République Algérienne Démocratique et Populaire
Ministère de l'enseignement Supérieur et de la Recherche Scientifique
Université de Mohamed El Bachir El Ibrahimi de Bordj Bou Arréridj
Faculté des Mathématiques et d'Informatique
Département d'informatique

MEMOIRE

Présenté en vue de l'obtention du diplôme

Master en informatique

Spécialité : Réseaux et Multimédia

THEME

Fouille d'interaction dans les réseaux complexes multicouches

Présenté par :

Saiyem Nassime

Galoul Aymen

Soutenu publiquement le : 03/07/2023

Devant le jury composé de:

Président : Mme.Fares Nour Elhouda

Examinateur : Mme.Laifa Meriem

Encadreur : Mr.Charikhi Mourad

2022/2023

Dédicace

Nous *dédions ce travail qui n'aura jamais pu voir le jour sans les soutiens indéfectibles et sans limite de nos chers parents qu'ils étaient la base de nos existence et le bonheur infini.*

À *nos chers frères et sœurs , chers tantes et chers oncles, ceux qui n'ont pas hésité à nous donner de l'amour nécessaire pour arriver à ce que nous somme aujourd'hui.*

A *tous nos amies pour son soutien moral.*

Remerciement

Nous remercions dieu pour nous avoir donné santé, courage et patience afin de nous aider àréaliser ce travail.

Au terme de ce modeste travail nous tentons à remercier chaleureusement et respectivementtous ceux qui ont contribué de près ou de loin à la réalisation de ce modeste projet de fin d'étude, Nous tentons aussi à remercier particulièrement notre encadreur Mr. Charikhi Mourad de nousavoir suivis et guidés tout au long de réalisation de notre travail.

On remercie vivement mes dames et monsieurs les membres de jury d'avoir accepté d'évaluer notre modeste travail.

Résumé

La prédiction des liens, dont l'objectif est de comprendre en profondeur la structure des réseaux, est l'un des sujets de recherche les plus en vogue dans le domaine de l'analyse des réseaux sociaux.

L'objectif de ce travail est de proposer une approche qui facilite et améliore les méthodes de la prédiction des liens, manquants ou futurs dans un réseau, basée sur une approche de similarité. La solution que nous proposons dans ce mémoire se base, dans un premier temps, sur l'extraction des caractéristiques des nœuds et des liens et les combine, par la suite avec les caractéristiques de la structure du graphe afin de générer à la fin des caractéristiques optimales qui seront utilisés pour prédire les liens manquants ou futurs.

Les résultats de cette approche, dont les performances ont été comparées avec d'autres algorithmes de prédiction de liens, ont été testés sur différents types de réseaux.

Abstract

Link prediction, which aims to deeply understand the structure of networks, is one of the most popular research topics in the field of social network analysis.

The objective of this work is to propose an approach that facilitates and improves methods for predicting missing or future links in a network, based on similarity approach. The solution we propose in this paper is based, firstly, on extracting features from nodes and links and subsequently combining them with the characteristics of the graph structure to generate optimal features that will be used to predict missing or future links.

The results of this approach, whose performance has been compared with other link prediction algorithms, have been tested on different types of networks.

ملخص

توقع الروابط، الذي يهدف إلى فهم هيكل الشبكات بعمق، هو واحد من أهم مواضيع البحث في مجال تحليل الشبكات الاجتماعية.

يهدف هذا العمل إلى اقتراح نهج يسهل ويحسن طرق توقع الروابط المفقودة أو المستقبلية في شبكة ما، بناءً على نهج التشابه. تعتمد الحلول التي نقترحها في هذه الرسالة، في المرحلة الأولى، على استخراج سمات العقد والروابط ودمجها مع سمات هيكل الرسم البياني لتوليد تقنيات مثلى في النهاية سيتم استخدامها لتوقع الروابط المفقودة أو المستقبلية.

تم اختبار نتائج هذا النهج، وتمت مقارنة أدائها مع خوارزميات أخرى لتوقع الروابط، على أنواع مختلفة من الشبكات.

Table des matières

Liste des figures

Liste des tableaux

Introduction générale

Les réseaux complexes multicouches sont des structures de réseau qui se composent de plusieurs couches d'interactions, où les nœuds et les liens peuvent avoir différentes caractéristiques dans chaque couche. Ces réseaux sont omniprésents dans de nombreux domaines, tels que les réseaux sociaux, les réseaux biologiques, les systèmes de transport et bien d'autres encore. La prédiction des liens dans ces réseaux multicouches vise à identifier et à prédire les connexions manquantes ou les relations potentielles entre les nœuds, ce qui peut avoir une importance cruciale pour comprendre le fonctionnement du réseau et faciliter la prise de décision.

Les approches de prédiction des liens dans les réseaux complexes multicouches se divisent généralement en deux catégories principales : les approches basées sur la structure et les approches basées sur les attributs.

Les approches basées sur la structure exploitent la topologie du réseau pour découvrir des motifs et des régularités qui peuvent prédire les liens manquants. Ces méthodes utilisent des algorithmes de propagation d'informations, tels que la diffusion de l'information ou la propagation de l'activation, pour estimer les probabilités de connexion entre les nœuds. Elles se basent également sur des mesures de similarité, telles que la similarité de voisinage ou la similarité de chemins, pour évaluer la probabilité de connexion entre les nœuds.

D'autre part, les approches basées sur les attributs utilisent les caractéristiques des nœuds et des liens dans différentes couches du réseau pour prédire les liens. Ces caractéristiques peuvent inclure des attributs démographiques, des informations de localisation, des centres d'intérêt, des caractéristiques biologiques, etc. Les méthodes d'apprentissage automatique, telles que les réseaux de neurones, les machines à vecteurs de support (SVM) ou les arbres de décision, sont souvent utilisées pour construire des modèles prédictifs en exploitant ces attributs

Il est important de noter que les réseaux complexes multicouches sont souvent dynamiques, c'est-à-dire que les liens et les attributs des nœuds peuvent évoluer dans le temps. Par conséquent, les techniques de prédiction des liens doivent également prendre en compte cette dimension

temporelle. Les modèles de prédiction des liens dynamiques intègrent des éléments de modélisation temporelle, tels que les processus de Markov cachés (HMM), les modèles de diffusion ou les méthodes de prévision temporelle, pour capturer les changements et les évolutions du réseau au fil du temps.

L'évaluation des méthodes de prédiction des liens dans les réseaux complexes multicouches se fait à l'aide de différentes mesures. Parmi les mesures les plus couramment utilisées, on retrouve la précision, le rappel, la F-mesure, l'AUC (Area Under the Curve), la courbe de précision-rappel. Ces mesures permettent d'évaluer la qualité des prédictions par rapport aux liens réels présents dans le réseau et de comparer différentes méthodes de prédiction.

La prédiction des liens dans les réseaux complexes multicouches revêt une grande importance dans de nombreux domaines d'application. Par exemple, dans les réseaux sociaux, cela peut aider à recommander de nouveaux amis ou à prédire les interactions futures entre les utilisateurs. Dans les réseaux biologiques, cela peut aider à identifier les interactions protéine-protéine manquantes ou à prédire les relations fonctionnelles entre les gènes. Dans les systèmes de transport, cela peut faciliter la planification des itinéraires ou la prévision de la congestion du trafic.

En résumé, la prédiction des liens dans les réseaux complexes multicouches est un domaine de recherche passionnant qui permet de mieux comprendre les interactions entre les nœuds et de prédire les connexions manquantes ou les relations potentielles. Les approches basées sur la structure et les attributs, ainsi que les modèles de prédiction dynamiques, constituent des méthodes couramment utilisées dans ce domaine. Ces avancées ont un impact significatif dans de nombreux domaines d'application, offrant de nouvelles perspectives pour la modélisation, la prédiction et l'analyse des réseaux complexes multicouches.

Il existe plusieurs solution presente avec une abilitée d'amelioration dans la predicition des liens et la vitesse du processus montionées precedement, le probleme qui s'impose dans ces solutions c'est que leurs resultats n'est pas aussi exacte et leurs redaction n'est pas aussi vite, donc on a essayé de manipuler quelque algorithmes critiques pour accelerer performance et renforcer les resultats. On va presenter notre travaille dans quatre chapitre dont le premier chapitre nous allons inteoduire les graphes et les reseaux complexes, dans le deuxieme chapitre on prsentera les methodes et les outils de prediction de liens, le troisieme chapitre est consacrer pour definir les algorithmes d'une methodes et nos algorithmes devloppés et le quatrieme chapitre est la discussion des resultats

Chapitre 1

Introduction aux graphes et réseaux complexes

1. Introduction

Dans ce chapitre nous présentons les concepts fondamentaux des graphes ainsi que les différents types de réseaux complexes. Nous détaillons par la suite les notions de bases inhérentes au domaine de la prédiction des liens dans les réseaux sociaux.

2. Présentation des notions générale des graphes

Un graphe est un ensemble de sommets (points ou nœuds) désignés par V et d'arcs (lien dirigées) ou d'arêtes (lien non dirigées) désignés comme reliant des paires particulières de points. La notation usuelle est : |V|= N (nombre de nœuds), |E| = M (nombre de liens). Grâce à la fonction de poids m : $\boldsymbol{E} \rightarrow \boldsymbol{\mathcal{R}}+$ nous pouvons pondérer les arêtes du graphe, ce qui nous permet de modéliser plus en détail les interactions entre sommets. Nous obtenons donc le graphe pondéré G = (V, E, m). Le poids de l'arête $\{\boldsymbol{i}, \boldsymbol{j}\}$ entre deux sommets i et j est noté $\boldsymbol{mij}$. Par convention, aucune arête ne se voit attribuer un poids de 0 ($\boldsymbol{mij} = \boldsymbol{0}$ $\{\boldsymbol{i}, \boldsymbol{j}\} \notin$ E). Pour les graphes non pondérés, les poids des arêtes dans E sont fixés à 1, donc dans ce cas particulier $\forall \boldsymbol{ij} \in \boldsymbol{V}$, $\boldsymbol{mij} \in \{\boldsymbol{0}, \boldsymbol{1}\}$.

• Le degré d(v) d'un sommet v $\in \boldsymbol{V}$ est le nombre d'arêtes appartenant au sommet v. C'est le nombre de sommets adjacents dans v. Nous définissons également le poids m(i) du sommet i comme la somme des poids de ces arêtes incidentes (**i**) = $\sum \boldsymbol{mij}; \boldsymbol{j} \in \boldsymbol{V}$.

Notez que pour les graphiques non pondérés, les poids des sommets correspondent aux définitions des sommets. • Pour un graphe G = (V, E), la distance d'un sommet V à un autre sommet est appelée la longueur du plus court chemin/chaîne entre ces deux sommets. Appelez-le 1 si aucun chemin/chaîne de ce type n'existe. Le diamètre d'un graphe est la distance maximale qui peut exister entre ses deux sommets. • La densité du graphe est définie comme $\boldsymbol{2m}$, qui est le rapport du nombre d'arêtes à la densité du graphe.

($\boldsymbol{n}-\boldsymbol{1}$) est le nombre maximal d'arêtes possibles compte tenu du nombre de sommets dans le diagramme. Si deux sommets d'un graphe non orienté sont reliés par une arête, ils sont dits adjacents ou contigus. Dans un graphe non orienté G, c'est le voisinage d'un sommet v ∈ V, souvent noté $\boldsymbol{NG}$(v).

Il peut représenter tous ces sommets adjacents ou un sous-graphe associé. Les graphes orientés utilisent généralement les termes prédécesseur ou successeur. Si un graphe G est représenté par une matrice d'adjacence A, et G est pondéré non orienté, alors A est défini comme $\mathbf{0}\ \boldsymbol{si}\ \{\boldsymbol{i}, \boldsymbol{j}\} \notin \boldsymbol{E}$ et $\boldsymbol{Aij}$ = W$\boldsymbol{ij}\ \boldsymbol{si}\ \{\boldsymbol{i}, \boldsymbol{j}\} \in \mathbf{E}$.

Pour un graphe non pondéré $\boldsymbol{Aij} \in \{\mathbf{0}, \mathbf{1}\}$ (car nous fixons le poids $\boldsymbol{wij}$ à 1 pour toutes les arêtes de E.

3. Types des graphes

1. Graphe non orienté : Dans un graphe non orienté, les arêtes n'ont pas de direction. Les relations entre les nœuds sont symétriques, ce qui signifie que si le nœud A est connecté au nœud B, alors le nœud B est également connecté au nœud A.
2. Graphe orienté : Dans un graphe orienté, les arêtes ont une direction définie. Les relations entre les nœuds sont asymétriques, ce qui signifie que si le nœud A est connecté au nœud B, cela ne garantit pas que le nœud B est également connecté au nœud A.
3. Un graphe est simple si au plus une arête relie deux sommets et s'il n'y a pas de boucle surun sommet. On peut imaginer des graphes avec une arête qui relie un sommet à lui-même(une boucle), ou plusieurs arêtes reliant les deux mêmes sommets. On appelera ces graphesdes multigraphes.
4. Un graphe est connexe s'il est possible, à partir de n'importe quel sommet, de rejoindre tous les autres en suivant les arêtes. Un graphe non connexe se décompose en composantes connexes.
5. Graphe complet : Un graphe complet est un graphe non orienté dans lequel chaque paire de nœuds est reliée par une arête. Autrement dit, il y a une arête entre chaque paire de nœuds distincts.
6. Un graphe aléatoire est un graphe qui est généré par un processus aléatoire. Il est caractérisé par la distribution des degrés suivant une loi puissance, fort nombre de triangles et aussi par la densité de ces types des graphes. Elle dite petite si les degrés des sommets sont petits comparé à la taille du graphe.

Les graphes aléatoires sont des modèles pour étudier les grands graphes comme les graphes des réseaux sociaux, biologique, information et technologie etc.… **La figure 1** présente la structure d'un graphe.

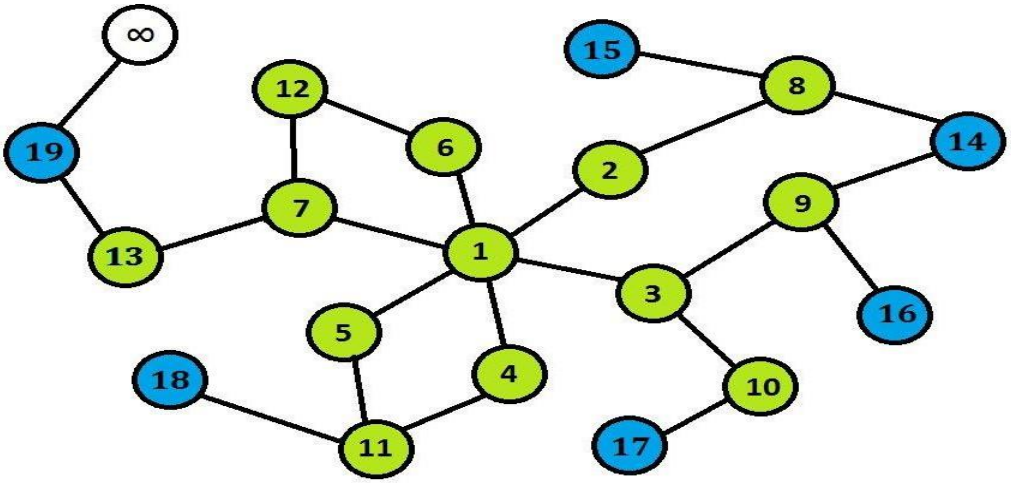

Figure 1 structure d'un graphe

4. Réseaux complexe

Les diagrammes sont couramment utilisés pour représenter tous les types de réseaux, y compris les systèmes complexes du monde réel tels que les réseaux biologiques, les réseaux sociaux, les réseaux cibles, le World Wide Web (WWW) et les réseaux de communication.

Un réseau complexe est un schéma constitué de nœuds pouvant être reliés par des liens (personnes, organisations, objets) qui sont des interactions ou des relations. Ainsi, des données sont collectées qui correspondent à la vérité, comme les réseaux biologiques (interactions protéine-protéine, réseaux de neurones, réseaux de gènes, etc.).

La section suivante présente plusieurs types de réseaux complexes.

4.1. Les réseaux sociaux

Le terme "réseau social" a été utilisé par le juge Burns en 1954 pour décrire les relations humaines dans les salles de classe et les comités de l'église sur l'île norvégienne [2,3]. Un réseau social est considéré comme une structure sociale composée de différents nœuds de réseau. Chaque "nœud" représente un individu ou une organisation.

En général, un réseau social est une carte de tous les nœuds et liens étiquetés comme dans la figure 1.2. Chaque nœud représente une seule entité. Ce sont soit des individus, soit des groupes.

De nombreuses connexions/liens relient les nœuds les uns aux autres. Ces liens comprennent des relations telles que la famille, les amis et les collègues. En étudiant les graphes et leurs processus évolutifs dynamiques, nous pouvons généralement trouver des informations précieuses pour résoudre des problèmes pratiques dans le monde réel [4,5]. Un réseau social est défini comme un réseau d'interactions ou de relations où les nœuds sont constitués d'acteurs et les bords sont constitués de relations ou d'interactions entre acteurs [1]. Une généralisation de l'espace des réseaux sociaux est l'espace des réseaux d'information, où les nœuds contiennent des acteurs et des entités et dont les contours montrent les relations entre eux. Il est clair que la notion de réseau social ne se limite pas au cas particulier des réseaux sociaux basés sur Internet comme Facebook. Il existe deux catégories de réseaux sociaux. La première catégorie concerne les réseaux sociaux humains et la seconde catégorie concerne les réseaux sociaux en ligne. En sociologie, le problème des réseaux sociaux est souvent considéré en termes d'interactions générales entre tous les groupes de sujets. De telles interactions peuvent prendre toute forme conventionnelle, telles que des interactions entre différents contacts, des interactions de télécommunication, des interactions par courrier électronique, des post-interactions [1]. La figure 2 ci-dessous montre la structure d'un réseau social.

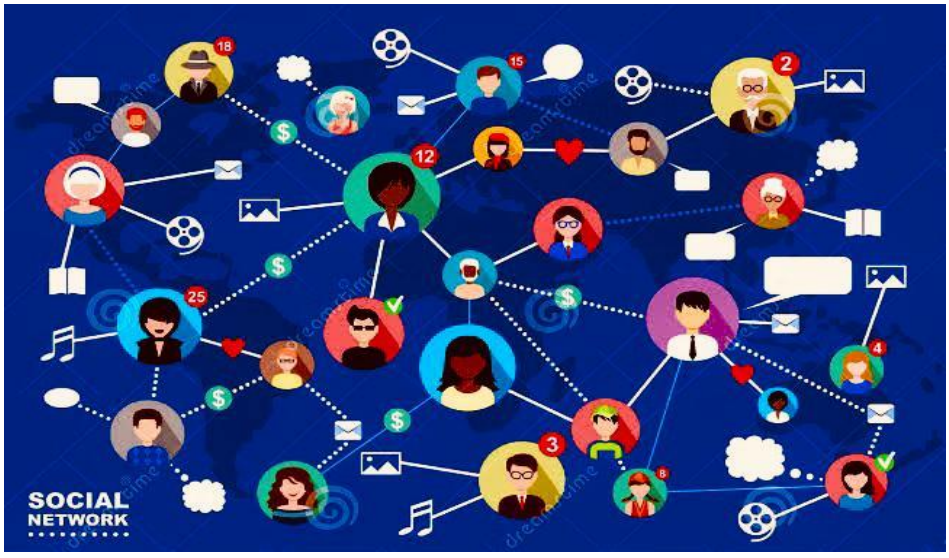

Figure 2 Structure d'un réseau social

4.2. Les réseaux d'informations

Les réseaux d'information sont des réseaux complexes qui modélisent les flux d'information entre différentes entités, telles que les personnes, les documents, les sites web, les médias sociaux, etc. Ces réseaux permettent de représenter et d'analyser la diffusion, la propagation et la circulation de l'information dans divers contextes.

Les réseaux d'information peuvent être construits à partir de différentes sources de données, telles que les citations dans les publications scientifiques, les liens hypertextes entre les pages web, les interactions sur les médias sociaux, les recommandations de produits ou de contenus, etc. Ils sont généralement représentés sous la forme de graphes, où les nœuds représentent les entités (par exemple, les auteurs, les documents, les utilisateurs) et les arêtes représentent les liens ou les relations entre ces entités (par exemple, les citations, les liens hypertextes, les interactions sociales).

4.3. Les réseaux biologiques

Les biologistes rencontrent des réseaux métaboliques [19], des réseaux d'interaction protéine-protéine [20] ou des réseaux de régulation génique [21] qui modélisent les processus de production et de dégradation de matière et d'énergie dans les organismes vivants.

Vous pouvez aussi les citer Nourriture Internet. Ces réseaux présentent des distributions sans échelle selon les propriétés des réseaux complexes, notamment les études topologiques dont ils font l'objet [22,23].

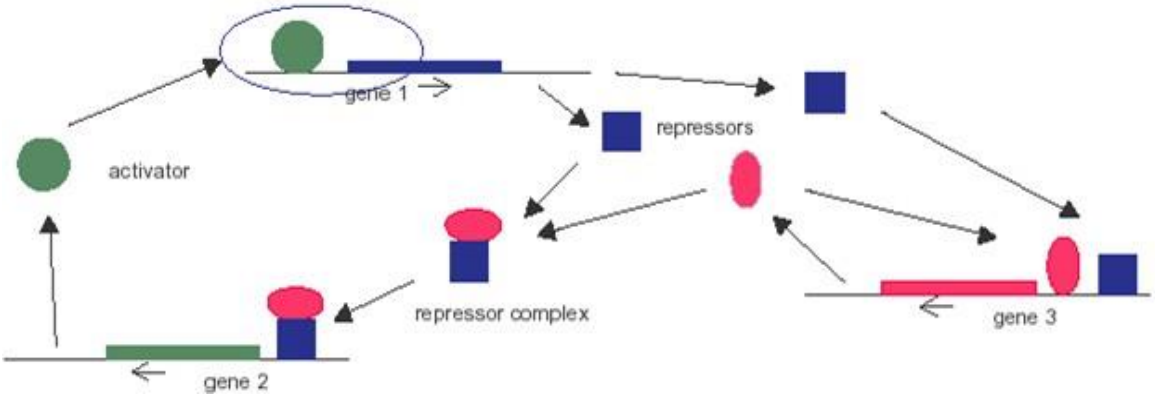

Figure 3 structure d'un réseaux biologiques

4.4 Les réseaux technologiques

Les réseaux technologiques sont des réseaux complexes qui représentent les interconnexions entre les objets technologiques tels que les ordinateurs, les appareils mobiles, les serveurs, les routeurs, les capteurs, etc. Ces réseaux sont essentiels pour permettre la communication, le partage de ressources et l'échange d'informations dans le domaine de la technologie de l'information.

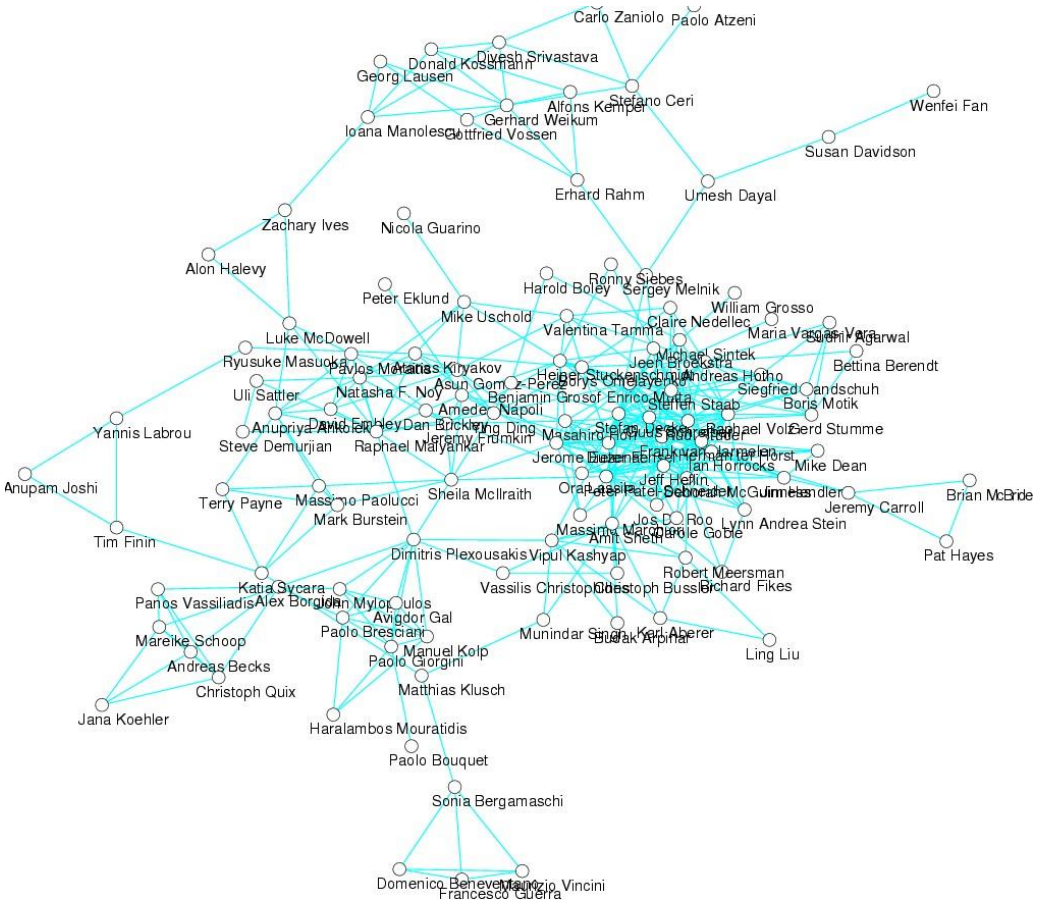

figure 4 un réseau social d'un site web de communauté

5. conclusion

Ce premier chapitre a introduit les concepts les plus importants liés aux graphes et aux différents types de réseaux complexes. Nous avons également défini la prédiction de liens et ses concepts de base. Le chapitre suivant, Chapitre 2, présente l'état de l'art lié à la prédiction de liens et les différentes méthodes utilisées dans ce domaine.

Chapitre 2

Etat de l’art

❖ Introduction :

La prédiction des liens est une tâche visant à anticiper les relations et interactions au sein d'un réseau. Les techniques d'apprentissage automatique sont utilisées pour prédire les liens manquants ou futurs entre les nœuds d'un graphe. L'objectif principal est de prédire l'évolution des liens qui ne sont pas encore observés dans l'état actuel du réseau. Cette tâche revêt une grande importance et a attiré l'attention de chercheurs issus de différentes disciplines. Au fil des dernières années, un grand nombre de méthodes ont été proposées pour résoudre ce problème. Ces méthodes se distinguent par plusieurs aspects, tels que les changements dans l'évolution, le type d'information traitée et sa quantité.

Dans ce chapitre, nous réalisons une étude approfondie des approches de prédiction de liens et nous nous efforçons de présenter les principales méthodes existantes. Pour ce faire, nous avons choisi de classifier les méthodes de prédiction de liens en deux catégories principales : les approches heuristiques et les approches d'apprentissage automatique. À travers cette classification, nous effectuons une comparaison des méthodes proposées, qui sera illustrée dans un tableau récapitulatif mettant en avant les défis et les avantages de chaque méthode.

1. Prédiction de lien :

Les réseaux sont de plus en plus utilisés pour modéliser des systèmes complexes composés d'éléments interagissant les uns avec les autres, tels que les réseaux sociaux et les réseaux biologiques mentionnés dans les sections précédentes. Des études ont démontré qu'il est possible de prédire de nouvelles relations entre les éléments présents dans la topologie d'un réseau. Cette problématique, connue sous le nom de prédiction de liens, consiste à rechercher de nouvelles relations au sein des réseaux. Son objectif est de prédire le comportement des liens, c'est-à-dire déterminer si une relation peut être établie entre deux éléments d'un réseau ou si une relation entre eux est manquante, en se basant sur les relations déjà observées.

En raison de l'applicabilité de cette thématique, de nombreuses études et recherches se sont concentrées sur ce domaine. Ainsi, plusieurs méthodes ont été développées et appliquées pour la recherche et la prédiction de liens dans différents types de réseaux.

1.1. Intérêt de la prédiction de liens dans les réseaux sociaux :

L'analyse des réseaux sociaux est devenue un sujet de recherche très populaire dans le domaine de l'informatique. Il est largement reconnu que la prédiction dans les réseaux sociaux est un domaine complexe et difficile, en particulier pour les réseaux sociaux en ligne qui peuvent compter des millions, voire des milliards, d'utilisateurs et de connexions. De plus, les données dans les réseaux sociaux en ligne sont extrêmement dynamiques. Les activités sociales des utilisateurs dans ces réseaux sont souvent imprévisibles, avec des utilisateurs qui rejoignent ou quittent le réseau et des connexions qui se forment ou se rompent à tout moment.

Les relations présentes dans les réseaux sociaux peuvent être extrêmement diverses. Différents types de réseaux présentent différents types de relations, avec des degrés de force variés, des orientations spécifiques, et ainsi de suite. Comprendre et modéliser cette diversité des relations est un défi majeur dans l'analyse des réseaux sociaux. [17]

Ces caractéristiques complexes des réseaux sociaux en font un domaine de recherche stimulant et en évolution constante. Les chercheurs explorent de nombreuses approches, techniques et modèles pour tenter de comprendre et de prédire les comportements et les dynamiques des réseaux sociaux.

Si nous pouvons prédire avec précision les limites qui seront créées entre deux nœuds du réseau pendant un intervalle de temps allant de t à un temps futur donné t’(t'> t) [18], nous pouvons comprendre comment un réseau social évolue et quel est la dynamique qui est derrière

Effectivement, les liens dans un réseau social sont le reflet du maintien et de la qualité des relations, ce qui peut fournir des informations précieuses sur les comportements sociaux des individus et des communautés. Dans cette ère de l'information où de plus en plus de personnes participent à des communautés en ligne ou hors ligne, comme les clubs sportifs, la recherche en

prédiction de liens devient extrêmement pertinente. Elle permet d'évaluer quantitativement et qualitativement les relations humaines.

La prédiction de liens revêt une importance capitale dans l'analyse des réseaux sociaux. Elle permet de mieux comprendre les dynamiques et les interactions au sein des réseaux sociaux, en prédisant les relations qui peuvent émerger ou qui sont manquantes. Ainsi, de nombreux algorithmes et méthodes de prédiction de liens sont appliqués à une grande variété de réseaux, afin d'apporter des éclairages sur les relations humaines dans divers contextes.

En utilisant ces techniques de prédiction de liens, les chercheurs peuvent obtenir des informations précieuses sur la structure et la dynamique des réseaux sociaux, contribuant ainsi à une meilleure compréhension des comportements sociaux et à une évaluation approfondie des relations humaines.

1.2. Définition formelle de prédiction de liens

Etant donné un réseau G_t**= (V,)** à un instant donné **t**, nous devons prévoir l'ensemble des nouveauxliens **E** qui apparaîtront probablement dans le réseau dans l'intervalle de temps **[t,′]**, où t'**> t**. Le réseau $G_{t'}$ à l'instant t'peut être représenté par :′ **= (V,′) où** $E_{t'} = E_t \cup E$

Il est important de noter que, dans le problème de la prédiction de liens, **V** reste statique avec l'heure. En revanche, **E** varie d'une heure à l'autre à mesure que de nouveaux liens sont ajoutés au réseau **[7]**. De nombreux ensembles de données peuvent naturellement être représentés sous forme de graphique où les nœuds **V** représentent les instances et les liens **E** représentent les relations entre ces instances **[8].**

Des liens peuvent être manquants entre deux nœuds liés ou des liens qui peuvent être créent à l'avenir **[8]**.L'objectif de la prédiction de lien est donc de prédire l'existence de ces liens futurs ou manquants entre lesnœuds du graphe. **Figure 2.1** illustre la prédiction des liens entre plusieurs instants.

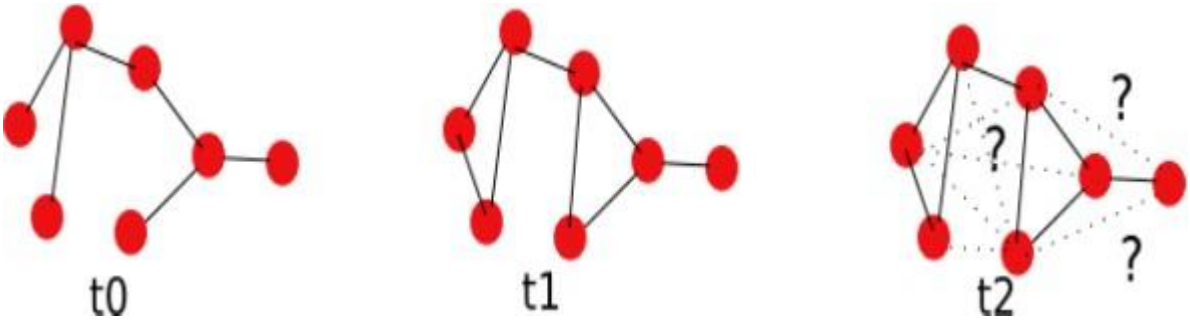

Figure *5* : la prédiction de lien dans les instants t1, t2

1.3. Catégories des modèles de prédiction de lien

Nous citons dans ce qui suit, les deux principales catégories de modèles de prédiction de lien utilisées à savoir, la prédiction des liens manquants et la prédiction des liens futurs.

❖ Prédiction des liens manquants (Link Prediction):

La prédiction des liens manquants vise à estimer les liens qui pourraient exister entre des paires de nœuds dans un réseau, mais qui ne sont pas encore présents ou observés. Cette catégorie de modèles se concentre sur l'identification des relations potentielles entre les nœuds qui ne sont pas directement connectés. Les méthodes de prédiction des liens manquants utilisent généralement des mesures de similarité entre les nœuds, des modèles probabilistes ou des techniques d'apprentissage automatique pour évaluer la probabilité ou la présence potentielle d'un lien entre deux nœuds donnés. La formule pour la prédiction des liens manquants (Link Prediction) est donnée par:

P(u, v) = f(x(u), x(v)). Elle permet d'estimer la probabilité ou la mesure de similarité entre les nœuds u et v, indiquant la présence potentielle d'un lien entre eux. Les caractéristiques des nœuds, x(u) et x(v), sont utilisées comme entrées pour une fonction f() qui calcule cette probabilité ou mesure de similarité. La formule pour la prédiction des liens manquants (Link Prediction) est donnée *par*

P(u, v) = f(x(u), x(v)). Elle permet d'estimer la probabilité ou la mesure de similarité entre les nœuds u et v, indiquant la présence potentielle d'un lien entre eux. Les caractéristiques des nœuds, ***x(u)*** et ***x(v)***, sont utilisées comme entrées pour une fonction ***f()*** qui calcule cette probabilité ou mesure de similarité.

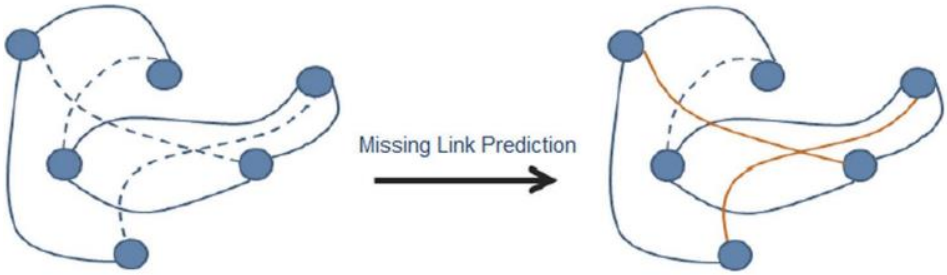

Figure *6*: structure de graphe avant et après la prédiction des liens manquant

❖ Prédiction des liens futurs (Link Forecasting):

La prédiction des liens futurs se concentre sur la prédiction des liens qui se formeront à l'avenir dans un réseau. Contrairement à la prédiction des liens manquants qui se base sur l'état actuel du réseau, la prédiction des liens futurs utilise des informations temporelles pour estimer les relations qui se développeront entre les nœuds à un moment ultérieur. Les modèles de prédiction des liens futurs intègrent souvent des composantes temporelles, telles que des tendances, des motifs saisonniers ou des dynamiques d'évolution du réseau, pour anticiper les liens qui se formeront dans le futur. La formule pour la prédiction des liens futurs (Link Forecasting) est donnée par ***P(u, v, t) = f(x(u), x(v), t)***. Cette formule est utilisée pour estimer la probabilité ou la mesure de similarité entre les nœuds u et v à un moment futur **t**, indiquant la formation potentielle d'un lien entre eux. Les caractéristiques des nœuds, ***x(u)*** et ***x(v)***, ainsi que le moment temporel t, sont pris en compte dans une fonction ***f()*** pour effectuer cette prédiction.

2. Classification des approches de prédiction de liens

Les approches de prédiction de liens peuvent être divisées en deux catégories : les approches basées sur l'apprentissage et les approches heuristiques. Tout d'abord, dans l'approche heuristique, la phase de prédiction a lieu immédiatement après la détermination des caractéristiques [35, 36, 35]. Cet ensemble d'algorithmes calcule les scores de similarité entre les paires de nœuds [36,37].

La seconde est basée sur un modèle d'apprentissage qui extrait le modèle des données d'entrée. Ces données d'entrée peuvent être un vecteur de caractéristiques prétraité qui donne un modèle de données pour prédire les liens.

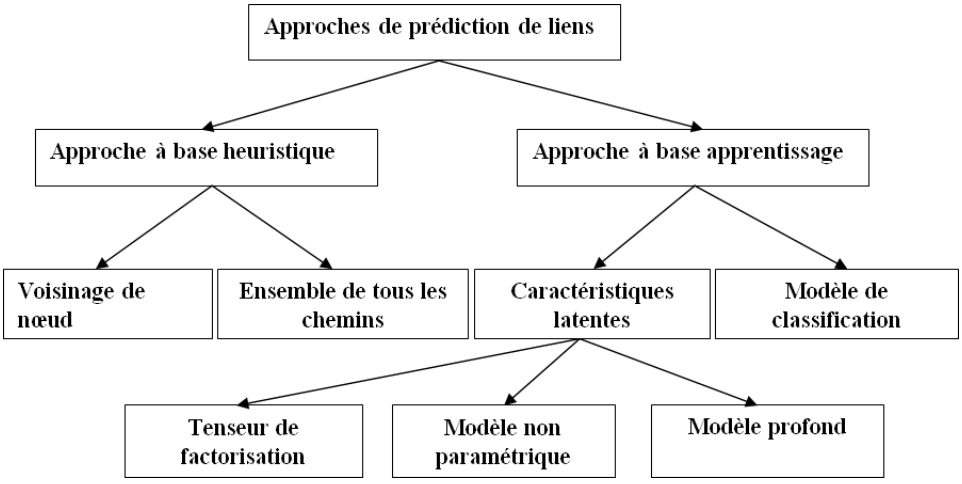

Figure *7* : Classification des approches de prédiction de lien

Cette décomposition est due au fait que les algorithmes d'apprentissage extraient eux-mêmes un modèle à partir de données afin de prédire les liens futurs par contre d'autres méthodes de prédiction basée sur l'heuristique, prédire les liens grâce à des similitudes connues de la structure du graphe **[56].**

Dans la suite d'étude, chaque approche est décrite en détail. Nous commençons par l'approche heuristique et ces sous-catégories, puis nous traitons l'approche d'apprentissage automatique.

2.1. Approches à base heuristique

Une heuristique est une méthode de calcul qui fournit rapidement une solution réalisable, pas nécessairement optimale ou exacte, pour un problème d`optimisation difficile. C'est un concept utilisé entre autres en optimisation combinatoire, en théories des graphes, en théories de la complexité des algorithmes et en intelligence artificielle. Cette approche inclut des méthodes qu`ils tentent de prédire les liens via des informations heuristiques, cette information capture les caractéristiques partagées ou les contextes de deux nœuds, en raison de ces informations capturées, l'approche heuristique est classifiée en deux classes qui sont : voisinage de nœud et ensemble de tous les chemins [35, 29, 37,15]. Les méthodes basées sur le voisinage prennent en compte les méthodes basées sur les indicateurs locaux et les chemins, sont appelées indicateurs globaux [29,40].

Ces techniques définissent une fonction S (x, y) qui attribue un score appelé similarité à chaque lien non observé pour chaque paire de nœuds x et y, et les K premiers liens avec le score le plus élevé sont prédits [56,77].la fonction de similarité peut varier d`un réseau à l`autre, même du même domaine [77]. En outre, il peut résoudre efficacement le problème de la prédiction des liens dans les réseaux hautement dynamiques et changeants tels que les réseaux sociaux en ligne [77]. La supériorité de ces algorithmes ne dépend pas de la connaissance du domaine requise pour calculer les scores de similarité. L'approche heuristique est aussi appelée approche par similarité [56].

2.1.1. Voisinage de nœud

Pour un nœud x, soit Γ(x) l'ensemble des voisins de x dans le réseau. De nombreuses approches sont basées sur l'idée que deux nœuds x et y sont susceptibles de former une connexion dans le futur s'il y a un grand chevauchement dans les ensembles adjacents Γ(x) et Γ(y). Cette approche est basée sur l'intuition naturelle que ces nœuds x et y représentent des auteurs communs à de nombreux collègues et ont donc tendance à se contacter [35]. Les nœuds "similaires" sont censés être des connexions plus prévisibles. Il est utilisé pour les études de prédiction de lien en raison de sa simplicité et du nombre réduit de paramètres. Quatre indices communs de voisinage de nœuds sont décrits ci-dessous : [56]

➢ Voisins communs (CN) :

La métrique CN (Common Neighbors) est l'une des métriques les plus utilisées pour les problèmes de prédiction de connexion, principalement en raison de sa simplicité [78]. Pour deux nœuds x et y, CN est défini comme le nombre de nœuds avec lesquels x et y interagissent directement [79]. Plus les voisins sont communs, plus il est facile d'établir une connexion entre x et y.

Cette échelle est définie par la formule : $\boldsymbol{Score\ CN\ (x,y) = |\Gamma(x) \cap \Gamma(y)|}$

où Γ(x) et Γ(y) représentent l'ensemble des nœuds adjacents x et y respectivement.

La métrique CN n'est pas normalisée, elle reflète donc généralement la similarité relative entre les paires de nœuds. L'utilisation de cette méthode pour calculer la similarité de toutes les paires possibles donne une méthode de prédiction de liaison locale avec une complexité temporelle **O(vk2(k+k))) = [O(Vk3)].** Par conséquent, certaines métriques basées sur le quartier examinent comment normaliser la métrique CN de manière significative [79].

➢ L'indice de Jaccard (JA) :

Cette métrique de similarité est principalement utilisée pour la recherche d'informations. Les coefficients de Jaccard sont des voisinages normalisés.

$$\mathbf{Score}_{\mathbf{JC}}(\mathbf{x}, \mathbf{y}) = \frac{|\boldsymbol{\Gamma}(\boldsymbol{x}) \cap \boldsymbol{\Gamma}(\boldsymbol{y})|}{\boldsymbol{\Gamma}(\boldsymbol{x}) \boldsymbol{U} \boldsymbol{\Gamma}(\boldsymbol{y})|}$$

En effet, il définit la probabilité que le voisin commun des nœuds x et y soit choisi si la sélection est faite aléatoirement à partir de l'union des ensembles voisins des nœuds x et y. Cependant, les résultats expérimentaux [35] montrent que la performance du coefficient de Jaccard est moins bonne que celle du nombre de voisins communs [56].

La complexité algorithmique de cette méthode est :**O(vk2(2k+2k)) = O(vk3)**

➢ L'indice Adamique-Adar (AA) :

Cette mesure de similarité a été initialement proposée par LadaAdamic et Eytan Adar [73] dans le but de mesurer la similarité entre deux entités en fonction de leurs caractéristiques communes [73]. Chaque poids d'une entité est pénalisé logarithmiquement par sa fréquence d'occurrence [77]. En prenant le voisinage comme caractéristique, on peut écrire :

$$\boldsymbol{Score\ AA}\ (\boldsymbol{x}, \boldsymbol{y}) = \sum_{z \in \boldsymbol{\Gamma}(\boldsymbol{x}) \cap \boldsymbol{\Gamma}(\boldsymbol{y})} \frac{1}{\log|\boldsymbol{\Gamma}(\boldsymbol{z})|}$$

Conceptuellement, Adamic/Adar [73] affine le simple nombre de voisins communs [58] en attribuant plus de poids aux voisins les moins connectés. Les résultats enregistrés pour la prédiction des liens existants montrent qu'Adamic/Adar [73] surpasse les deux métriques précédentes. Cette méthode est une autre variante de la méthode du voisin commun, et il est facile de voir qu'il y a une pénalité pour chaque voisin non partagé. La complexité temporelle de l'algorithme de cette méthode est : O(vk2(2k + 2k)) = O(vk3)[77].

➢ L'indice de pièces jointes préférentiel (PA) :

Cet indice est un résultat direct du modèle bien connu de formation de réseaux complexes [74] [75].

Les nœuds de réseau réels suivent les distributions de puissance à de nombreuses étapes, ce qui entraîne des échelles de réseau qui ne peuvent pas être prises en compte par les modèles de représentation de réseau précédents. Albert Laszlo Barabasiet Rek [74] a construit un modèle théorique basé sur l'observation que la probabilité de formation de liens entre deux nœuds augmente lorsque le degré de ces nœuds augmente [77].

Cette information « renforce » le concept lui-même pour produire la distribution des degrés de la loi de puissance observée dans les réseaux sans échelle [77]. La similarité entre deux nœuds peut être estimée selon le modèle de Barabasi-Albert [74] comme S (x, y) =|Γ(x) ||Γ(y)|

Cette mesure est également applicable dans des contextes non locaux, car elle ne repose pas sur des voisins communs. Cependant, lorsqu'elle est appliquée en tant que mesure globale, la précision des prévisions est généralement médiocre. La complexité de la méthode O(vk2) est plus rapide que celle de la méthode basée sur le voisinage commun [77].

2.1.2. Ensemble de tous les chemins :

De nombreuses méthodes affinent la notion de distance du plus court chemin en considérant implicitement l'ensemble de tous les chemins entre deux nœuds [35].

Les chemins entre deux nœuds sont une autre heuristique qui peut être utilisée pour calculer la similarité entre des paires de nœuds. Vous trouverez ci-dessous une brève introduction aux quatre principaux indices mondiaux [56].

- **KATZ :**

Katz est une méthode déclarée dans l'approche basée sur les chemins qui compte tous les chemins entre deux nœuds [80]. Les chemins sont exponentiels et décroissent avec leur longueur, de sorte que des chemins plus courts peuvent avoir plus de poids [79]. La formule est :

$$\sum_{l=1}^{\infty} Q^l . |path^l_{x,y}| = QA + Q^2A^2 + Q^3A^3 + \cdots$$

Où chemins $path^l$ est l'ensemble de tous les chemins de longueur **l** reliant **x** et **y**, et **β** un paramètre libre (c'est-à-dire le facteur d'amortissement) contrôlant les poids de chemin. La complexité de cette méthode est : **O (V_K + V_3 + v) [29].**

- **Temps de frappe**

Pour deux sommets, x et y dans un graphe, le temps de frappe, H_x, définit le nombre attendu d'étapes requises pour une marche aléatoire commençant à x pour atteindre y. Un temps de frappe plus court montre que les nœuds sont similaires, ce qui permet de créer des liens. Pour un graphe non dirigé, cela peut être considéré comme [56] :

$$score_{HTugraph}(x, y) = H_{x,y} + H_{y,x}$$

Il est facile de calculer la métrique du temps de frappe en effectuant quelques essais aléatoires. À la baisse, sa valeur peut avoir une variance élevée ; par conséquent, la prédiction par cette fonctionnalité peut être mauvaise. En raison de la nature sans échelle d'un réseau social, certains des sommets peuvent avoir une probabilité stationnaire très élevée (π) dans une marche aléatoire ; pour se protéger contre cela, le temps de frappe peut être normalisé en le multipliant par la probabilité stationnaire du nœud respectif, comme indiqué ci-dessous [56]

$$score_{NHTugraph}(x, y) = H_{x,y,\pi y} + H_{y,x,\pi x}$$

- **Page Rank**

Le score de similarité entre deux sommets x et y peut être mesuré comme la probabilité stationnaire de y dans une marche aléatoire qui revient à x avec une probabilité 1-β dans chaque étape, se déplacer vers un voisin aléatoire avec une probabilité β est un Page Rank pour la prédiction de lien [72].

$$Score\ RPR\ (x, y) = (1-\beta)\ (I - \beta N)\ -1$$

- **SIM RANK :**

SimRank est défini de manière cohérente en supposant que deux nœuds sont similaires s'ils sont connectés à des nœuds similaires [56].

$$score_{SR(x,y)} = \begin{cases} 1 & if\ x = y \\ \frac{\sum_{a \in (x)} \sum_{b \in (y)} simRank(a,b)}{|\Gamma(x)||\Gamma(y)|} & autres \end{cases}$$

où γ ∈ [0,1] est le vecteur de décomposition. SimRank peut également être interprété par une marche aléatoire. C'est $(x,)$qui mesure quand deux marcheurs aléatoires, chacun partant du nœud

x à y, se rencontrent à un nœud particulier [29]. Cependant, la complexité de ce processus d'expansion récursive est O(k2l). La complexité de chaque paire de nœuds est 0 (k2l+2) car il y a 2 sommes imbriquées à effectuer. Cette valeur doit être calculée pour chaque paire de nœuds afin que la complexité temporelle de l'algorithme final soit O(v2k2l+2) [77].

Nom	**La complexité**
Voisins communs (CN) **[50]**	O (VK^3)
L'indice de Jaccard (JA) **[7, 27]**	O (vk^3).
L'indice Adamique-Adar (AA) **[44]**	O (vk^3).
L'indice de pièces jointes préférentiel (PA) **[45]**	O (vk^2).
Katz KI **[51]**	O(v^3)
Sim Rank **[48]**	O (v^2k^{2l+2})

Tableau.1 Complexité et références pour les méthodes de prédiction de liens basées sur la similarité.

2.2. Approches à base apprentissage

L'apprentissage automatique ou apprentissage statistique est un domaine de recherche en intelligence artificielle qui utilise des approches statistiques pour donner aux ordinateurs la

capacité d'apprendre à partir de données. L'apprentissage automatique se compose généralement de deux phases. La première consiste à estimer le modèle à l'aide d'une quantité finie de données, appelées observations, disponibles lors de la phase de conception du système. L'estimation du modèle implique la résolution de problèmes pratiques tels que l'estimation des densités de probabilité et la prédiction des connexions manquantes ou futures dans les réseaux. C'est le cas pour le problème proposé. Cette phase est appelée

La « formation » a généralement lieu avant que le modèle ne soit réellement utilisé. La deuxième phase correspond au démarrage de la production. Une fois le modèle déterminé, de nouvelles données peuvent être soumises pour obtenir des résultats correspondant à la tâche souhaitée. À ce niveau d'abstraction, les caractéristiques sont apprises, extraites du modèle et finalement présentées au modèle menant à la prédiction de lien [54]. Conceptuellement, les modèles d'apprentissage basés sur les objectifs visent à extraire la structure des chiffres d'entrée et à prédire les connexions futures à l'aide du modèle appris [34]. Cette approche se divise en deux types : les modèles de classification et les modèles basés sur les caractéristiques latentes. L'idée clé derrière ce regroupement est la nature du modèle d'apprentissage. La classification des modèles extrait les modèles des données d'entrée et les apprend à prédire les liens manquants ou futurs. Les données d'entrée sont des vecteurs de caractéristiques prétraités, et chaque entrée est connue sous le nom d'indice de similarité, de nœud externe ou d'informations de lien [34, 78]. Cependant, pour les modèles basés sur les caractéristiques latentes, ces vecteurs de caractéristiques de prétraitement peuvent être facultatifs. Le modèle est construit de manière unique sur la base des caractéristiques latentes et le modèle est appris en extrayant les caractéristiques latentes du graphique d'entrée. D'autre part, ce modèle d'apprentissage peut utiliser des informations externes ainsi que des données de réseaux sociaux, et peut être utilisé pour prétraiter des vecteurs de caractéristiques. Les deux éléments suivants décrivent les modèles de classification et les modèles basés sur les caractéristiques latentes.

2.2.1. Modèle de classification

Toutes les méthodes de cette catégorie sont des apprentissages supervisés. Après avoir identifié un ensemble de caractéristiques essentielles pour l'apprentissage supervisé [34, 38], le problème de prédiction connexe correspond à la classification binaire [39] (Fig. 2.2). La prédiction des liens

est importante pour la classification binaire afin de prédire les deux classes, mais il est important de prédire les chaînes manquantes ou les futurs liens. Dans les modèles de classification prédictive de liens, les chercheurs utilisent des machines à vecteurs de support [40], Trees of décision [41], perceptron multicouche [34] et marche aléatoire supervisée [38]. Ils ont seulement constaté que les marches aléatoires fonctionnent bien pour prédire les futures connexions dans les modèles supervisés [42]. Les attributs de nœud et les caractéristiques topologiques sont déterminés principalement par des méthodes heuristiques. Cela signifie qu'au lieu de prédire directement les liens, ils peuvent être utilisés comme entrée de vecteur de caractéristiques pour former des modèles ou faire référence à des caractéristiques d'attribut de nœud telles que des informations de profil de réseau social. La figure 2.3 montre quelques propriétés des vecteurs de caractéristiques. Alors que la simple présence d'un vecteur de caractéristiques peut prédire la pertinence avec un modèle de classification binaire, la création d'un vecteur de caractéristiques intéressant nécessite une tâche supplémentaire. Voici quelques algorithmes pour les modèles supervisés.

- **Marche aléatoire supervisé (supervised random walk SRW):**

Une marche aléatoire est un modèle mathématique d'un système mécanique discret composé d'une série d'étapes aléatoires. Le concept de marche aléatoire est basé uniquement sur chaque instant. L'avenir du système dépend de l'état actuel du système, mais pas du passé. Même au plus près, il étudie souvent les marches aléatoires dans les réseaux ordinaires et les diagrammes plus complexes. Ceci est un exemple de la méthode Missing and Future Connections Prediction (SRW). Cette méthode est décomposée en unités de base appelées étapes, dont les longueurs elles-mêmes peuvent être constantes, aléatoires ou fixées par un réseau ou un graphe cyclique. Chaque étape a donc de nombreuses possibilités de randomiser la direction et la taille de la scène. Cette plage de possibilités peut être discrète (choisie parmi un nombre fini de valeurs) ou continue. Dans le problème de prédiction de lien, le défi consiste à combiner efficacement les informations de structure de réseau avec les données de nœud et d'attribut pour les liens qui restent largement ouverts. Les marches aléatoires supervisées combinent naturellement les informations sur la structure du réseau avec les attributs des nœuds et des liens. Ces attributs sont utilisés pour contrôler les marches aléatoires dans le graphique. Plus la tâche d'apprentissage est formulée, plus il est probable qu'une marche aléatoire visite le nœud où une nouvelle connexion est établie, car le

but est d'apprendre une fonction qui attribue des avantages aux connexions dans le réseau est plus élevé. [38].

➢ Machine vectorielle

Les machines à vecteurs de support (SVM) sont un ensemble de techniques d'apprentissage supervisé pour résoudre les problèmes de discrimination et de régression. SVM est une généralisation des classificateurs linéaires. Les SVM sont utilisées dans de nombreux domaines (bioinformatique, recherche d'information, vision par ordinateur, finance, prédiction de lien, etc.). Selon les données, les performances sont

Les machines à vecteurs de support sont des ordres de grandeur plus grands que les réseaux de neurones et les modèles de mélange gaussiens.

➢ Perceptron multicouche

Un perceptron multicouche (MLP) est un type de réseau neuronal formel composé de plusieurs couches, où les informations ne circulent que de la couche d'entrée à la couche de sortie. Il s'agit donc d'un réseau à propagation directe (feedforward). Chaque couche est constituée d'un nombre variable de neurones, et les neurones de la dernière couche (appelés "sorties") sont les sorties de l'ensemble du système. Dans la première version, le perceptron était une seule couche et n'avait qu'une seule sortie à laquelle toutes les entrées étaient connectées.

➢ Arbre de décision :

Un arbre de décision est un outil d'aide à la décision qui représente un ensemble de choix sous la forme graphique d'un arbre. Différents choix sont possibles au bout de la branche (feuille sur l'arbre), qui se font en fonction des choix effectués à chaque étape. Le principal avantage des arbres de décision est qu'ils peuvent être calculés automatiquement à partir de bases de données à l'aide d'algorithmes d'apprentissage supervisé. Ces algorithmes sélectionnent automatiquement des variables discriminantes à partir de quantités potentiellement importantes de données non structurées.

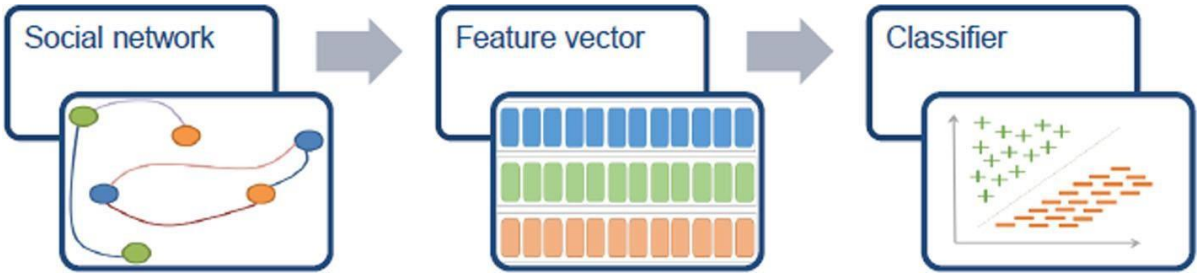

Figure *8* : modèle de classification

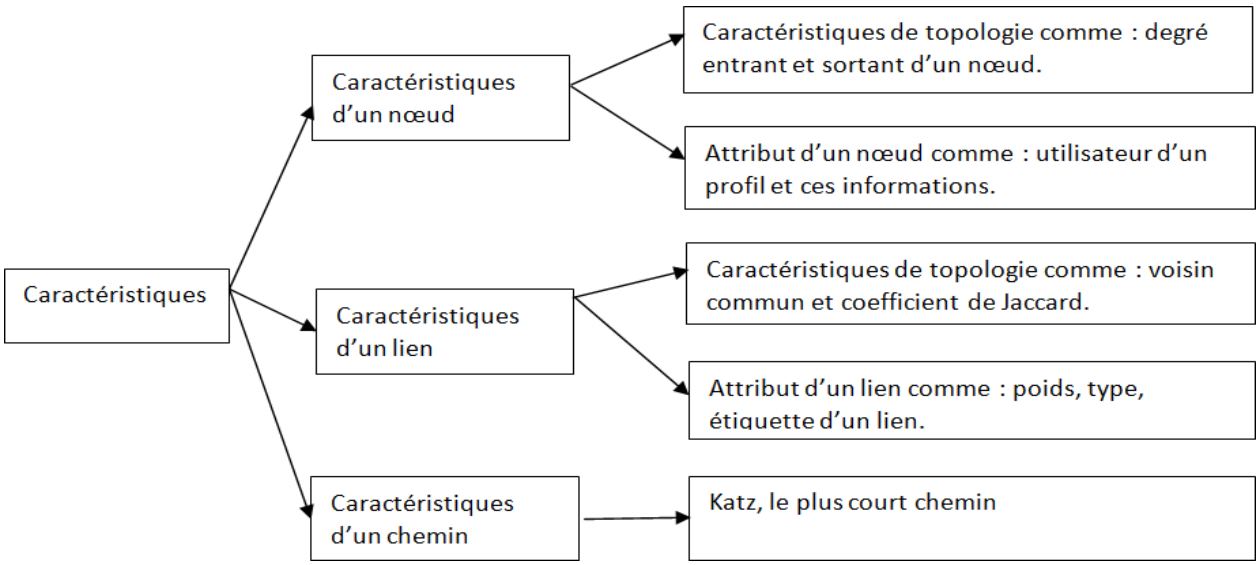

Figure *9* : regroupement des caractéristiques les plus utilisé

2.2.2. Modèle basé sur les caractéristiques latentes

L'une des prémisses des modèles basés sur les caractéristiques latentes (caractéristiques latentes, c'est-à-dire non directement extractibles) est de construire un modèle capable de découvrir les caractéristiques latentes de la structure du diagramme [43, 31, 44, 45]. Comme mentionné précédemment, il existe deux types d'informations qui peuvent être obtenues à partir du réseau

(Fig. 2.3). Néanmoins, les chercheurs dans le domaine ont constaté que la structure du diagramme et la combinaison de ces deux types d'informations ne sont potentiellement pas évidentes avec des techniques simples, en particulier lorsque des caractéristiques de réseau dynamiques ou hétérogènes sont capturées dans les données et sont censées présenter des propriétés similaires. [31, 46, 47]. L'objectif des approches basées sur les caractéristiques latentes est d'apprendre des modèles de liens observés qui peuvent prédire la valeur des entrées non observées. La représentation latente de chaque nœud correspond à un point sur la surface de l'hypersphère unitaire. Dans la fonction latente du modèle, chaque caractéristique est associée à un vecteur $\boldsymbol{e_i} \in \mathbf{R}\ \mathbf{H}^{\mathbf{e}}$, où **He Ne** (**Ne** est le nombre d'entités).

Chaque lien est décrit par une propriété potentielle de l'entité. Par exemple, chaque nœud peut être modélisé par un vecteur, comme le montre la figure 2.4. La composante ei1 correspond aux capacités potentielles d'un développeur talentueux, et ei2 correspond à la santé. Alors les fourmis l'ont

Nous pouvons conclure qu'il est en meilleure santé s'il a un lien avec ses coéquipiers. Alternativement, nous pouvons conclure que Sam a des fonctionnalités de développeur compétentes potentielles plus élevées en raison de ses relations avec les collègues de Reza et Borna. Notez que contrairement à cet exemple, les caractéristiques latentes des modèles dérivés suivants sont généralement difficiles à interpréter. Une intuition centrale derrière les modèles de caractéristiques latentes relationnelles est que les relations entre les entités peuvent être déduites des interactions de leurs caractéristiques latentes [43, 48, 49, 45]. Cependant, il existe de nombreuses façons de modéliser ces interactions et d'en déduire l'existence de relations. L'utilisation de contraintes de dimensionnalité rend la prédiction de connexion efficace en termes de temps de calcul et de coût mémoire [43, 50, 47]. De plus, la variation de la dimensionnalité de l'espace latent offre la possibilité de définir précisément le compromis entre la complexité de calcul et la qualité de la solution. Une dimensionnalité plus élevée se traduit par une représentation plus précise de l'espace latent pour chaque nœud, mais à un coût de calcul plus élevé [47]. Plusieurs approches ont été présentées pour obtenir ces propriétés potentielles. Chaque approche est détaillée dans les trois points suivants.

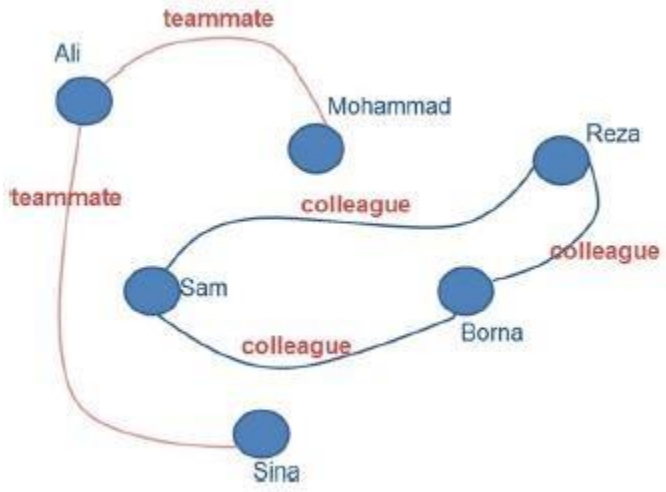

Figure *10* : Exemples d'un réseau avec des caractéristiques latentes

➢ Tenseur de factorisation

Il est important de noter que les tenseurs factorisés sont une approche de données structurées bien connue dans une variété de contextes d'apprentissage. Le succès des tenseurs factorisés dans les problèmes de prédiction de connexion découle de leur capacité supérieure à modéliser et à analyser des données relationnelles [51, 52, 53]. Les méthodes basées sur les tenseurs sont généralement disponibles pour deux matrices [54] et trois ordres. Pour prédire les liens futurs, le troisième domaine est pris comme un autre instantané horaire. Cette approche convient à la détection de caractéristiques latentes à long terme. Plus formellement, utilisez des expressions telles que :

$$Z(i,j,t) = \begin{cases} 1 \text{ si le noeud } i \text{ a un lien avec le noeud } j \\ 0 \text{ autrement} \end{cases}$$

Ce qui montre que le lien i à j est apparu au moment t [56, 57, 51, 58, 47, 59]. D'autre part, dans des réseaux hétérogènes avec des données multi-relationnelles, la troisième dimension montre différents types de liens. Il est le plus applicable dans des réseaux hétérogènes où les liens

ont une forte dépendance [53, 52, 59, 61, 62]. Le troisième tenseur d'ordre est utilisé pour définir des données multi-relationnelles comme suit :

$$Z(i,j,t) = \begin{cases} 1 \text{ si } relation_k(noeud\ i, noeud\ j) = vrai \\ 0 \text{ autrement} \end{cases}$$

➢ Modèle non paramétrique

L'utilisation du modèle non paramétrique est un type de modèle de fonction latente. Dans ce modèle, les méthodes utilisent principalement des méthodes non paramétriques bayésienne pour découvrir les caractéristiques latentes discriminants et en déduisent automatiquement la dimension sociale inconnue **[69, 71, 47]**. Fondamentalement, chaque entité est décrite par un ensemble de fonctions binaires.

Les modèles non paramétriques permettent infère simultanée du nombre de fonctions latentes en même temps **[44]**. D'autre part, certains modèles sont non paramétriques basées sur un noyau. Ces modèles à base de noyau comprennent la régression du noyau, de calcul des similitudes entre la requête et tous les membres de l'ensemble de la formation **[44, 71]**. **[71**] a introduit un modèle non paramétrique de base. Ils ont montré que chaque entité est décrite par un ensemble de caractéristiques binaires et il n'y a pas de priorité pour chacun d'eux. La probabilité d'avoir un lien à partir d'une entité à une autre est entièrement déterminé par un effet combiné de toutes les interactions de caractéristiques appariées. S'il y a **K**caractéristiques, puis **Z** sera le **N × K** matrice binaire où chaque rangée correspond à une entité et chaque colonne correspond à une caractéristique de telle sorte que $Z_{ik} \equiv$ **Z (i, k) = 1** si la i ^{me} entité a une fonction **k** et $Z_{ik} =$ **0** autrement. Le modèle comporte une matrice de poids réel de la valeur **(K × K).**Où $W_{kk} \equiv$ **W (k, k ')** est le poids qui influe sur la probabilité d'occurrence d'un lien à partir d'une entité **i** à **j** si l'entité **i** a une fonction **k** et l'entité **j** a une fonction **k** .On suppose que les liens sont indépendamment conditionnés sur **Z** et **W,** et que seules les caractéristiques des entités **i et j** influencent laprobabilité d'un lien entre ces entités. Cette notion définit la probabilité :

Pr ***(Y | Z, W)*** $= \prod_{i,}$ **Pr** (| ***Z***,)

Où le produit se situe au-dessus de toutes les paires d'entités. Compte tenu de la matrice de fonction **Z** et la matrice de poids **W,** la probabilité qu'il y ait un lien à partir de l'entité je à l'entité j est donnée comme :Pr (y_{ij} = 1 | Z, W) = (Z_i W Z_J) = σ ($\sum_{k,}$)

Où **σ (.)** est une fonction sigmoïde, qui transforme les valeurs de **(-∞, + ∞)** à **(0, 1)**. Les modèles non paramétriques ont unegrande capacité à explorer les modèles d'évolution particulièrement de fluctuation

Saisonnières. Il est à noter que, cette efficacité est juste par rapport aux méthodes basées heuristiques et non à l'autre modèle basé sur la fonction latente **.**Parmi les modèles basés sur l'apprentissage, le modèle le plus rapide est non paramétrique. La raison étant qu'il n'a pas ou quelques paramètres. D'autre part, la plupart des méthodes utilisent la mise en œuvre LSH pour les rendre plus rapides. LSH ou localité hachage sensible est souvent utilisé dans la base de données pour les recherches de la table ou la récupération des éléments qui correspondent à la base de données [**44]**. Ce modèle est également connu sous le nom modèle de probabilité **[56].**

- **Modèle profond**

Les réalisations importantes des approches d'apprentissage en profondeur dans les domaines de la vision par ordinateur, de la reconnaissance de la parole et du traitement du langage naturel [82, 84, 85] ont motivé les chercheurs à utiliser des modèles profonds dans les tâches de prédiction de liens [83, 86, 88, 87, 12]. , 81]. En apprentissage général, un modèle de profondeur est, au mieux, un ensemble d'algorithmes d'apprentissage automatique qui effectuent des tâches d'apprentissage à plusieurs niveaux correspondant à différents niveaux d'abstraction. Les réseaux de neurones artificiels sont couramment utilisés. Les niveaux de ces modèles statistiques appris correspondent à différents concepts de niveau, un concept de niveau supérieur est défini à partir d'un concept de niveau inférieur, et le même concept de niveau inférieur définit plusieurs concepts de niveau supérieur [82, 85].

3. Tableau comparatif des approches :

Nous avons vu quelques méthodes de prédiction de liens d'apprentissage automatique et heuristique. **Le Tableau 2.2** récapitule les avantages et les défis des approches présentées précédemment.

Approche	Avantages	Défis
Voisinage de nœud	- Idée de base simple et intuitive. - Ne nécessite pas de connaissances spécifiques du domaine.	- Ne pas détecter les caractéristiques d'évolution future. - Difficulté à reconnaître les modèles d'évolution.
Ensemble de tous les chemins	- Prise en compte des relations indirectes entre les nœuds. - Capacité à capturer les motifs de connectivité complexes.	- Complexité de calcul lors de la génération de tous les chemins possibles. - Sensibilité au bruit ou aux erreurs dans les données des chemins.
Modèle de classification	- Capacité à modéliser des relations non linéaires entre les nœuds. - Interprétabilité des résultats en termes de classes prédites.	- Besoin de données d'entraînement étiquetées pour chaque lien. - Difficulté à modéliser des relations complexes ou à longue portée.
Tenseur de factorisation	- Prise en compte des interactions entre les nœuds et les attributs.	- Besoin de données structurées sous forme de tenseurs.

	- Capacité à gérer des ensembles de données sparses.	- Complexité de calcul lors de la factorisation du tenseur.
Modèle non paramétrique	- Flexibilité dans la modélisation des relations complexes. - Capacité à capturer des motifs non linéaires et non paramétriques.	- Besoin de grandes quantités de données pour estimer les paramètres. - Difficulté d'interprétation des résultats en termes de paramètres spécifiques.
Modèle profond	- Capacité à apprendre des représentations hiérarchiques des nœuds. - Performance élevée pour la prédiction de liens complexes.	- Besoin de grandes quantités de données et de puissance de calcul. - Sensibilité au surapprentissage si le modèle est trop complexe.

Tableau 2 Avantages et défis des modèles de prédiction de lien.

Chapitre 3

Méthode Développée

Introduction :

Dans ce chapitre on va montrer comment on a amélioré les résultats et la vitesse de la méthode de prédictions de lien créer par Shikhar Sharma et Anurag Singh dans leur recherche nommé : **An efficient méthode for Link prédiction in weighted multiplex networks** par modifier la méthode de travail des fonctions critiques dans le code source par des fonctions récursives qui servent a simultanément améliorer la performance et l'exactitude des résultats .

1. plan du travail :

De nombreux systèmes et leurs interactions peuvent être modélisés comme des abstractions qui représentent assez précisément leur dynamique. Vous pouvez simuler ces dynamiques pour découvrir de nouvelles propriétés, interactions et différentes expressions. Il peut également être utilisé pour prédire et prédire le comportement. Ces entités peuvent être modélisées comme des nœuds sur un diagramme avec des connexions/arêtes représentant des interactions/relations entre individus. Par exemple, il peut représenter un simple réseau d'amitié. Les nœuds représentent les personnes faisant l'objet de l'enquête et les arêtes indiquent si deux personnes sont amies. Alternativement, vous pouvez ajouter un poids normalisé à chaque lien pour indiquer le degré d'amitié entre les deux personnes, déduit d'autres facteurs. La prédiction de connexion ou la prédiction de formation de connexion dans des systèmes abstraits tels que des réseaux complexes est utile de plusieurs manières. La prédiction de lien tente de formuler la probabilité qu'il existe un lien entre deux nœuds. De nombreux réseaux nécessitent une expérimentation intensive en temps et en ressources pour déterminer si une connexion existe ou peut exister à l'avenir. La prédiction de liens est une bonne alternative car elle peut se concentrer sur les liens probables issus de modèles sophistiqués. À mesure que les réseaux se développent et que la demande de mécanismes d'analyse de réseau efficaces augmente, la prédiction de liaison devient nécessaire pour résoudre le problème des liaisons manquantes. De plus, la prédiction de liens constitue la base de divers systèmes de recommandations utilisées dans le marketing en ligne, les services de commerce électronique et de nombreux réseaux sociaux. La prévision des connexions au réseau de communication terroriste peut aider à prévoir et à intercepter des informations critiques sur la sécurité nationale. De plus, la prédiction de connexion est effectuée dans diverses situations telles que les stratégies de réseau de transport efficaces et la recherche sur les maladies génétiques. Ces réseaux peuvent être étudiés et ont différentes facettes. Ils peuvent être représentés comme des

réseaux temporels, mais les informations peuvent également être visualisées comme des réseaux multicouches.

Diverses stratégies ont été utilisées pour la prédiction de lien. Certaines stratégies sont basées sur des informations topologiques locales du réseau correspondant à l'indice de notation. L'algorithme Common Neighbor (CN) est l'une de ces stratégies où plus deux nœuds non connectés ont toujours un voisin commun, plus ils sont susceptibles de se rejoindre à l'avenir. L'indice de Salton normalise l'indice de voisinage commun en tenant compte du degré de chaque nœud. Une nouvelle stratégie a été proposée pour prédire le chaînon manquant

En plus des voisins communs, considère également le nombre de connexions entre les deux ensembles de voisins anormaux d'un nœud donné. D'autres indices basés sur la similarité locale (indice de Jaccard, indice de préférence d'attachement, indice de déclin de la concentration, etc.)

L'indice de promotion du hub a également été utilisé pour la prédiction des liens. Les indices qui prennent en compte les informations globales du graphe (distance parcourue ou temps de trajet moyen), tels que l'exposant de Katz et les méthodes basées sur le cosinus, ont également été utilisés avec succès pour la prédiction des liens. Des stratégies utilisant des attributs de nœud ont été proposées avec de bons résultats.

De plus, Bliss et al. He et al ont utilisé avec succès un algorithme évolutif pour combiner différents indices afin d'obtenir des résultats optimisés. J'ai utilisé l'opérateur OWA pour la combinaison. Voir la section suivante pour une brève description de certaines de ces méthodes. Cependant, la plupart des réseaux du monde réel sont mieux considérés comme plusieurs types de relations, plutôt qu'un seul. Par exemple, les prédictions de liens entre deux entités dans un réseau social sont probablement influencées par une combinaison de facteurs tels que des intérêts communs, une présence physique à un moment donné et des connaissances partagées. La prédiction des cibles est basée sur un paradigme d'information complexe, et non sur un seul paramètre. Par exemple, le fait que deux personnes soient amies dépend non seulement du fait qu'elles ont les mêmes intérêts, mais aussi d'autres aspects tels que le temps passé ensemble, les connaissances communes et divers autres facteurs. . Il n'est pas pratique d'établir et de déterminer des relations basées sur un seul paramètre. Un réseau multiplexé est un réseau dans lequel chaque

paire de nœuds est connectée par un type de bord différent et peut être considérée comme plusieurs couches. Ces réseaux contiennent différents types d'informations, mais elles sont distribuées par nature. Un réseau multiplexé stocke chaque type d'informations dans une couche distincte, encapsulant ainsi de nombreuses relations différentes entre les mêmes entités. Les algorithmes actuels sont très spécifiques au réseau et au contexte. Celles-ci ont tendance à s'appuyer sur des hypothèses de réseau sous-jacentes plutôt que de dériver des hypothèses du réseau lui-même. Leurs performances dépendent également d'un grand nombre d'arêtes. Par conséquent, dans les réseaux multiplexés, ces algorithmes donnent des résultats mitigés et ne peuvent pas utiliser efficacement les informations provenant de différentes couches. Dans ce travail, nous proposons une approche qui peut être utilisée pour prédire la connectivité dans les réseaux multiplexés lorsque l'extension des algorithmes existants n'est pas possible. Peu de recherches ont été effectuées à cet égard, les recherches récentes se concentrant sur des types spécifiques de réseaux sociaux plutôt que sur plusieurs réseaux en général.

L'importance des poids dans les réseaux est bien connue. Dans ce travail, pour détecter la force et la similarité des bords, nous proposons un algorithme de prédiction de poids qui repose sur l'évaluation par l'algorithme stochastique proposé. Ces scores sont utilisés de manière normalisée pour prédire les poids dans le réseau. La normalisation tient compte des écarts significatifs entre les scores des arêtes considérées. Cette méthodologie est comparée à d'autres indices en utilisant l'erreur quadratique moyenne normalisée comme indicateur de performance. Il s'agit de la racine carrée normalisée de la somme des différences au carré dans les prédictions de poids pour un cas de test particulier. Ce travail utilise plusieurs sources d'information et les combine pour créer une stratégie de prévision optimale. Il est indépendant du contexte et du réseau. Cela fonctionne pour tout réseau avec une corrélation positive dans les couches. Les résultats montrent une grande précision avec une erreur minimale. Il s'agit d'une approche évolutive qui combine les idées de prédiction de lien et de poids et peut être utilisée en parallèle pour les réseaux avec des millions de nœuds. Ce document est organisé comme suit : La deuxième section décrit diverses stratégies de prédiction de lien. La troisième section présente un modèle de réseaux multiples. La quatrième section propose une méthodologie pour utiliser l'algorithme de prédiction de lien. La cinquième section présente les résultats et l'analyse de l'algorithme proposé. La sixième section examine les

poids des liens et propose comment prédire les poids des liens prévus à l'aide des résultats et de l'analyse. Et la section 7 présente un résumé des résultats analytiques de cette étude. (88)

2. Méthode concerné :

La méthode de Shikhar Sharma et Anurag Singh utilise les informations de toutes les autres couches du réseau dans le but de prédire les connexions à une couche particulière, appelée couche de destination. L'autre couche s'appelle la couche de prédiction. Chaque couche à un impact différent sur les prédictions de la couche cible. Certaines couches peuvent bien représenter la couche cible, tandis que d'autres peuvent ne pas bien la représenter. Ces informations sont nécessaires pour une bonne prédiction globale des liens. Les informations sur la relation et la représentation entre la couche cible et chaque couche de prédiction sont extraites de la structure et des connexions du réseau. Ces informations sont testées pour obtenir une mesure de la qualité de représentation d'une tranche prédite donnée par rapport à la tranche cible. La valeur de prédiction de lien finale est le résultat pondéré des résultats précédents.

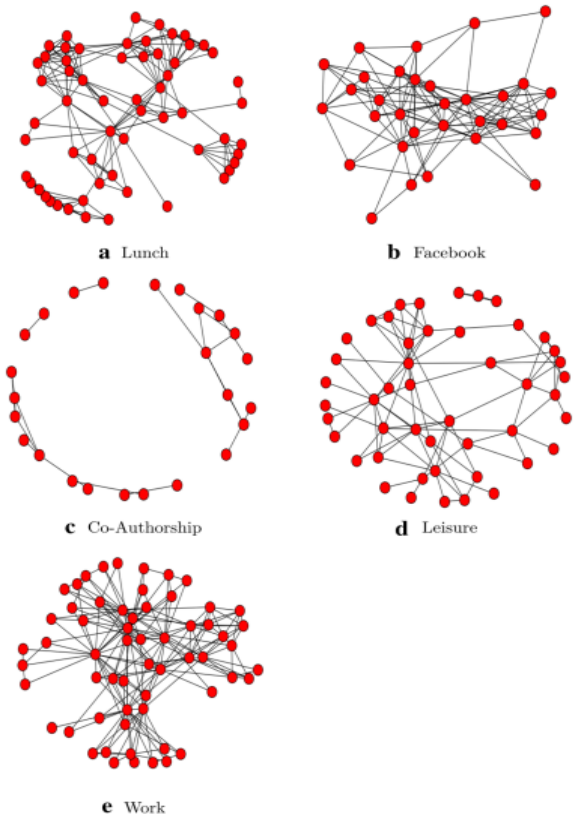

Figure 11 Relations hors ligne (Facebook, loisirs, travail, co-publication, déjeuner) entre les employés du département d'informatique à Aarhus.(88)

Plus précisément, le score final est déterminé en pondérant et en combinant les évaluations de chaque strate. Les poids sont obtenus en examinant la correspondance des liens entre les deux couches en fonction de la probabilité que le lien soit présent dans la couche cible s'il est présent dans la couche de prédiction. Cela détermine l'importance relative d'un niveau de prédicteur pour un niveau cible donné. Un schéma de l'algorithme proposé est illustré à la figure 3. Chaque couche de prédiction fournit des informations sur la probabilité qu'une connexion existe dans la couche cible. Additionnez les probabilités et attribuez un score à chaque paire de nœuds. Ces résultats sont triés et testés pour leur précision. De même, les liens non observés et observés sont sélectionnés au hasard et comparés en fonction de leurs scores de probabilité attribués pour obtenir l'aire sous la courbe (AUC).

Le premier algorithme (Algorithme 1) est utilisé pour attribuer la probabilité qu'une connexion existe dans une couche sur la base des informations obtenues à partir d'autres couches. Les probabilités sont calculées séparément pour chaque couche et utilisées comme pondérations. La probabilité globale est une combinaison de probabilités déterminées individuellement [22]. Les probabilités sont la dépendance probabiliste de l'existence de connexions dans la couche cible en observant un instantané précédent du graphique (représentant un sous-ensemble du graphique après suppression des bords et utilisé à des fins de test de performance) et en considérant la couche prédictive calculée en estimant le sexe. Le même processus est effectué pour chaque paire. Un deuxième algorithme (algorithme 2) est utilisé pour obtenir les mesures AUC pour la méthode proposée. Sélectionnez itérativement deux arêtes, une de l'ensemble d'apprentissage et une qui n'était pas présente (non observée) dans le graphique. Comparez les probabilités et augmentez le résultat si la probabilité d'un bord correctement prédit est supérieure à la probabilité d'un bord non observé.

L'algorithme de détermination de la précision (algorithme 3) trie toutes les arêtes en fonction de leur score attribué et vérifie le nombre d'arêtes pertinentes (correctement prédites) qui y contribuent.

Algorithm 1 Proposed Likelihood Assignment Algorithm

```
1: procedure ASSIGN–LIKELIHOOD
2:     for each layer i ∈ L^P do
3:         Score(j) <- 0
4:         Weight(layer) ← Likelihood(Link in L^T| Link in i)
5:     end for
6:     for each edge j ∈ (U − E) ∪ Ω do
7:         for each layer i ∈ L^P do
8:             Score(j) ← Score(j) + Weight(i) ∗ linkInPredictorLayer(j, i)
9:         end for
10:    end for
11: end procedure
```

Figure 12 algorithme de likelihood utilisé (88)

Cet algorithme prend un réseau en entrée, le renvoie et attribue des points à chaque arête en fonction de sa probabilité de formation.

La procédure **linkInPredictorLayer** est utilisée pour obtenir des informations sur la présence de liens dans la couche de prédiction pour un instantané de réseau donné. Évaluez la couche de prédiction pour la présence d'arêtes et utilisez l'estimation de probabilité pour prédire la présence d'arêtes dans la couche cible. Renvoie vrai (valeur 1) si le niveau de prédicteur a un bord, sinon renvoie la valeur 0. Ceci est répété pour chaque paire de niveaux et les probabilités sont combinées pour attribuer un score final. Nous connaissons la structure du réseau dans l'instantané précédent. La détermination de l'opinion combinée des couches de probabilité et de prédiction nous permet de créer une idée de l'existence de connexions futures.

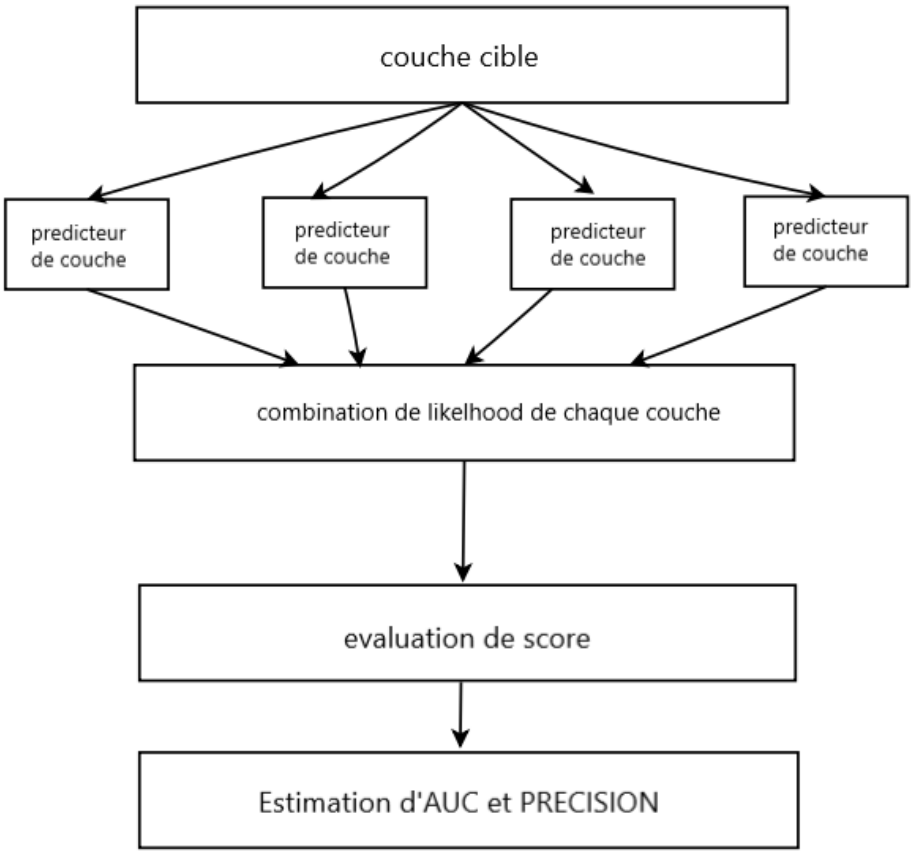

Figure 13 prédictions des liens dans les réseaux multicouches (88)

Algorithm 2 AUC Measure

```
procedure GET–AUC
    Set AUC, α, β, γ ← 0
    nTests ← Set number of tests to perform
    do
    i ← randEdge(U-E)
    j ← randEdge(Ω)
    if edgeScore(j) > edgeScore(i)
        increment β
    else if edgeScore(j) == edgeScore(i)
        increment γ
    end if
    increment α
    decrement nTests
    While nTests ≠ 0
    AUC ← (β+γ)/α
end procedure
```

Figure 14 algorithme 2 mesure de AUC(88)

Algorithm 3 Precision

```
1: procedure GET–PRECISION
2:     sortedScores ← Sort(Scores,Descending)
3:     highLikelihood ← sortedScores(1:delta)
4:     for each score s ∈ highLikelihood do
5:         if edge(score) ∈ Ω
6:             increment ϵ end if
7:     end for
8:     Precision ← ϵ/δ
9: end procedure
```

Figure 15 algorithme3 mesure de precision (88)

3. Méthode proposé :

Notre méthode consiste à améliorer une méthode existante créer par **Shikhar Sharma et Anurag Singh,** en remplaçant les fonctions critiques qui utilisent des boucles (loop) par des fonctions récursives. Cette approche peut offrir plusieurs avantages potentiels.

3.1. Définition de la fonction récursive :

En informatique et en mathématiques, le terme fonction récursive désigne deux concepts liés, mais distincts. Plus généralement les termes " récursif ", " récursivement ", " récursion ", " récursivité ", peuvent faire référence suivant le contexte à l'une ou l'autre des deux notions.

Pour déclarer une fonction récursive, on doit :

- assurer qu'il y a une variable de contrôle.
- commencer par traiter les cas de base. (89)

Les avantages des fonctions récursives :

- Clarté du code : Les fonctions récursives permettent d'exprimer des problèmes de manière plus concise et intuitive. Elles peuvent fournir des solutions récursives naturelles à des problèmes qui se prêtent à une approche itérative.
- Élégance de la solution : Les fonctions récursives peuvent souvent conduire à des solutions plus élégantes et plus simples. Elles permettent d'exprimer des algorithmes de manière plus abstraite, en se concentrant sur la logique du problème plutôt que sur les détails de l'itération.
- Résolution de problèmes complexes : Les fonctions récursives sont particulièrement utiles pour résoudre des problèmes complexes qui peuvent être décomposés en sous-problèmes plus simples. Elles permettent de diviser un problème en plusieurs sous-problèmes de même nature, facilitant ainsi la compréhension et la résolution.

- Utilisation de la pile d'appels : Les fonctions récursives utilisent la pile d'appels pour stocker les informations relatives à chaque appel récursif. Cela permet de conserver les états intermédiaires des appels récursifs et de les traiter dans l'ordre approprié, simplifiant ainsi la logique de l'algorithme.
- Flexibilité : Les fonctions récursives sont souvent plus flexibles que les approches itératives. Elles permettent de résoudre des problèmes de manière générique, en s'adaptant aux variations et aux dimensions changeantes des données. (90)

4. Méthode utilisé :

4.1. Algorithme AUC :

L'algorithme "calculate_auc" calcule l'aire sous la courbe ROC (Receiver Operating Characteristic) pour un problème de classification binaire. Il effectue un certain nombre d'itérations défini par le paramètre "n". À chaque itération, il sélectionne aléatoirement un score pour une arête manquante et un score pour une arête non existante. En comparant ces scores, il met à jour les compteurs "beta" et "gamma" pour compter les occurrences où le score de l'arête manquante est supérieur ou égal au score de l'arête non existante. Une fois toutes les itérations terminées, l'algorithme calcule l'aire sous la courbe ROC normalisée en utilisant les compteurs "beta" et "gamma" et retourne ce résultat. Cela permet d'évaluer la performance d'un modèle de classification binaire en termes de taux de vrais positifs et de faux positifs.

ALGORITHME calculate_auc

```
1   ALGORITHME calculate_auc (n, missing_edges_scores, non_exist_scores, i = 0, beta = 0,
    gamma = 0)
2    SI i est égal à n
3          RETOURNER (beta + 0.5 * gamma) / n
4    FSI
5    missing_len <- longueur (missing_edges_scores) – 1
6    non_exist_len <- longueur (non_exist_scores) – 1
7    missing_index <- randint(0, missing_len)
8    non_exist_index <- randint (0, non_exist_len)
9    missing_score <- missing_edges_scores[missing_index].score
10    non_exist_score <- non_exist_scores[non_exist_index].score
11       SI missing_score > non_exist_score
12                beta <- beta + 1
13        SINON SI missing_score est égal à non_exist_score
14                     gamma <- gamma + 1
15           FSI
16    RETOURNER calculate_auc (n, missing_edges_scores, non_exist_scores, i + 1, beta, gamma)
17  FIN ALGORITHME
```

4.2. Algorithme Méthode de Probabilité :

L'algorithme "likelihood" est conçu pour calculer la probabilité de correspondance entre une cible (target) et un prédicteur (predictor), en utilisant la fonction auxiliaire "count_edges". Dans cet algorithme, la fonction "count_edges" est appelée pour déterminer le nombre d'arêtes correspondantes entre la cible et le prédicteur. Ce compte est ensuite divisé par la longueur totale des arêtes du prédicteur pour obtenir la probabilité de correspondance. L'algorithme "count_edges" fonctionne en prenant en compte une cible et un prédicteur, ainsi que des paramètres optionnels pour les arêtes et un compteur. Il commence par créer une liste des arêtes de la cible s'il n'y en a pas déjà une. Ensuite, il vérifie si la liste des arêtes est vide, auquel cas il renvoie le compteur actuel. Sinon, il extrait le premier élément de la liste des arêtes et vérifie si cet élément ou son symétrique (l'échange des nœuds) est présent dans les arêtes du prédicteur. Si une correspondance est trouvée, le compteur est augmenté. L'algorithme répète ensuite ce processus récursivement en passant la liste des arêtes modifiée et le compteur mis à jour jusqu'à ce que toutes les arêtes de la cible aient été vérifiées. Finalement, l'algorithme "likelihood" retourne la probabilité calculée. En utilisant ces deux algorithmes conjointement, il est possible d'évaluer la probabilité de correspondance entre une cible et un prédicteur à partir de leurs arêtes respectives.

```
ALGORITHME calculate_auc
1    ALGORITHME likelihood (target, predictor)
2       count <- count  edges(target, predictor)
3           prob <- count / len(predictor.edges())
4        RETOURNER prob
5    ALGORITHME count  edges(target, predictor, edges = None, count = 0)
6       SI edges est égal à None
7                edges <- list (target.edges())
8    FSI
9       SI edges est vide
10
11   FSI
12                        edge <- edges.pop(0)
13          SINON SI missing_score est égal à non_exist_score
14                                count <- count + 1
15              FSI
16       RETOURNER count_edges(target, predictor, edges, count)
17   FIN ALGORITHME
```

Chapitre 4

Expérimentation

Introduction :

Dans ce dernier chapitre, nous présentons l'implémentation d'une technique de prédiction d'interactions dans les réseaux complexe multicouches et une methode devloppée a la base du premiere methode. La première est basée sur les calcules avec l'utilisation des boucles et la deuxieme est basée sur l'utlisation des fonctions recursives et nous allons faire une comparaison entre les deux methodes dans les resultats et le temps d'execution

. Nos expérimentations sur la meme dataset que l'ancienne recherche a été faite ainsi que les tests et les résultats obtenus serons exposés à la fin du chapitre.

1. Technologies utilisées :

Notre choix s'est porté sur le langage de programmation Python pour réaliser les diverses implémentations. Python s'est révélé être une option idéale, facilitant grandement nos expérimentations.

1.1. Python :

Python est en effet un langage de programmation orienté objet, interprété et de haut niveau. Il offre une sémantique dynamique et robuste, ce qui le rend compatible avec différentes plateformes telles que Unix, MacOs et Windows. Python est reconnu pour sa puissance, permettant de créer des représentations simples et flexibles pour les graphes. De plus, il propose des expressions claires et concises pour le Graph Mining, facilitant ainsi le traitement et l'analyse des données graphiques.

1.2. Bibliothèques utilisées :

1.2.1. NetwokX :

NetworkX est une bibliothèque Python dédiée à l'analyse de graphes complexes. Pour comprendre les fonctionnalités de NetworkX, il est nécessaire de comprendre d'abord les graphes. Les graphes sont des structures mathématiques utilisées pour modéliser de nombreux types de relations et processus dans les systèmes physiques, biologiques, sociaux et d'information. Un graphe est constitué de nœuds ou de sommets (représentant les entités du système) qui sont reliés par des arêtes (représentant les relations entre ces entités). Travailler avec des graphes consiste à naviguer à travers les arêtes et les nœuds afin de découvrir et comprendre des relations complexes et/ou d'optimiser les chemins entre des données liées dans un réseau. (91)

1.2.2. Matplotlib Matplotlib :

Est une bibliothèque pour créer et tracer des visualisations statiques, interactives et animées en Python.

1.3. Anacaonda :

Anaconda est une distribution open-source des langages de programmation Python et R pour la science des données qui vise à simplifier la gestion des packages et le déploiement. Les versions des packages dans Anaconda sont gérées par le système de gestion de packages conda, qui analyse l'environnement actuel avant d'exécuter une installation afin d'éviter de perturber d'autres frameworks et packages.(92)

La distribution Anaconda est livrée avec plus de 250 packages installés automatiquement. Plus de 7500 packages open-source supplémentaires peuvent être installés depuis PyPI ainsi que le gestionnaire de packages et d'environnements virtuels conda. Elle comprend également une interface graphique (GUI), Anaconda Navigator, comme une alternative graphique à l'interface en ligne de commande. Anaconda Navigator est inclus dans la distribution Anaconda et permet aux utilisateurs de lancer des applications, de gérer les packages, les environnements et les canaux conda sans utiliser de commandes en ligne. Navigator peut rechercher des packages, les installer dans un environnement, les exécuter et les mettre à jour.

1.4. Spyder :

Spyder est un environnement scientifique gratuit et open-source écrit en Python, pour Python, et conçu par et pour les scientifiques, les ingénieurs et les analystes de données. Il offre une combinaison unique de fonctionnalités avancées d'édition, d'analyse, de débogage et de profilage d'un outil de développement complet, avec les capacités d'exploration des données, d'exécution interactive, d'inspection approfondie et de visualisation magnifique propres à un package scientifique.(93)

2. Description du dataset :

Le réseau social multiplexe est composé de cinq types de relations en ligne et hors ligne (Facebook, Loisirs, Travail, Co-auteurs, Déjeuner) entre les employés du département d'informatique de Aarhus.

Il y a au total 61 nœuds, étiquetés avec des identifiants entiers de 1 à 61, avec 620 connexions. Le multiplexe est non dirigé (une seule direction spécifiée) et non pondéré, stocké sous la forme d'une liste d'arêtes dans le fichier CS-Aarhus_multiplex.edges avec le format suivant :

CS-Aarhus_multiplex.edges : Ce fichier contient les arêtes du réseau multiplexe. Chaque ligne représente une arête et suit le format suivant : "layerID nodeID nodeID weight". Le layerID identifie le type de relation, et les valeurs nodeID correspondent aux identifiants des nœuds impliqués dans la relation. Le poids (weight) est fixé à 1 pour toutes les arêtes dans ce cas.

CS-Aarhus_layers.txt : Ce fichier contient les identifiants de toutes les couches du réseau multiplexe. Chaque ligne représente une couche et inclut l'ID de la couche (layerID).

CS-Aarhus_nodes.txt : Ce fichier contient les identifiants des nœuds dans le réseau multiplexe. Chaque ligne représente un nœud et inclut l'ID du nœud (nodeID).

3. Plan De Travail :

Initialement on a pris quelques etudes sur la prediction des liens dans les reseaux complexes on a trouvé les meilleurs resultats dans l'etude passé . Mais le probleme de son code est qu'il est un peu lent donc on a trouvé que le probleme est dans les fonctions utilisé qui sont basées sur des boucles donc l'accès au mémoire sera un peu lent .

On a va utiliser dans la methode devloppée des fonctions recursives pour pouvoir accelerer le travail on a basé sur les fonctions essentiels et toutes les autres fonctions , tout ça pour avoir une amelioration dans l'acces au variables et ce qui donnera une amelioration dans la vitesse de notre prediction ameliorer l'exactitude de nos resultats .

Etapes et explication du code :

➢ Implémentation 1 :

Le code suis le processus de travail suivant :

1. Importation des modules : Les modules nécessaires tels que `networkx`, `pprint`, `random`, `operator`, `data_class` et `timeit` sont importés en début de script.

2. Définition des chemins des fichiers de données : Les chemins des fichiers de données, tels que les fichiers de nœuds, de couches et d'arêtes, sont définis dans les variables `node_path`, `layer_path` et `edge_path`.

3. Fonction `add_nodes()` : Cette fonction lit le fichier de nœuds spécifié et crée un graphe vide. En parcourant chaque ligne du fichier, elle ajoute les nœuds au graphe avec leurs étiquettes. Le graphe résultant est retourné.

4. Fonction `create_graphs()` : Cette fonction crée une liste de graphes en appelant la fonction `add_nodes()` pour chaque graphe. Ensuite, elle lit le fichier d'arêtes spécifié et ajoute les arêtes aux graphes correspondants en utilisant les identifiants des nœuds. La liste de graphes est renvoyée.

5. Fonction `get_predictors()` : Cette fonction prend une liste de graphes en entrée et crée une liste de couches (instances de la classe `Layer`) à partir de ces graphes. Chaque couche est initialisée avec le graphe correspondant et un poids calculé à l'aide de la fonction `layer_likelihood()`. La liste de couches est renvoyée.

6. Fonction `remove_edges()` : Cette fonction prend un graphe et un nombre `n` en entrée, puis supprime aléatoirement `n` arêtes du graphe. Les arêtes supprimées sont stockées dans une liste et renvoyées.

7. Fonction `layer_likelihood()` : Cette fonction calcule la probabilité de présence d'une arête dans une couche en utilisant la liste des arêtes manquantes, le graphe de prédicteur et un compteur. La fonction est récursive et utilise une approche récursive descendante pour calculer la probabilité.

8. Fonction `calculate_edge_scores()` : Cette fonction calcule les scores des arêtes manquantes et des arêtes inexistantes en fonction des poids des couches et de la présence d'arêtes dans les prédictions. Les scores sont stockés dans les listes `missing_edges_scores` et `non_exist_scores`.

9. Fonction `calculate_auc()` : Cette fonction calcule l'Aire Sous la Courbe (AUC) en comparant les scores des arêtes manquantes et des arêtes inexistantes de manière aléatoire. La fonction est récursive et utilise une approche récursive ascendante pour calculer l'AUC.

10. Fonction `calculate_precision()` : Cette fonction calcule la précision en fusionnant les listes de scores d'arêtes manquantes et d'arêtes inexistantes, puis en triant la liste fusionnée. Elle renvoie la précision en fonction d'un paramètre `delta` spécifié.

11. Bloc principal : Le bloc principal du code exécute les étapes suivantes :

- Création d'un graphe vide pour la cible à partir de la fonction `add_nodes()`.

- Création d'une liste de graphes à partir de la fonction `create_graphs()`.

- Récupération des prédicteurs à partir de la fonction `get_predictors()`.

- Calcul des poids des couches à l'aide de la fonction `layer_likelihood()`.

- Suppression d'arêtes du graphe cible à l'aide de la fonction `remove_edges()`.

- Calcul des scores des arêtes manquantes et inexistantes à l'aide de la fonction `calculate_edge_scores()`.

- Calcul de l'AUC et de la précision à l'aide des fonctions `calculate_auc()` et `calculate_precision()`.

- Affichage des résultats de l'AUC et de la précision pour chaque couche cible.

12. Affichage du temps d'exécution total.

➢ Implementaion 2 :

Ce code implémente un processus de travail qui effectue les tâches suivantes :

1. Importation des modules : Les modules `networkx`, `pprint`, `random`, `operator` et `timeit` sont importés en début de script. Le module `data_class` est également importé pour les classes `Node`, `Layer` et `Edge`.

2. Définition des chemins des fichiers de données : Les chemins des fichiers de données, tels que les fichiers de nœuds, de couches et d'arêtes, sont définis dans les variables `node_path`, `layer_path` et `edge_path`.

3. Définition des couches : Une liste de noms de couches est définie dans la variable `layers`.

4. Mesure du temps de démarrage.

5. Fonction `add_nodes()` : Cette fonction lit le fichier de nœuds spécifié et crée un graphe vide à l'aide de la bibliothèque `networkx`. Elle parcourt chaque ligne du fichier et ajoute les nœuds au graphe en utilisant leurs identifiants et étiquettes. Le graphe résultant est retourné.

6. Fonction `create_graphs()` : Cette fonction crée une liste de graphes en appelant la fonction `add_nodes()` pour chaque graphe. Ensuite, elle lit le fichier d'arêtes spécifié et ajoute les arêtes aux graphes correspondants en utilisant les identifiants des nœuds. La liste de graphes est renvoyée.

7. Fonction `remove_edges()` : Cette fonction prend un graphe et un nombre `n` en entrée, puis supprime aléatoirement `n` arêtes du graphe. Les arêtes supprimées sont stockées dans une liste `missing_edges_list` et renvoyées.

8. Fonction `likelihood()` : Cette fonction prend en entrée un graphe cible et un graphe prédicteur, puis calcule la probabilité de présence d'une arête dans le graphe prédicteur en comparant les arêtes communes entre les deux graphes. La probabilité est renvoyée.

9. Fonction `link_in_predictor_layer()` : Cette fonction vérifie si une arête donnée est présente dans une couche prédicteur. Elle renvoie 1 si l'arête est présente et 0 sinon.

10. Fonction `assign_likelihood()` : Cette fonction prend une liste de graphes, un graphe cible et une liste d'arêtes manquantes en entrée. Elle calcule les poids des couches prédicteurs en utilisant la fonction `likelihood()` et crée une liste d'objets `Edge` contenant les arêtes du graphe cible et leurs scores de probabilité. La liste d'objets `Edge` est renvoyée.

11. Fonction `calculate_auc()` : Cette fonction calcule l'Aire Sous la Courbe (AUC) en comparant aléatoirement les scores des arêtes manquantes et des arêtes inexistantes dans une boucle. Elle renvoie l'AUC calculée.

12. Fonction `precision()` : Cette fonction calcule la précision en triant les scores des arêtes et en comparant les `delta` premières arêtes avec la liste des arêtes manquantes. La précision est renvoyée.

13. Bloc principal : Le bloc principal du code exécute les étapes suivantes :

- Création d'un graphe vide à l'aide de la fonction `add_nodes()`.

- Création d'une liste de graphes en utilisant la fonction `create_graphs()`.

- Suppression d'un certain nombre d'arêtes du graphe cible en utilisant la fonction `remove_edges()` et stockage des arêtes supprimées dans `missing_edges_list`.

- Attribution des poids de probabilité aux arêtes du graphe cible en utilisant la fonction `assign_likelihood()`, ce qui donne une liste d'objets `Edge` contenant les arêtes et leurs scores.

- Calcul de la précision en utilisant la fonction `precision()` et affichage du résultat.

- Mesure du temps d'exécution total et affichage.

Ce code semble être une partie d'un programme plus vaste qui effectue une analyse de graphe et des calculs de probabilité pour évaluer les performances d'un modèle prédictif.

4. Mesures De comparaison :

4.1. Moyenne de précisions :

C'est la moyenne du taux de precision dans chaque couche

$$Mp = \frac{Pc1 + Pc2 + \cdots + Pcn}{Nc}$$

$\boldsymbol{Mp}$: $Moyenne\ Precision$

$\boldsymbol{Pc}$: $Precesion\ de\ couche$

$\boldsymbol{Nc}$: $Nombre\ de\ couches$

4.2. Moyenne d'AUC :

C'est la moyenne du taux de Area under curve dans chaque couche

$$Mauc = \frac{AUCc1 + AUCc2 + \cdots + AUCcn}{Nc}$$

$\boldsymbol{Mauc}$: $Moyenne\ AUC$

$\boldsymbol{AUCc}$: $AUC\ de\ couches$

$\boldsymbol{Nc}$: $Nombre\ de\ couches$

4.3. Temps D'execution :

C'est la moyenne du temps prit par chaque methode pour donner le score de chaque execution

Dans cette etude on va comparer entre deux resultats .

5. Résultats et expérimentation :

5.1. Resultats du methode ancienne :

On va montrer les resultats de l'etude faite par Shikhar Charma et anyrag sing

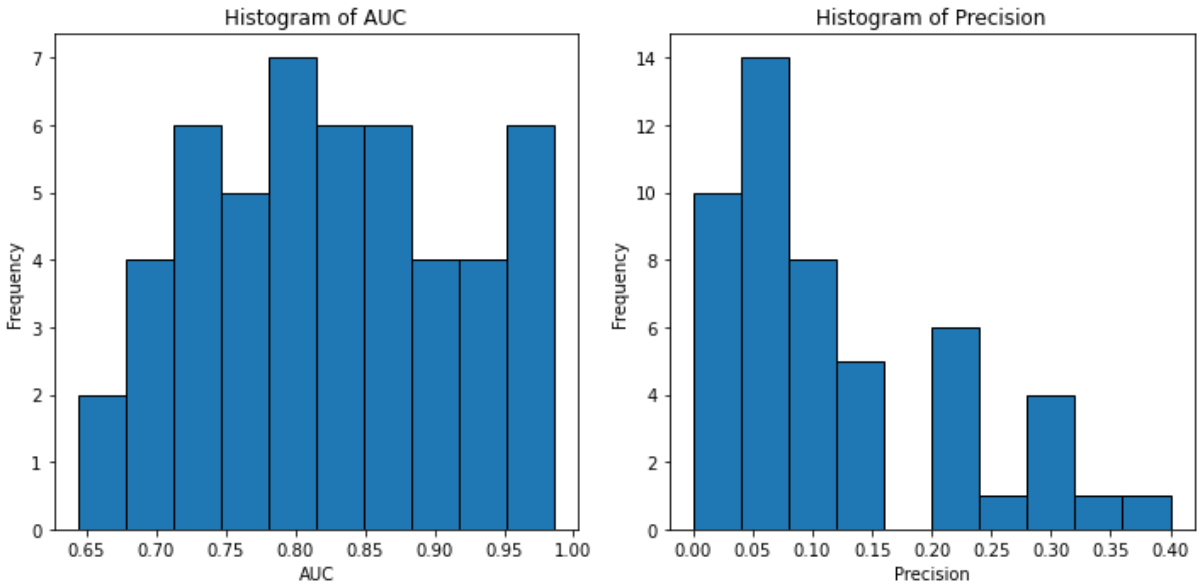

Figure 16 AUC et Precision de 100 implementation

5.2. Résultats de notre méthode :

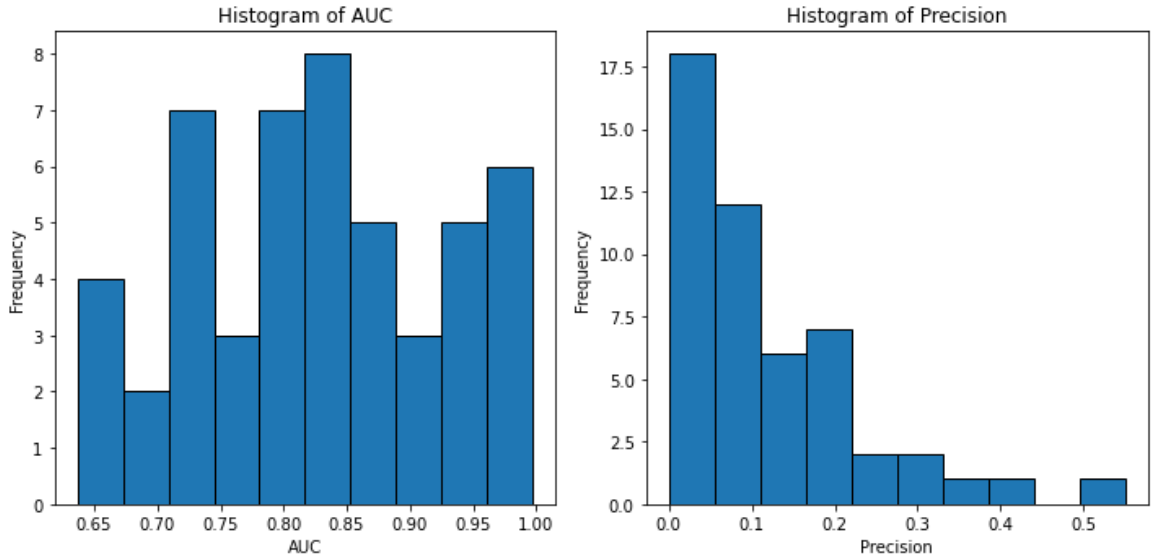

Figure 17 resultat et temps de calcule de AUC et precision

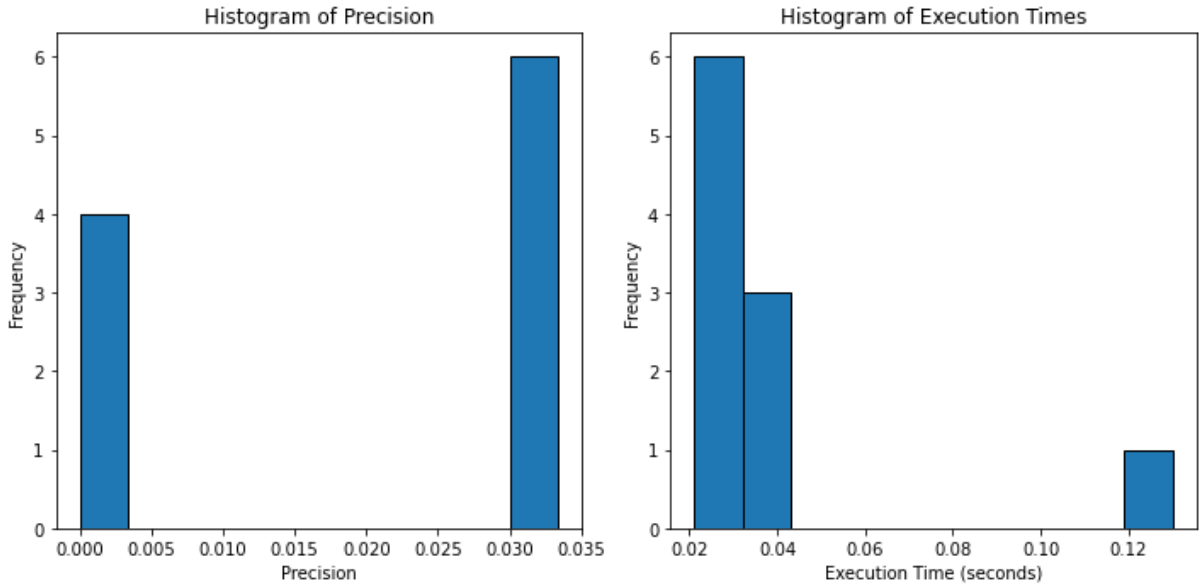

Figure 18 temps de calcule et precision de 10 implenetation

6. Discussion des résultats

Pour evaluer chaque methode nous avons exécuté chaque methode presque dans les memes cas et

On a calculé les moyennes et aussi le temps d'exécution.

Tout d'abord nous avons tracer les graphes de chaque couche et calculer la moyenne , on a eu une moyenne de AUC pour l'ancienne méthode de **0.80** qui est une tres bonne résultat par a port aux autres méthodes et algorithme . Pour notre méthode on a eu **0.850** ce qui signifie que le changement apporter a la fonction ancienne a donner un développement de **6.25%** dans les résultats .

Ensuite On a tracé les graphes a bars pour la precision de chaque couche et on a calculer la moyenne , les résultats été comme suit , pour l'ancienne methode on a eu une moyenne de **0.1**

Et **0.13** pour notre methode ce qui signifie qu'on a eu un devloppement de **30%** dans les resultats .

Le temps de calcule et dessin du graphe été **0.7418ms** pour l'ancienne methode et **0.7381ms** pour notre methode cela ne peut pas etre remarquable dans les petite dataset mais il fera une grande difference dans les dataset geante .

Apres, On a calculer les scores entre deux nœuds avec l'aide de l'algorithme likelihood avec deux méthodes une avec fonction avec boucles et une avec une fonction recursives les resultats n'ont pas dégrader étais les memes pour chaque methode , dans autres testes on a pu minimiser les Missing edges mais c'etais pas le cas pour touts les essais donc nous avons nigliger les missing

edges et on a basé sur l'acceleration de l'ancienne methode qui a fait un temps de **0.12ms** et on a eu un temps de **0.039ms** pour la methode recursives .

Nous conclurons par dire que la methode recursive a ameliorer les resultats des calcules dans un temps moins que toutes les autres methode de prediction de liens dans les reseaux complexes .

Conclusion générale :

A l'occasion de notre projet de fin d'études, nous avons choisi un domaine de recherche très intéressent et récent qui est l'analyse des réseaux complexe.

Ce domaine met en œuvre des techniques intelligentes de recherche et de traitement de connaissances à partir de vastes ensembles de données. Ces connaissances sont utilisées dans divers domaines d'application. Parmi eux, la prédiction des liens dans les réseaux complexes est un domaine de recherche très actif, en raison de la multiplication des documents numériques disponibles aujourd'hui. Avec la croissance continue du flux et du volume d'informations disponibles, il est essentiel de fournir aux individus une compréhension des interactions entre ces données, ainsi que de faciliter la visualisation et la navigation au sein de réseaux de grande taille.

Nous avons consacré la première partie de notre travail(chapitre1 et 2) à une étude théorique. Nous avons abordé les réseaux complexes de manière générale et avons également caractérisé le problème de prédiction des liens, ainsi que les différentes approches et algorithmes existants.

La deuxième partie a été consacrée aux expérimentations. Nous avons testé les methodes utilisés par Sharma and Singh sur la prediction des liens et nous avons montrés les résultats de leur methodes et on a montré les resultats de notre methode devloppée a base de la methode de Sharma and Singh par faire des changement essentiels dans le code pour avoir des resultats mieux.

Les objectifs de notre projet de fin d'étude sont multiples. Tout d'abord, nous visons à acquérir une solide base de connaissances dans le domaine de l'analyse des réseaux complexes. Ensuite, nous souhaitons obtenir une bonne initiation à la recherche scientifique en explorant des problématiques spécifiques liées à ce domaine. De plus, nous cherchons à développer nos compétences en utilisant des outils tels que Python pour l'analyse et la manipulation des données, ainsi que la théorie des graphes pour comprendre les structures et les interactions au sein des réseaux. En résumé, notre projet vise à combiner acquisition de connaissances, initiation à la

recherche et développement de compétences pratiques pour approfondir notre compréhension de l'analyse des réseaux complexes.

A la fin de notre travail nous voulons bien dire que nous avons atteint les objectifs et les buts tracés derrière cette étude.

❖ Liste des références :

[1] Aggarwal, C. C. (2011). An introduction to social network data analytics. In Social network data analytics (pp. 1-15). Springer, Boston, MA.

[2] Tang, F., Mao, C., Yu, J., & Chen, J. (2011, October). Notice of Retraction The implementation of information service based on social network systems. In The 5th International Conferenceon New Trends in Information Science and Service Science (Vol. 1, pp. 46-49). IEEE.

[3] Travers, J., & Milgram, S. (1969). An exploratorystudy of the small world problem. Sociometry, 32, 425-43.

[4] CLAUSET, A. & EAGLE, N. 2012. Persistence and periodicity in a dynamicproximity network. arXivpreprint arXiv:1211.7343.

[5] LESKOVEC, J., KLEINBERG, J. & FALOUTSOS, C. 2005. Graphs over time. Proceeding of the eleventh ACM SIGKDD international conference on Knowledgediscovery in data mining - KDD '05.

ACM Press

[6] Gu, S., Chen, L., Li, B., Liu, W., & Chen, B. (2019). Link prediction on signed social networks basedon latent space mapping. Applied Intelligence, 49(2), 703-722.

[7] Srinivas, V., & Mitra, P. (2016). Link prediction in social networks:role of power law distribution.Springer International Publishing.

[8]Guns, R. (2014). Link prediction. In Measuringscholarlyimpact(pp. 35-55). Springer, Cham.

[9] Tylenda, T., Angelova, R., &Bedathur, S. (2009, June). Towards time-awarelinkprediction in evolving social networks. In Proceedings of the 3rd workshop on social network mining and analysis (p. 9). ACM.

[11] Sarkar, P., Chakrabarti, D., & Jordan, M. (2012). Nonparametriclinkprediction in dynamic networks. arXivpreprint arXiv:1206.6394.

[12] Li, X., Du, N., Li, H., Li, K., Gao, J., & Zhang, A. (2014, April). A deeplearningapproach to linkprediction in dynamic networks. In Proceedings of the 2014 SIAM International Conference on Data Mining (pp. 289-297). Society for Industrial and AppliedMathematics.

[13] Dong, Y., Tang, J., Wu, S., Tian, J., Chawla, N. V., Rao, J., & Cao, H. (2012, December). Link prediction and recommendation across heterogeneous social networks. In 2012 IEEE 12th International conference on data mining (pp.181-190). IEEE.

[14] Ge, L., & Zhang, A. (2012, April). Pseudo cold start linkpredictionwith multiple sources in social networks. In Proceedings of the 2012 SIAM International Conference on Data Mining (pp. 768-779). Society for Industrial and AppliedMathematics.

[15] Kuo, T. T., Yan, R., Huang, Y. Y., Kung, P. H., & Lin, S. D. (2013, August). Unsupervisedlinkpredictionusingaggregativestatistics on heterogeneous social networks. In Proceedings of the 19th ACM SIGKDD international conference on Knowledgediscovery and data mining (pp. 775- 783). ACM.

[16] Raymond, R., & Kashima, H. (2010, September). Fast and scalable algorithms for semi-supervisedlinkprediction on static and dynamic graphs. In Joint europeanconference on machine learning and knowledgediscovery in databases (pp. 131-147). Springer, Berlin, Heidelberg.**s**

[17] Tang, F. (2017). Link-Prediction and its Application in Online Social Networks (Doctoral dissertation, Victoria University).

[18] Jagadishwari, V., &Umadevi, V. (2015). EmpiricalAnalysis of Traditional Link Prediction Methods. International Journal of Computer Applications, 121(2).

[19] Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N., &Barabási, A. L. (2000). The large-scaleorganization of metabolic networks. Nature, 407(6804), 651.

[20] Ito, T., Chiba, T., Ozawa, R., Yoshida, M., Hattori, M., &Sakaki, Y. (2001). A comprehensivetwo- hybridanalysis to explore the yeastproteininteractome. Proceedings of the National Academy of Sciences, 98(8), 4569-4574.**[10]**Rümmele, N., Ichise, R., &Werthner, H. (2015, May). Exploringsupervisedmethods for temporal linkprediction in heterogeneous social networks. In Proceedings of the 24th International Conferenceon World Wide Web (pp. 1363-1368). ACM.

[21] Guelzim, N., Bottani, S., Bourgine, P., &Képès, F. (2002). Topological and causal structure of the yeasttranscriptionalregulatory network. Nature genetics, 31(1), 60.

[22] Albert, R. (2005). Scale-free networks in cellbiology. Journal of cell science, 118(21), 4947-4957.

[23] Zhu, X., Gerstein, M., & Snyder, M. (2007). Gettingconnected:analysis and principles of biological networks. Genes&development, 21(9), 1010-1024.

[24] Barabási, A. L., &Bonabeau, E. (2003). Scale-free networks. Scientific american, 288(5), 60-69.

[25] Clauset, A., Shalizi, C. R., & Newman, M. E. (2009). Power-law distributions in empirical data. SIAM review, 51(4), 661-703.

[26] Schafer, J. B., Frankowski, D., Herlocker, J., & Sen, S. (2007). Collaborative filteringrecommendersystems. In The adaptive web (pp. 291-324). Springer, Berlin, Heidelberg.

[27] Leininger, S., Urich, T., Schloter, M., Schwark, L., Qi, J., Nicol, G. W, &Schleper, C. (2006). Archaea predominateamongammonia-oxidizingprokaryotes in soils. Nature, 442(7104), 806.

[28] Pavlov, M., &Ichise, R. (2007). Finding experts by linkprediction in co-authorship networks. FEWS, 290, 42-55.

[29] Lü, L., & Zhou, T. (2011). Link prediction in complex networks: A survey. Physica A: statistical mechanics and its applications, 390(6), 1150-1170.

[30] Brandes, U., & Wagner, D. (2004). Analysis and visualization of social networks. In Graph drawing software (pp. 321-340). Springer, Berlin, Heidelberg.

[31] Steyvers, M., Miller, B., Hemmer, P., & Lee, M. D. (2009). The wisdom of crowds in the recollection of order information. In Advances in neural information processing systems (pp. 1785-1793).

[32] Tylenda, T., Angelova, R., &Bedathur, S. (2009, June). Towards time-aware link prediction in evolving social networks. In Proceedings of the 3rd workshop on social network mining and analysis (p. 9). ACM.

[33] Yang, Y., Chawla, N., Sun, Y., & Hani, J. (2012, December). Predicting links in multi-relational and heterogeneous networks. In 2012 IEEE 12th international conference on data mining (pp. 755-764).IEEE.

[34] Al Hasan, M., Chaoji, V., Salem, S., &Zaki, M. (2006, April). Link prediction using supervised learning. In SDM06: workshop on link analysis, counter-terrorism and security.

[35] Liben-Nowell, D., & Kleinberg, J. (2007). The link-prediction problem for social networks. Journal of the American society for information science and technology, 58(7), 1019-1031.

[36] Lichtenwalter, R. N., Lussier, J. T., & Chawla, N. V. (2010, July). New perspectives and methods in link prediction. In Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 243-252). ACM.

[37] Sarkar, P., Chakrabarti, D., & Moore, A. W. (2011, June). Theoretical justification of popular link prediction heuristics. In Twenty-Second International Joint Conference on Artificial Intelligence.

[38] Backstrom, L., &Leskovec, J. (2011, February). Supervised random walks: predicting and recommending links in social networks. In Proceedings of the fourth ACM international conference on Web search and data mining (pp. 635-644). ACM.

[39] Lee, K., Agrawal, A., &Choudhary, A. (2013, August). Real-time disease surveillance using twitter data: demonstration on flu and cancer. In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 1474-1477). ACM.

[40] Bliss, C. A., Frank, M. R., Danforth, C. M., &Dodds, P. S. (2014). An evolutionary algorithm approach to link prediction in dynamic social networks. Journal of Computational Science, 5(5), 750-764.

[41] Wang, D., Pedreschi, D., Song, C., Giannotti, F., &Barabasi, A. L. (2011, August). Human mobility, social ties, and link prediction. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 1100-1108). Acm.

[42] Nguyen-Thi, A. T., Nguyen, P. Q., Ngo, T. D., & Nguyen-Hoang, T. A. (2015). Transfer AdaBoost SVM for link prediction in newly signed social networks using explicit and PNR features. Procedia Computer Science, 60, 332-341.

[43] Sarkar, P., & Moore, A. W. (2006). Dynamic social network analysis using latent space models.In Advances in Neural Information Processing Systems (pp. 1145-1152).

[44] Sarkar, P., Chakrabarti, D., & Jordan, M. (2012). Nonparametric link prediction in dynamic networks. arXiv preprint arXiv:1206.6394.

[45] Sewell, D. K., & Chen, Y. (2016). Latent space models for dynamic networks with weightededges. Social Networks, 44, 105-116.

[46] Bordes, A., Glorot, X., Weston, J., &Bengio, Y. (2014). A semantic matching energy function for learning with multi-relational data. Machine Learning, 94(2), 233-259.

[47] Zhu, L., Guo, D., Yin, J., VerSteeg, G., &Galstyan, A. (2016). Scalable temporal latent space inference for link prediction in dynamic social networks. IEEE Transactions on Knowledge and Data Engineering, 28(10), 2765-2777.

[48] Rastelli, R., Friel, N., &Raftery, A. E. (2016). Properties of latent variable network models. Network Science, 4(4), 407-432.

[49] Rahman, M., & Al Hasan, M. (2016, September). Link prediction in dynamic networks using graphlet. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 394-409). Springer, Cham.

[50] Nickel, M., Murphy, K., Tresp, V., &Gabrilovich, E. (2016). A review of relational machine learningfor knowledge graphs. Proceedings of the IEEE, 104(1), 11-33.

[51] Dunlavy, D. M., Kolda, T. G., &Acar, E. (2011). Temporal link prediction using matrix and tensor factorizations. ACM Transactions on Knowledge Discovery from Data (TKDD), 5(2), 10.

[52] Ermiş, B., Acar, E., &Cemgil, A. T. (2012). Link prediction via generalized coupled tensor factorisation. arXiv preprint arXiv:1208.6231.

[53] Gao, S., Denoyer, L., &Gallinari, P. (2011, April). Link pattern prediction with tensor decompositionin multi-relational networks. In 2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM) (pp. 333-340). IEEE.

[54] Menon, A. K., &Elkan, C. (2011, September). Link prediction via matrix factorization. In Joint european conference on machine learning and knowledge discovery in databases (pp. 437-452). Springer, Berlin, Heidelberg.

[55] Haghani, S., &Keyvanpour, M. R. (2017). A systemic analysis of link prediction in social network. Artificial Intelligence Review, 1-35.

[56] Spiegel, S., Clausen, J., Albayrak, S., &Kunegis, J. (2011, May). Link prediction on evolving data using tensor factorization. In Pacific-Asia Conference on Knowledge Discovery and Data Mining (pp. 100-110). Springer, Berlin, Heidelberg.ss

[57] Yao, L., Sheng, Q. Z., Qin, Y., Wang, X., Shemshadi, A., & He, Q Context-aware point-fIiterest commendation using tensor factorization with social regularization. In Proceedings

of the 38th ternational ACM SIGIR conference on research and development in information retrieval (pp. 1007- 010). ACM.

[58] Han, Y., &Moutarde, F. (2016). Analysis of large-scale traffic dynamics in an urban transportation network using non-negative tensor factorization. International Journal of Intelligent Transportation Systems Research, 14(1), 36-49.

[59] Nickel, M., &Tresp, V. (2013, September). Tensor factorization for multi-relational learning. In JointEuropean Conference on Machine Learning and Knowledge Discovery in Databases (pp. 617-621). Springer, Berlin, Heidelberg.

[60] London, B., Rekatsinas, T., Huang, B., &Getoor, L. (2013). Multi-relational learning using weighted tensor decomposition with modular loss. arXiv preprint arXiv:1303.1733.

[61] Nickel, M., Jiang, X., &Tresp, V. (2014). Reducing the rank in relational factorization models by including observable patterns. In Advances in Neural Information Processing Systems (pp. 1179-1187).

[62] Keyvanpour, M. R., & Moradi, S. S. (2014). A perturbation method based on singular valuedecomposition and feature selection for privacy preserving data mining. International Journal of DataWarehousing and Mining (IJDWM), 10(1), 55-76.

[63] Narita, A., Hayashi, K., Tomioka, R., & Kashima, H. (2012). Tensor factorization using auxiliaryinformation. Data Mining and Knowledge Discovery, 25(2), 298-324.

[64] Yılmaz, K. Y., Cemgil, A. T., &Simsekli, U. (2011). Generalised coupled tensor factorisation.In Advances in neural information processing systems (pp. 2151-2159).

[65] Nakatsuji, M., Toda, H., Sawada, H., Zheng, J. G., &Hendler, J. A. (2016). Semantic sensitive tensor factorization. Artificial Intelligence, 230, 224-245.

[66] Jiang, X., Tresp, V., Huang, Y., & Nickel, M. (2012). Link Prediction in Multi-relational Graphs using Additive Models. SeRSy, 919, 1-12.

[67] Riedel, S., Yao, L., McCallum, A., & Marlin, B. M. (2013). Relation extraction with matrix factorization and universal schemas. In Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 74-84).

[68] Wang, C., Satuluri, V., & Parthasarathy, S. (2007, October). Local probabilistic models for linkprediction. In Seventh IEEE international conference on data mining (ICDM 2007) (pp. 322-331). IEEE.

[69] Wang, C., Satuluri, V., & Parthasarathy, S. (2007, October). Local probabilistic models for linkprediction. In Seventh IEEE international conference on data mining (ICDM 2007) (pp. 322-331). IEEE.

[70] Nguyen, C. H., &Mamitsuka, H. (2011, September). Kernels for link prediction with latent feature models. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 517-532). Springer, Berlin, Heidelberg.

[71] Feng, X., Zhao, J. C., & Xu, K. (2012). Link prediction in complex networks: a clusteringperspective. The European Physical Journal B, 85(1), 3.

[72] Adamic, L. A., & Adar, E. (2003). Friends and neighbors on the web. Social networks, 25(3), 211-230.

[73] Barabási, A. L., & Albert, R. (1999). Emergence of scaling in random networks. science, 286(5439),509-512.

[74] Mitzenmacher, M. (2004). A brief history of generative models for power law and lognormaldistributions. Internet mathematics, 1(2), 226-251.

[75] Rossetti, G., Guidotti, R., Pennacchioli, D., Pedreschi, D., & Giannotti, F. (2015, August). Interactionprediction in dynamic networks exploiting community discovery. In 2015 IEEE/ACM InternationalConference on Advances in Social Networks Analysis and Mining (ASONAM) (pp. 553-558). IEEE.

[76] Martínez, V., Berzal, F., & Cubero, J. C. (2017). A survey of link prediction in complexnetworks. ACM Computing Surveys (CSUR), 49(4), 69.

[77] Newman, M. E. (2001). Clustering and preferential attachment in growing networks. Physical reviewE, 64(2), 025102.

[78] Wang, P., Xu, B., Wu, Y., & Zhou, X. (2015). Link prediction in social networks : the state-of-the-art. Science China Information Sciences, 58(1), 1-38.

[79] Dunlavy, D. M., Kolda, T. G., &Acar, E. (2011). Temporal link prediction using matrix and tensorfactorizations. ACM Transactions on Knowledge Discovery from Data (TKDD), 5(2), 10.

[80] Fire, M., Tenenboim, L., Lesser, O., Puzis, R., Rokach, L., &Elovici, Y. (2011, October). Li prediction in social networks using computationally efficient topological features. In 2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing (pp. 73-80). IEEE.

[81] Zhai, S., & Zhang, Z. (2015, June). Dropout training of matrix factorization and autoencoder for link prediction in sparse graphs. In Proceedings of the 2015 SIAM International Conference on Data Mining (pp. 451-459). Society for Industrial and Applied Mathematics.

[82] Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and newperspectives. IEEE transactions on pattern analysis and machine intelligence, 35(8), 1798-1828. **[83]**Socher, R., Chen, D., Manning, C. D., & Ng, A. (2013). Reasoning with neural tensor networks forknowledge base completion. In Advances in neural information processing systems (pp. 926-934).

[84] Li Deng, D. Y. (2014). Deep learning: methods and applications. Tech. rep., https://www.microsoft.com/en-us/research/publication/deep-learning-methods-and-applications.

[85] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning http://www.deeplearningbook.org.

[86] Liu, F., Liu, B., Sun, C., Liu, M., & Wang, X. (2013, November). Deep learning approaches for link prediction in social network services. In International Conference on Neural Information Processing (pp. 425-432). Springer, Berlin, Heidelberg.

[87] Li, K., Gao, J., Guo, S., Du, N., Li, X., & Zhang, A. (2014, December). Lrbm: A restricted boltzmannmachine based approach for representation learning on linked data. In 2014 IEEE International Conference on Data Mining (pp. 300-309). IEEE.

[88] An effficient méthod for link prédiction in weighted multiplex networks,springer 2016.

[89] https://lycee-champollion.fr/IMG/pdf/cours_article-2.pdf

[90] https://www.geeksforgeeks.org/recursive-functions/

[91] nvidia.com

[92] anaconda.com

[93] spyder.com

❖ Liste des références de figure:

Fig1 https://zestedesavoir.com/

Fig2 https://www.monde-economique.ch/

Fig3 https://www.scirp.org/

Fig4 https://www.w3.org/2001/sw/Europe/events/foaf galway/papers/fp/bootstrapping_the_foaf_web/

Fig5 https://lipn.univ-paris13.fr/~kanawati/ars/ARS-cours4-VertexSim.pdf

Fig6 https://link.springer.com/

Fig 7 https://www.cairn.info/

Fig 8 https://link.springer.com/

Fig 9 https://www.semanticscholar.org/

Fig 10 https://link.springer.com/

Fig 11 https://www.researchgate.net/

Fig 12 https://www.springeropen.com/

Fig 13 https://www.researchgate.net/

Fig 14 https://www.sciencedirect.com/

Fig 15 https://www.springeropen.com/