Article InfoGraph (ICLR 2020)

F.Y.Sun, J. Hoffmann, V. Verra, J. Tiang
National Taiwan Univ, MQILA, Aalto Univ, Harvard, HEC Montréal, CIFAR

Apprentissage de rep. d'un graphe dans un ensemble
Approche non supervisée (mais peut être étendue au cas ½ sup.)
Basé sur Deep InfoMax (DIM) de Hjelm et al. (ICLR 2019)

—

Préalables : papiers DIM et Nowozin et al. (NIPS 2016)

DIM maximise la MI entre l'entrée (X) et la représentation latente $E_\psi(X)$ (↳ paramètre du modèle)
L'objectif est alors de maximiser $I(X, E_\psi(X))$
tout en imposant certaines contraintes sur la loi marginale $U_{\psi,\mathbb{P}}$
$\mathbb{P}$ est la distribution empirique de X

Or, maximiser directement I est difficile et pas forcément nécessaire –
On peut passer par des estimateurs, ce qui nous permet de recourir à des fonctions de la famille des f-divergences en plus de la KL sur laquelle se base la MI
cf. papier de Nowozin et al.

En effet, on a :

$$D_f(P \| Q) \geq \sup_{T \in \mathcal{C}} \left(\underbrace{\mathbb{E}_{x\sim P}[T(x)]}_{\text{terme 1}} - \underbrace{\mathbb{E}_{x\sim Q}[f^*(T(x))]}_{\text{terme 2}} \right) \quad \text{(eq. 4)}$$

ce qui correspond à une borne inférieure de la divergence
où : f^* est la fonction duale de f (conjuguée)
et $\mathcal{C}$ une classe de fonctions arbitraires

Il peut être prouvé que la borne est très proche pour $T^* = f'\left(\frac{p(x)}{q(x)}\right)$

Si f = Jensen-Shannon (JS) alors on peut calculer :

$$D_{JS} = \frac{1}{2}\int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)}\, dx$$

Le "générateur" f est : $f(u) = -(u+1)\log\frac{1+u}{2} + u \log u$
et T^* est : $T^*(x) = \log \frac{2p(x)}{p(x)+q(x)}$

DIM

Reprenons le terme 1 de l'eq. 4 pour le cas où $p := p(x,y)$ et $q := p(x)p(y)$

$$\mathbb{E}_{x\sim\mathbb{P}}\left[T^*(\underbrace{x}_{\text{objet}}, \underbrace{E_\psi(x)}_{\text{rep. latente } y})\right] = \mathbb{E}_{\mathbb{P}}\left[\underbrace{-\log \frac{2p(x,y)}{p(x,y)+p(x)p(y)}}_{\text{(A)}}\right]$$

(A) se réécrit : $-\log \frac{p(x,y)+p(x)p(y)}{2p(x,y)}$

$$\propto -\log(1+e^{-M}) \qquad \text{si on pose : } M \overset{\text{def}}{:=} \log\left(\frac{p(x,y)}{p(x)p(y)}\right)$$

$$= -sp(-M) \qquad \text{avec } sp(z) = \log(1+e^z)$$

Même travail avec le terme 2 de l'eq. 4 :

$$\mathbb{E}_{x\sim Q}\left[f^*(T^*(x, E_\psi(x)))\right] = \mathbb{E}_Q\left[-\log\left(2 - \exp\left(\log \frac{2p(x,y)}{p(x,y)+p(x)p(y)}\right)\right)\right]$$

$\text{conf}^*(z) = -\log(2 - \exp(z))$ pour JS

(B) se réécrit : $-\log\left(2 - \frac{2p(x,y) + 2p(x)p(y) - 2p(x)p(y)}{p(x,y) + p(x)p(y)}\right)$

$= -\log\left(\frac{2p(x)p(y)}{p(x,y) + p(x)p(y)}\right)$

$\propto -\log \frac{1}{e^M + 1} = \log(e^M + 1) = sp(M)$

On peut enfin réécrire la borne en suivant l'estimateur JS :

$$\hat{I}_\theta^{(JSD)}(X, E_\psi(X)) := \mathbb{E}_{\mathbb{P}}[-sp(-M(x, E_\psi(x)))] - \mathbb{E}_{\textcircled{A}}[sp(M(x', E_\psi(x)))] \quad \} (1)$$

(A) $\hookrightarrow \tilde{\mathbb{P}} \times \mathbb{P}$

- $\tilde{\mathbb{P}}$: feature « local » d'un objet distinct de x
- $\mathbb{P}$: feature « global » associé à x

Une autre interprétation :

$$\hat{I}_\theta^{(JSD)} := \mathbb{E}_{\mathbb{P}}[-sp(-M(x, E_\psi(x)))] - \mathbb{E}_{\tilde{\mathbb{P}} \times \mathbb{P}}[sp(M(x', E_\psi(x)))]$$

- M : « discriminateur »
  - valeur ↗ si MI forte entre x et $E_\psi(x)$
  - valeur ↘ si MI faible entre x' et $E_\psi(x)$
- E_ψ : encodeur, paramétré par ψ

On remplace M par une fonction paramétrique (un réseau de neurones) : M_ω (cf. Belghazi et al., cf. plus loin)

donc $\theta = \{\psi, \omega\}$

Au final, on cherche $\hat{\theta}$ t.q. :

$$\hat{\theta} = \arg\max_\theta \hat{I}_\theta^{(JSD)}(X, E_\psi(X)) \quad \} (2)$$

DIM repose sur le principe **InfoMax** (Linsker 1988 ; Bell & Sejnowski, 1995) qui consiste à maximiser la MI entre l'entrée X et la sortie d'une fonction comme un réseau calculant une rep. latente Z, ici $E_\psi(X)$ en suivant un principe d'apprentissage adverse (adversarial training) inspiré des GAN et AAE (Makhzani et al., 2015) pour contraindre l'espace latent

L'adv. training agit comme une forme de régularisation dans l'espace appris par E

L'important est d'être capable d'échantillonner :

$\mathbb{P}$: $(x, E_\psi(x))$, avec x pouvant être « une partie » (feature local) de l'objet et $E_\psi(x)$ la représentation globale de X

$\tilde{\mathbb{P}} \times \mathbb{P}$: $(x', E_\psi(x))$, avec x' pouvant être une partie d'un autre objet X' = exemple adverses

Plusieurs remarques :

- M, la fonction qui calcule la MI, est approximée par une famille de fonctions paramétrées en l'occurrence un discriminateur D_ω : $\mathbb{P} \times \mathbb{P} \to \mathbb{R}$

Dans le papier MINE (Belghazi et al.), ce point est expliqué en expliquant le fait que :
$I(X;Z) \geq I_\theta(X;Z)$, où I_θ est appelée la MI neuronale

- Il existe des alternatives à D_{JS}, comme D_{DV} et D_{NCE} (pour Negative Contrastive Est mais D_{JS} semble + robuste et nécessiter moins d'exemples négatifs.
- Au sujet de la contrainte de rapprochement entre la distribution latente apprise $\mathbb{V}$ et celle à priori $\mathbb{U}$:
  a) entraînement du "discriminateur" D_ω qui estime $D(\mathbb{V}\|\mathbb{U})$ → attention, ≠ du discriminateur précédent --.
  b) entraînement de l'encodeur pour minimiser cette estimation

$$(\hat{\omega}, \hat{\psi}) = \arg\min_\psi \arg\max_\omega \hat{D}_\omega(\mathbb{V}\|\mathbb{U}) = \mathbb{E}_{\mathbb{U}}[\log D_\omega(y)] + \mathbb{E}_{\mathbb{P}}[\log(1 - D_\omega(E_\psi(x)))]$$

C'est une stratégie similaire utilisée pour les GAN et AAE

Au final, DIM cherche à optimiser une fonction à trois objectifs : (notez x ou x')

$$\arg\max_{\omega_1,\omega_2,\psi} \left(\alpha \underbrace{\hat{I}_{\omega_1,\psi}(X; E_\psi(X))}_{\text{global}} + \frac{\beta}{n} \sum_{i=1}^{n} \underbrace{\hat{I}_{\omega_2,\psi}(X^{(i)}; E_\psi(X))}_{\text{local-global}} \right)$$

$$+ \underbrace{\arg\min_\psi \arg\max_\omega \gamma \hat{D}_\omega(\mathbb{V}\|\mathbb{U})}_{\text{contrainte}}$$

InfoGraph : – basé sur DIM
– peut être étendu au cas ½ supervisé (non traité ici)

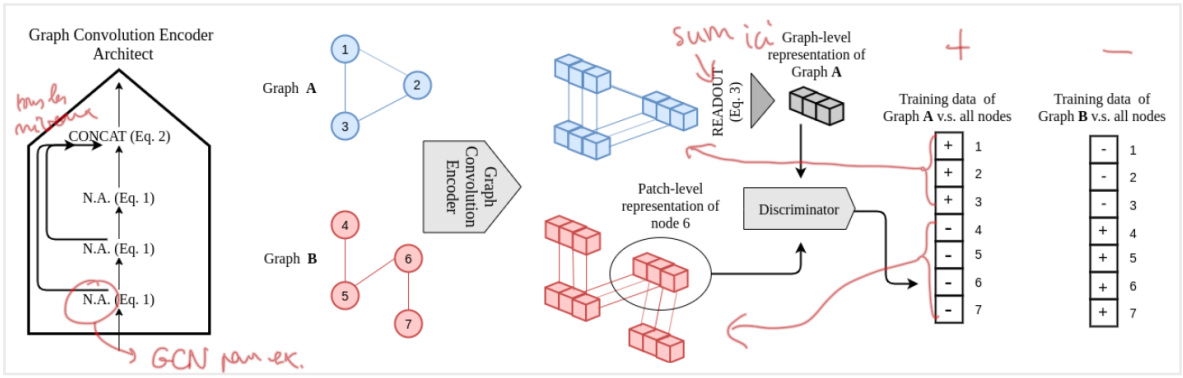
Etant donné un ensemble de graphes $\mathcal{G} = \{G_1, G_2 \dots\}$
et δ la taille de l'embedding à apprendre
L'objectif est d'apprendre $\Phi \in \mathbb{R}^{|\mathcal{G}| \times \delta}$ càd une rep. par chaque graphe
InfoGraph est basé sur un GNN (ici GIN de Xu et al. 2018)
Si $h_i^{(k)}$ est la représentation vectorielle du noeud i à l'étape (couche) k
Alors : $h_\phi^i = \text{CONCAT}(\{h_i^{(k)}\}_{k=1}^{K})$ est la rep. « multi-échelle » de i
et $H_\phi(G) = \text{READOUT}(\{h_\phi^i\}_{i=1}^{N})$ est l'agrégation des h_ϕ^i
qui permet d'obtenir une représentation globale pour G
(params. du GNN)

Dans le contexte DIM, on a également Ψ qui correspond au discriminateur qui va estimer à quel point un élément X (patch) fait bien partie de l'objet global
On cherche à optimiser $\Theta = \{\Phi, \Psi\}$ avec une fonction de coût basée sur l'inf mutuelle

$$\hat{\Phi}, \hat{\Psi} = \arg\max \sum_{G \in \mathcal{G}} \frac{1}{|\mathcal{G}|} \sum_{u \in G} I_{\Phi,\Psi}(h_\Phi^u; H_\Phi(G)) \quad \text{cf. (2)}$$

patch — info globale

Sachant que I est exactement calculée via (1)


Graph Convolution Encoder Architect
CONCAT (Eq. 2)
N.A. (Eq. 1)
N.A. (Eq. 1)
N.A. (Eq. 1)
Graph A
Graph B
Graph Convolution Encoder
READOUT (Eq. 3)
Graph-level representation of Graph A
Patch-level representation of node 6
Discriminator
Training data of Graph A v.s. all nodes
Training data of Graph B v.s. all nodes

En non sup, l'approche est utilisée pour avoir des rep. de graphes qui sont utilisés pour faire de la classification (entre 2 et 5 classes, jusqu'à 5000 graphes). La classif est faite avec LIBSVM en faisant varier C, mais pas d'info sur le kernel

---

Deep Graph InfoMax (Velickovic et al., ICLR 2019)

Publié de manière concomitante.

DGI est aussi basé sur DIM pour maximiser la MI entre une représentation globale et une représentation locale au niveau des « patch » (init. pour les images).
Ça encourage donc une info globale présente un peu partout dans l'objet.

DGI est aussi une approche contrastive car on apprend à distinguer (classification) entre des paires (info locale, info globale) qui sont associées ou non au même objet

L'encodeur est basé sur un GCN pour apprendre des $h_i \in \mathbb{R}^{F'}$, F' = taille espace latent
(h_i = patch representation)

La représentation globale, appelée ici « summary vector », est calculé une fois encore avec une fonction appelée READOUT $\mathcal{R}$:

$$\vec{s} = \mathcal{R}(\mathcal{E}(X, A))$$

($\mathcal{R}$: readout, $\mathcal{E}$: encoder, X : features, A : relations)

À noter que je n'ai pas trouvé comment est finalement calculé $\mathcal{R}$...
Comme proxy pour maximiser la MI, ils utilisent un discriminateur

$D : \mathbb{R}^F \times \mathbb{R}^F \to \mathbb{R}$ t.q. $D(\vec{h}_i, \vec{s})$ est un score de proba :

- → élevé si i appartient au graphe résumé
- → faible sinon (exemples négatifs)

Les exemples négatifs sont générés ici avec une procédure de corruption.

La fonction objectif est inspirée par DIM mais retranscrit par une cross entropy

(3)
$$\mathcal{L} = \frac{1}{N+M}\left(\sum_{i=1}^{N} \mathbb{E}_{(X,A)}\left[\log D(\vec{h}_i, \vec{s})\right] + \sum_{j=1}^{M} \mathbb{E}_{(\tilde{X},\tilde{A})}\left[\log\left(1 - D(\tilde{h}_j, \vec{s})\right)\right]\right)$$

(N : paires +, M : paires −, $\mathbb{E}_{(X,A)}$: paires +, $\mathbb{E}_{(\tilde{X},\tilde{A})}$: paires −)

C'est très similaire à (1) + (2) utilisé dans InfoGraph.

Un bien est fait avec les GAN qui optimisent une fonction similaire.
Des dérivations théoriques montrent que optimiser (3) revient à optimiser la MI