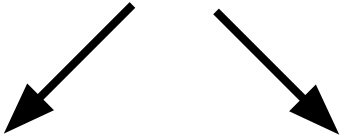
Graph convolutions



Spectral methods

I metodi “spettrali” cercano di usare il concetto di segnale per rappresentare le caratteristiche del nodo, e operazioni come la trasformata di Fourier per aggregare le informazioni tra nodi ed effettuare predizioni.

Spatial methods

I metodi “spaziali” usano tecniche di “message passing” e metodi rappresentativi come pseudo coordinate per aggregare le informazioni tra nodi ed effettuare predizioni.

Spectral methods

L’operazione di convoluzione può essere computata calcolando la scomposizione in autovalori ed autovettori della matrice Laplaciana associata al grafo. Calcolando gli autovettori troviamo anche la base di Fourier per il grafo. Questo tipo di calcolo è molto pesante, in quanto ci saranno tanti autovettori quanti nodi del grafo.
Esistono diverse soluzioni che cercano di approssimare e semplificare il problema:

- **ChebNets**
- **GCN**
- **FastGCN**
- **Altri...**

ChebNets

Le convoluzioni spettrali sono definite come la moltiplicazione di un segnale (nodo/features/attributi) per un kernel. E' simile al modo in cui viene applicata la convoluzione su un immagine, dove un pixel viene moltiplicato per il valore di un kernel.
Il kernel utilizzato in questo caso è fatto da **polinomi di Chebyshev** dalla matrice diagonale degli autovettori della Laplaciana.

$$g_\theta(\Lambda) = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\Lambda})$$

Dove g_theta è il kernel (theta è il vettore dei coefficienti di Chebyshev) applicato a Lambda, la matrice degli autovalori diagonale della Laplaciana. k è il valore d'ordine di vicinanza minimo, K il valore massimo. T_k sono i polinomi di chebyshev del k-esimo ordine. L'ordine rappresenta la profondità del grafo, ovvero viene misurata la somiglianza tra k nodi vicini. Il primo ordine quindi si basa solo sui nodi immediatamente nelle vicinanze, mentre il secondo ordine (e così via) continuano ad analizzare nodi più distanti.

GCNs (Graph Convolutional Network)

In questo caso la convoluzione è data da:

$$g_{\theta'} \star x \approx \sum_{k=0}^{K} \theta'_k T_k(\tilde{L}) x$$

Dove g_theta è il kernel (theta sono i parametri) applicato (**star**) a x, ovvero un segnale del grafo. k è il valore d'ordine di vicinanza minimo, K il valore massimo. T_k sono i polinomi di chebyshev del k-esimo ordine applicati ad L^~. Lambda_max rappresenta l'autovalore più grande di L, la Laplacian normalizzata del grafo.

$$\tilde{L} = \frac{2}{\lambda_{max}} L - I_N$$

Infine la regola per la propagazione di un layer è definita come: $Z = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} X \Theta$

Le ChebNets e le GCN sono molto simili, ma le GCN impongono il valore di **K=1**, per contrastare l'overfitting. Uno dei problemi delle GCN è che gli autovalori tendono a essere clusterati in un intervallo di valori molto piccolo, con dei grandi "buchi" tra ogni cluster.

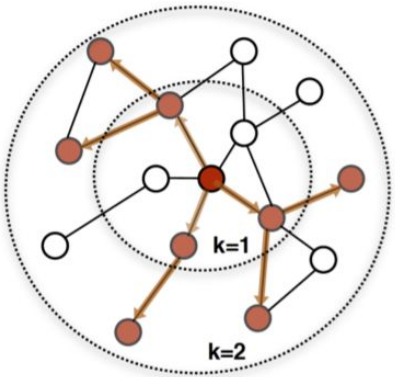
FastGCN

Data la complessità di calcolo O(N^2) della moltiplicazione fra la matrice di autovettori della Laplaciana con la matrice delle feature dei nodi si è trovato un modo (computazionalmente parlando) migliore, le FastGCN. Le FastGCN hanno modificato l'architettura di base delle GCN in modo da sfruttare un approccio **Monte Carlo (biased random sampling method)** che permette di effettuare **batch training**, andando di fatto ad alleggerire il carico complessivo computazionale e diminuendo il tempo di training.

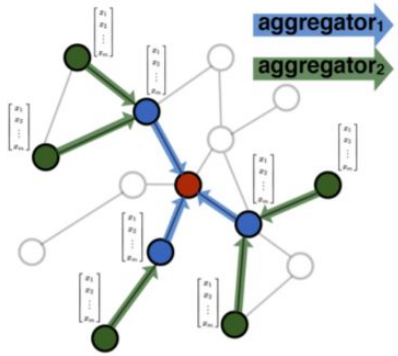
Spatial methods - GraphSage

Viene effettuato in 3 step:

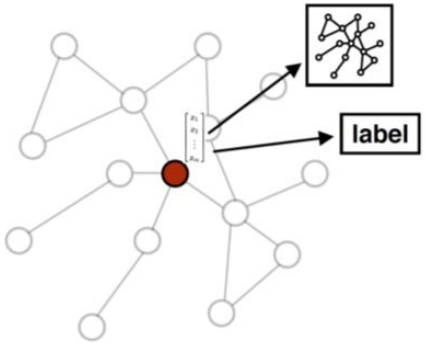
1. **Sampling del neighborhood**: dato un k definito, viene effettuato il sampling dei nodi a profondità k in modo ricorsivo
2. **Aggregazione**: Vengono aggregate le feature dei nodi di un neighborhood (possibili aggregazioni secondo il paper originale: *MeanAvg, LSTM, Pooling*)
3. **Predizione:** In modo supervisionato la rete apprende facendo classificazione dei nodi / determinando la struttura/contesto.



1. Sample neighborhood



2. Aggregate feature information from neighbors



3. Predict graph context and label using aggregated information

Papers

SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL NETWORKS

Adaptive Graph Convolutional Neural Networks