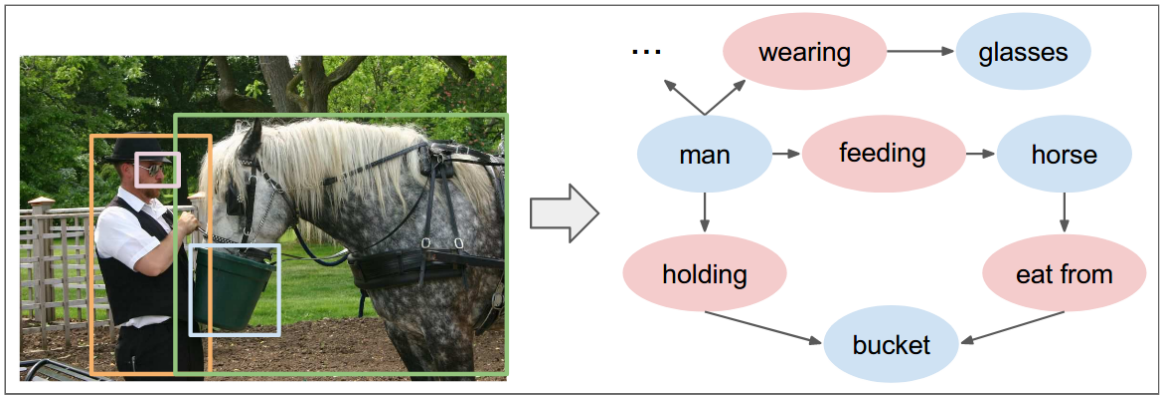


Traitement des nuages de points 3D.

- Capturer la géométrie d'une forme
- Classification, détection, segmentation

Qi et al. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, CVPR 2017

POURQUOI LES GRAPHES?



Diviser pour mieux régner!

- Problèmes trop complexes à traiter de front
- CNN modèlise mal les relations

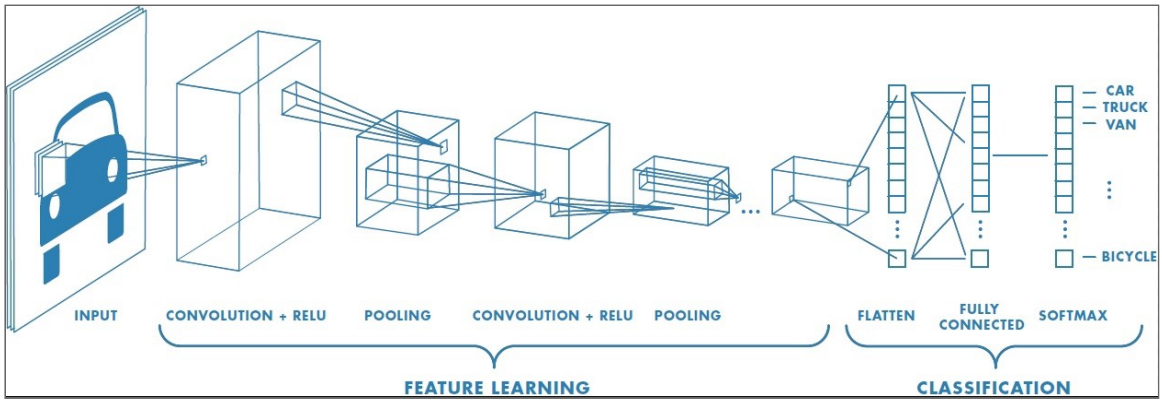
Xu et al. Scene graph generation by iterative message passing, ICCV 2017

TAXONOMIE DES PROBLÈMES

- Prédiction au niveau du noeud.
  - Transductif: prédiction au sein du même graphe
  - Inductif: Apprendre et tester sur des graphes différents
- Prédiction au niveau de l'arc.
  - Y-a-t'il une relation entre les noeuds i et j?
- classification globale du graphe.

On veut prendre en compte les descripteurs ET la topologie du graphe

PROPRIÉTÉS UTILES DES CNNS



- Invariance à la translation: convolution
- Filtres localisés dans l'espace
- Séparation des échelles: composition et pooling.
- Nbre de paramètres indépendants de la taille de l'image

DIFFICULTÉS DANS LE CAS DES GRAPHES

- La translation n'est plus définie
- La notion de distance non plus
- Le domaine varie
- En fait, la plupart des modules du CNN doivent être redéfinies

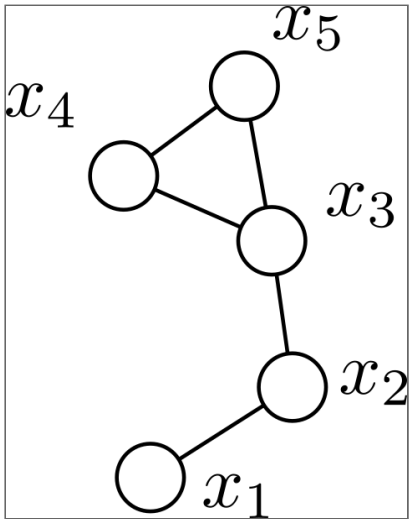
Comment apprendre sur un espace non euclidien?

Bronstein, Geometric deep learning: going beyond euclidean data IEEE Signal Proc. 2017

PARTIE 1 THÉORIE

Filtrage par modulation des valeurs propres du Laplacien

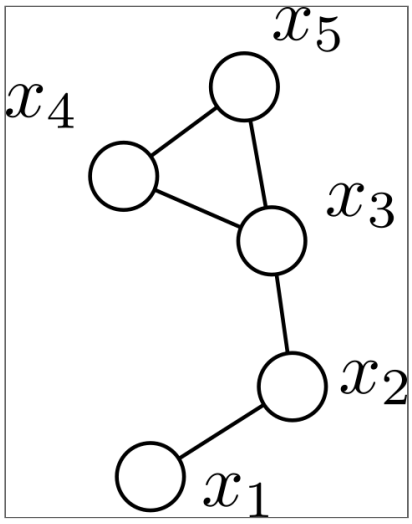
QUELQUES NOTATIONS



Soit G(E,V) un graphe avec

- un ensemble de noeuds $V = V_i, i \in 1..N$
- un ensemble d'arcs $E = e_{ij}(i,j) \in 1..N \times 1..N$
- Chaque noeud v_i est décrit par un vecteur f_i

QUELQUES NOTATIONS

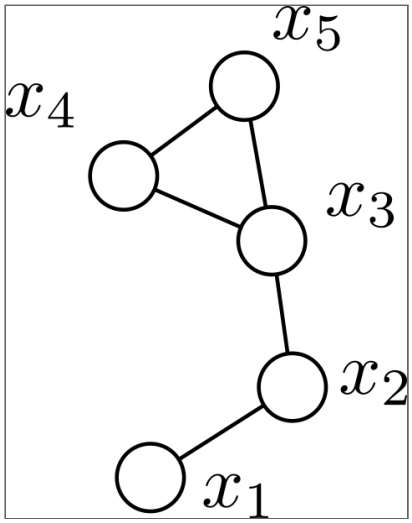


Matrice d'adjacence

$A \in B^{N \times N}, A(i,j) = 1$ si $(i,$

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

QUELQUES NOTATIONS

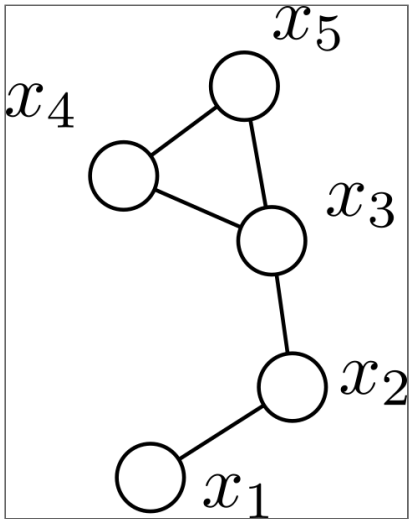


Matrice des degrés

$D \in \mathrm{N}^{N \times N}, D(i,i) = \sum_j A(i$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

QUELQUES NOTATIONS



La matrice laplacienne

$\Delta = D - A$

$$\begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 3 & -1 & -1 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & -1 & 2 \end{bmatrix}$$

INTÉRPRÉTATION DU LAPLACIEN

- Cette laplacienne encode la topologie du graphe

$$(\Delta f)_i = \sum_j a_{ij}(f_i - f_j)$$

où $f = f_1, \ldots, f_n$ contient les descripteurs 1D des noeuds.

- L'opérateur s'apparente donc à la différence entre un noeud et son voisinage.
- Opérateur qui transforme le signal d'entrée f en fonction de la topologie du graphe

RAPPEL SUR FOURIER

- Projection d'un signal x de son espace original par une base orthogonale

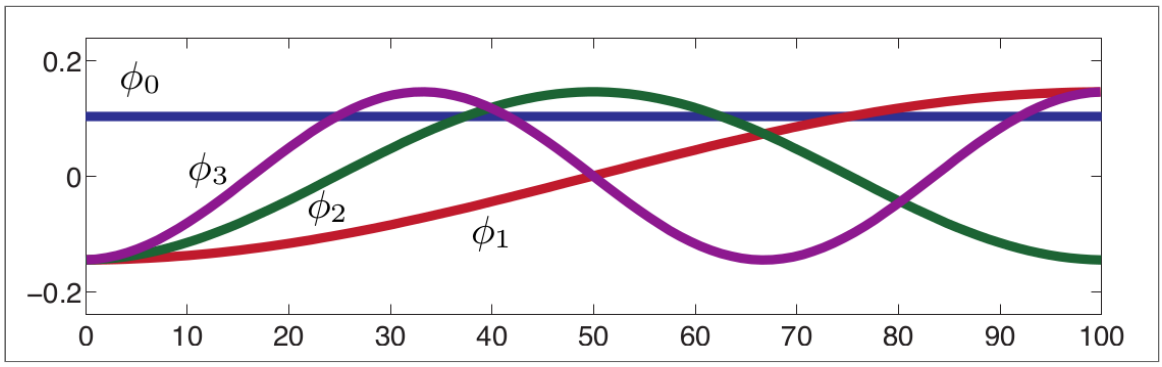
$$(f \star k)(\xi) = \int_{-\inf}^{\inf} f(x) e^{-i\xi x} dx$$

- Filtrage d'un signal dans l'espace des fréquences
- Lien entre les deux espaces

$$F(f * g) = F(g)F(f)$$

INTERPRÉTATION DES VECTEURS DE LA BASE

Dans le cas d'une transformée s'appuyant sur des cosinus

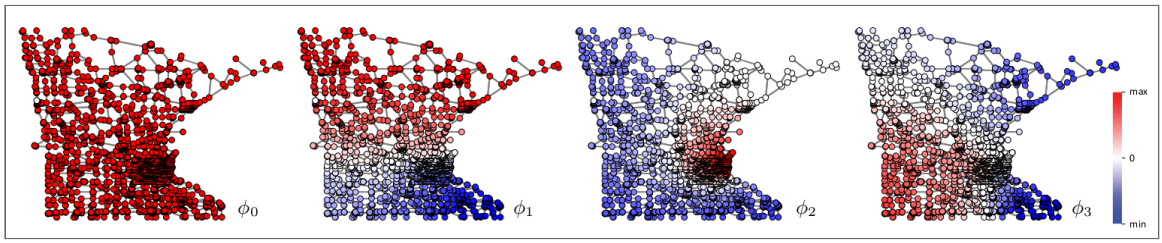


APPLICATION AUX GRAPHES

L'analogue de Fourier sur des graphes utilise la décomposition en valeur propre du Laplacien

$$\Delta = \Phi \Lambda \Phi$$

Visualisation des vecteurs propres Φ_i du Laplacien



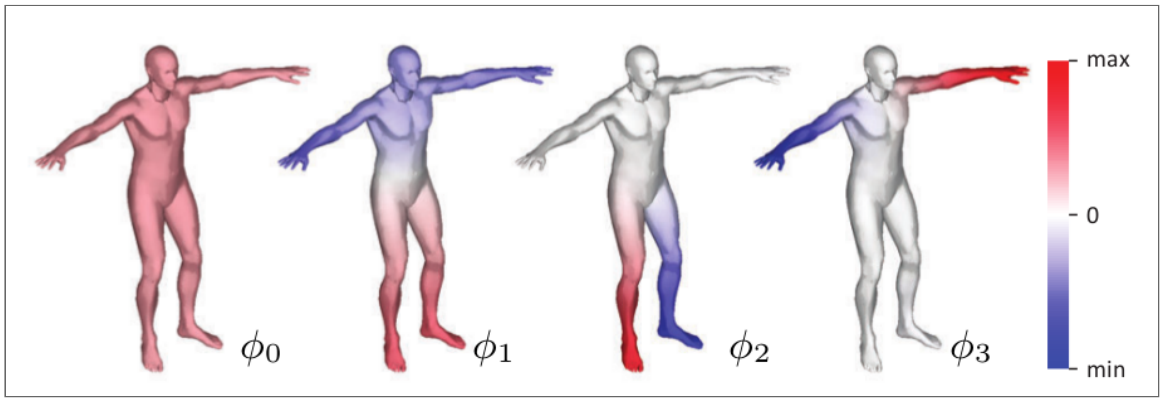
Réseau routier de l'état de New-York

APPLICATION AUX GRAPHES

L'analogue de Fourier sur des graphes utilise la décomposition en valeur propre du Laplacien

$$\Delta = \Phi \Lambda \Phi$$

Visualisation des vecteurs propres Φ_i du Laplacien



spectral theorem add the justification that you the number of eigen values which are 0 is the number of connected components, and that you have 1 on the eigen vectors.

CONVOLUTION DE GRAPHES

La non définition de la translation empêche de convoluer dans l'espace du graphe.

Mais on peut multiplier dans l'espace des vecteurs propres du Laplacien.

CONVOLUTION DE GRAPHES

Une couche de convolution spectrale est définie par:

$$g_l = \sigma(\sum_{l'=1}^{p} \Phi_k W_{l,l'} \Phi_k^T f_{l'})$$

- $F = (f_1 \dots f_p)$ est le signal d'entrée
- $G = (g_1 \dots g_q)$ est le signal de sortie
- Φ_k contient les k premiers vecteurs propres
- $W_{l,l'}$ est une matrice de poids de taille $k \times k$

CONVOLUTION SUR DES GRAPHES

Une couche de convolution spectrale est définie par:

$$g_l = \sigma(\sum_{l'=1}^{p} \Phi_k W_{l,l'} \Phi_k^T f_{l'})$$

- W peut être vu comme un modulateur des valeurs propres du laplacien Δ

QUELQUES PROBLÈMES

- Coût en calcul: décomposition en valeurs propres du Laplacien
- Filtres non localisés (la décomposition en vecteurs propres est globale)
- Dépendant du domaine
  - 2 graphes $\neq \Rightarrow$ 2 laplaciens $\neq$

Il faut adapter celà à la pratique

SIMPLIFICATION

- Utilisation de décomposition en polynômes:

$$\Phi_k W_{l,l'} \Phi_k^T \approx \sum_{k=1}^{K} \theta_k T_k(\Delta)$$

- avec $K = 1$, une couche de convolution devient

$$G = \sigma(\tilde{\Delta} F W)$$

  - avec $\tilde{\Delta}$ un laplacien normalisé et W une matrice de poids

QU'AVONS NOUS GAGNÉ?

Graph Convolutional Network (GCN)

$$G = \tilde{\Delta} F W$$

- Plus de décomposition en valeurs propres
- Localisation des filtres à cause de la nature locale du Laplacien

Kipf et al, Semi-supervised classification with graph convolutional networks, ICLR 2017

LIEN AVEC L'ENVOI DE MESSAGES

La multiplication du signal par le laplacien comme un opérateur 1-hop
Autrement dit, le descripteur du noeud i est mis à jour uniquement en fonction des descripteurs de ses voisins.

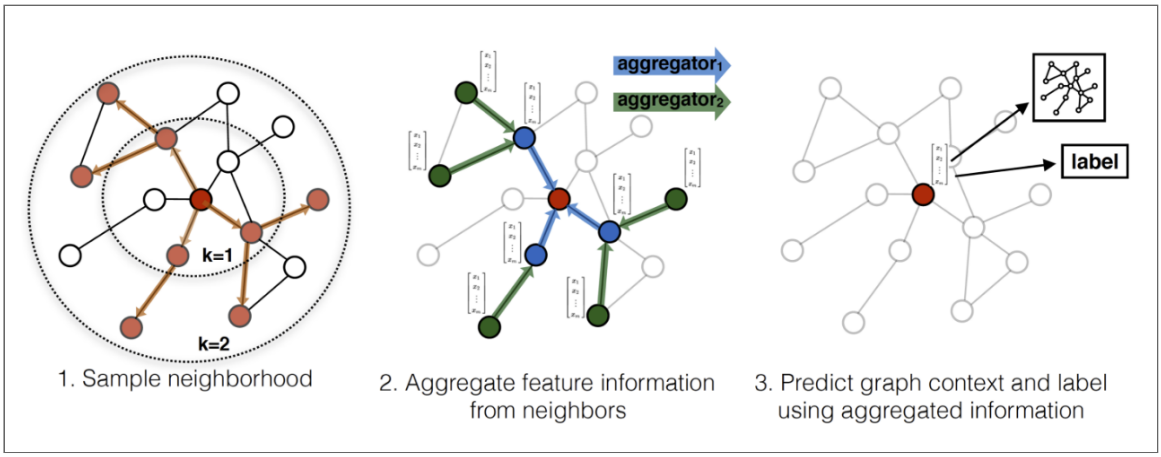
DERNIER PROBLÈME

Graph Convolutional Network (GCN)

$$G = \tilde{\Delta} F W$$

Ces méthodes sont transductive par nature
(2 graphes $\neq \Rightarrow$ 2 laplaciens $\neq$)

GRAPHSAGE: UNE MÉTHODE INDUCTIVE



- Apprendre un modèle générique de messages
- indépendamment de la position du noeud i.
- Partage des poids

GRAPHSAGE: UNE MÉTHODE INDUCTIVE

Algorithm 1: GraphSAGE embedding generation (i.e., forward propagation) algorithm

Input : Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$; input features $\{\mathbf{x}_v, \forall v \in \mathcal{V}\}$; depth K; weight matrices $\mathbf{W}^k, \forall k \in \{1, ..., K\}$; non-linearity σ; differentiable aggregator functions $\text{AGGREGATE}_k, \forall k \in \{1, ..., K\}$; neighborhood function $\mathcal{N} : v \rightarrow 2^{\mathcal{V}}$

Output : Vector representations $\mathbf{z}_v$ for all $v \in \mathcal{V}$

$\mathbf{h}_v^0 \leftarrow \mathbf{x}_v, \forall v \in \mathcal{V}$;
for $k = 1...K$ **do**
  for $v \in \mathcal{V}$ **do**
    $\mathbf{h}_{\mathcal{N}(v)}^k \leftarrow \text{AGGREGATE}_k(\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\})$;
    $\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W}^k \cdot \text{CONCAT}(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^k)\right)$
  end
  $\mathbf{h}_v^k \leftarrow \mathbf{h}_v^k / \|\mathbf{h}_v^k\|_2, \forall v \in \mathcal{V}$
end
$\mathbf{z}_v \leftarrow \mathbf{h}_v^K, \forall v \in \mathcal{V}$

COUCHE DE CONVOLUTION GÉNÉRALE

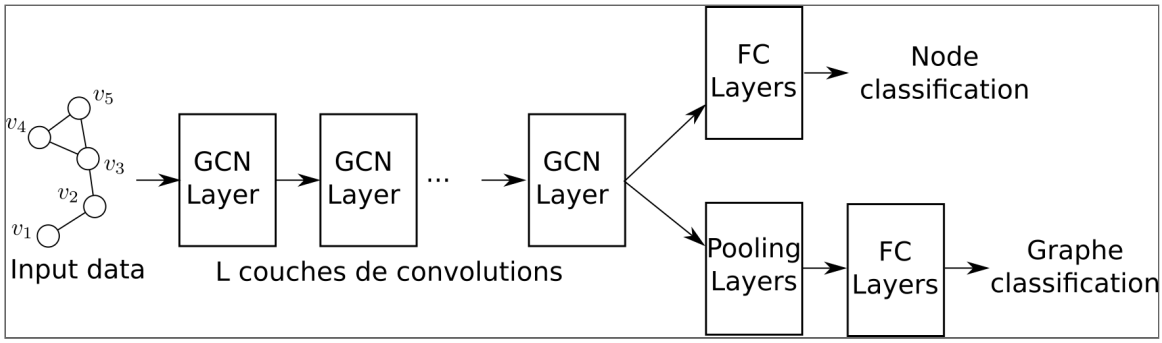
$$m_i = \text{AGGREGATE}(\{h_j^{k-1}, j \in N_i\})$$

$$h_i^k = \sigma(\text{UPDATE}(h_i, m_i))$$

- Récupérer les états $h_j^{k-1}, j \in N_i$ de la couche $k-1$
- Les aggréger pour les transformer en message m_i
- Mettre à jour h_i^k en utilisant m_i

UPDATE et AGGREGATE peuvent être implémentés par des réseaux de neurones

UNE ARCHITECTURE CLASSIQUE



- Aggrégation d'information au niveau des couches GCN et du pooling
- Augmenter la profondeur ⇒ propager l'information dans le graphe.

POSSIBILITÉS POUR AGGREGATE

- couche complétement connectée et aggr. SOMMME:

$$m_i = \sum_{j \in N_i} \text{MLP}(h_j)$$

- Somme pondéré par la distance

$$m_i = \sum_{j \in N_i} \text{MLP}(h_j) g(u(h_i, h_j))$$

avec u distance euclidienne et g un noyau gaussian.

POSSIBILITÉS POUR AGGREGATE

- Inclure la différence relative:

$$m_i = \sum_{j \in N_i} \text{MLP}[h_i, (h_i - h_j)]$$

- Pondération avec un méchanisme d'attention

$$m_i = \sum_{j \in N_i} \text{MLP}(h_j a_j) \text{ où } a_j = W[h_j, h_i]$$

Wang et al. Dynamic graph cnn for learning on point clouds, ACM TOG, 2019

POSSIBILITÉS POUR UPDATE

- Utiliser simplement l'addition
- Voir chaque noeud comme un RNN:

  $h_i = LSTM(h_i, m_i)$

  avec m_i l'observation et h_i l'état latent.
- Eventuellement une étape de normalisation (Stabilisation du gradient)

POOLING SUR DES GRAPHES

Comme pour la fonction merge, on cherche à résumer

- Opérateurs simples: Max, Mean
- Utilisation d'un LSTM

l'information

- Utilisation d'un méchanisme d'attention

ET ÇA MARCHE?

- État de l'art sur des jeu de données étalons (CORA, PROTEINS, SHAPENET)
- Travaux théoriques en cours pour comprendre la puissance de ces méthodes

Des librairies haut niveau très accessibles:

- Pytorch geometric
- DGL (surcouche générique pouvant s'appliquer à pytorch)
- GraphNets de deepmind