Auteurs : MOUTTAPA Arnaud, CHAROTTE Edwin, TELITSINE Thibault

Recommandation de liens entre pages Wikipedia avec méthode multicritère

Rapport dans le cadre du cours d'*Analyse des Réseaux Sociaux*
(3ème année de cycle ingénieur - Data Science)

À l'attention de M.KANAWATI Rushed.

Janvier 2024

Table des matières

Introduction

Wikipédia est chaque jour augmenté de nouvelles pages, de nouvelles connaissances dans tout les domaines encyclopédiques possibles. Ces innombrables pages se font réferences (les liens hypertextes) les une aux autres permettant de se compléter, permettant au lecteur d'aller plus loin dans le sujet ou alors de prendre du recul sur le sujet qui serait trop spécifique. Il est alors clair que certaines pages, à leur publication par exemple, manquent de réferences (en possèdent 1 ou 2) qui donneraient la possibilité de mieux comprendre la page, de mieux la sourcer.
L'objectif de ce projet est de pallier ce manque de reférences en utilisant une méthode de recommandation de liens issue de l'analyse des réseaux sociaux.

La recommandation de lien est la tâche consistant à suggérer des connexions pertinentes entre les sommets d'un graphe sur la base de ses liens existants. Notre objet d'étude est un graphe orienté des liens hypertexte des articles de Wikipedia, site qui est "open-source", ce qui implique qu'il est imcomplet car rempli au fur et à mesure par ses utilisateurs. Une utilisation d'un algorithme de recommandation de liens permettrait donc d'ajouter des liens de référencement manquants d'une page cible ayant peu de référence, vers une autre page pertinente par rapport à certains critères définis.

Dans ce rapport, nous développons une approche multicritère de recommandation de liens. Il s'agit de prendre en compte plusieurs critères lors de l'évaluation des meilleurs candidats pour ajouter un lien manquant au sommet cible.

1 Préparation de la donnée

Le graphe Wikipédia dont nous disposons est un graphe constitué de 27475 sommets qui représentent chacun un article. Un sommet est caractérisé par son wikiid (un identifiant unique à chaque page) ainsi que par un label qui correspond au nom de l'article.

Le graphe est aussi constitué de 85729 arcs. Un arc est établi d'un sommet à un autre si la première page fait référence à l'autre via un lien hypertexte. Nous cherchons à établir, depuis le sommet cible, un arc sortant vers une page qui serait similaire "thématiquement". L'idée est de compléter au mieux la page peu référencée avec la page la plus similaire non liée à la page cible. Ainsi, afin de nous adapter à notre objectif, nous considérons dans la suite du rapport que les voisins v' d'un sommet v sont les sommets pour lesquels il existe un lien du sommet v vers un sommet v'. Autrement dit, on considère les voisins au sens du mode "out" (comme utilisé dans R).

Dans l'optique d'extraire une communauté égo-centrée pour les sommets cibles, il est nécessaire de travailler sur une composante connexe. La première étape de la construction du graphe de test est donc de sélectionner la composante connexe la plus grande G1 du graphe Wikipedia. Cette étape n'est pas contraignante dans le sens où nous disposons de la majorité du graphe pour tester notre méthode : G1 possède 25178 sommets et 82914 liens, ce qui représente 91.6% des sommets d'origine et 96,7% des arêtes d'origine.

Sur G1, nous enlevons un pourcentage variable d'arêtes parmi les liens des sommets de degré 'out' faible (entre 2 et 5). En effet, l'objectif final de notre algorithme étant de recommander pertinemment des liens sur des sommets ayant peu de références externes, nous avons choisi de nous concentrer sur une suppression d'arêtes sur des sommets de petit degré afin de simuler le contexte réel d'utilisation de notre algorithme.

Les sommets cibles V1 sont les sommets de G1 qui ont perdu au moins un lien extérieur. Nous récupérons ces sommets sous forme de liste de couples contenant chacun le wikiid du sommet et le nombre de liens "out" qu'il a perdus. En sortie, nous avons un graphe potentiellement non connexe dû à la possible suppression de ponts. Encore une fois, il faut extraire la composante connexe la plus grande. En pratique, cette composante connexe G2 est suffisamment grande pour construire nos communautés égo-centrées. En effectuant cette opération, nous listons également les sommets cibles qui sont présents dans la seconde composante. Cette liste ainsi que la liste des couples précédente nous permettront de faire deux calculs de précision différente : un calcul de précision générale et un calcul de précision qui ne prend pas en compte l'erreur systématique que nous faisons sur les sommets cibles qui ne sont pas compris dans G2.

Ainsi, après cette étape, nous disposons d'un graphe connexe avec des arêtes supprimées,

d'une liste de couples wikiid-nombre de liens retirés et d'une liste des sommets de V1 appartenant à G2.

L'ensemble des étapes décrites précédemment sont codées dans la fonction wiki_deletion.

Voici un résumé de la construction du graphe de test :

Entrée : le graphe Wikipedia, un pourcentage de liens à retirer

Sortie : le triplet (graphe G2, liste de couples (wikiid-nombre de liens retirés), liste des sommets de V1 dans G2)

Algorithme :

- Extraction de la composante connexe la plus grande G1 depuis le graphe Wikipedia
- Suppression aléatoire de liens parmi ceux des sommets de degré faible selon le pourcentage en entrée.
- Construction de la liste des couples (wikiid des sommets cibles -nombre de liens retirés)
- Extraction de G2, la composante connexe la plus grande de G1. Du fait de la suppression des liens, le G1 modifié peut perdre sa connexité.
- Construction de la liste des sommets cibles appartenant à G2.
- Renvoi du résultat de sortie

2 Fonction de calcul des modularités

L'approche choisie originellement était le "Combine then rank" vu en cours : la moyenne de plusieurs modularités standardisées donnait une modularité finale. La standardisation a cependant posé problème : la qualité est majorée à 1, une valeur très rapidement atteinte durant les essais unitaires. Cela impliquait donc de comparer les vecteurs de modularités sans standardisation lors de l'évaluation du critère d'arrêt. Finalement, une approche "Ensemble ranking" semblait plus directe et a remplacé l'ancienne approche. Pour effectuer notre Ensemble Ranking, nous utilisons les trois modularités locales dont nous disposons : la modularité R, la modularité M et la modularité L . Comme nous sommes dans un contexte de graphe orienté, nous avons adapté ces fonctions, afin de compter des arcs plutôt que des arêtes non dirigées. En plus de cette modification, nous avons ajouté une exception lorsque S devient vide. Ce cas peut arriver au début de la construction de la communauté égo-centrée lorsque le sommet cible a un unique voisin et que ce dernier n'en possède pas.

L'ensemble des étapes décrites précédemment sont codées dans la fonction Q_generator.

Voici un résumé de la fonction de calcul de modularité :

Entrée :

- Le graphe de test G2
- Les ensembles de sommets C, B et S
- Un vecteur mod_vector contenant les fonction de modularité à utiliser

Sortie : Un couple (Sommet optimal à ajouter dans B, Vecteur de modularités de la communauté issue de l'ajout du sommet optimal dans B)

Algorithme :

- Construction de la matrice ($Q\_values = (q_{i,j})$) telle que $q_{i,j}$ correspond à l'image du j-ème sommet dans S par la i-ème modularité dans mod_vector
- Création d'un classement par modularité des sommets selon la valeur de modularité obtenu après ajout du sommet dans la communauté
- Combinaison des classements par modularité avec l'algorithme de vote de Borda
- Renvoi du meilleur sommet selon le vote de Borda ainsi que de son vecteur de qualités de communauté associé

3 Méthode de recommandation de liens

Après avoir récupéré la communauté égo-centrée de v, nous pouvons passer à la recommandation de lien via la fonction "ajout_lien".

La fonction “ajout_lien” possède 5 arguments : le graphe G2, la communauté de v, les similarités, le sommet cible et le nombre d’ajout de liens nécessaires.

Dans un premier temps, la fonction appelle la fonction de Borda_ER permettant d’avoir un classement des similarités selon Borda entre le sommet cible v et les autres sommets de la communauté. Encore une fois, c’est dans l’optique d’opérer de manière multi-critère. Les similarités de jaccard et de sorensen ont été choisies pour le voisinage, ainsi qu’une similarité de Katz tronquée pour que le chemin influe dans la sélection. Katz a été paramétré avec beta =0.01 et un ordre de 3 Le graphe étant orienté et en loi de puissance, la notion de voisin commun est modifiée et trouver un voisin commun semble plus complexe. Nous pensons donc que les similarités centrées chemin donneront un résultat plus pertinent que celles centrées voisinage.

Le classement étant effectué, on supprime les sommets non candidats c’est-à-dire les voisins direct de v tout en gardant le classement.

Ensuite nous pouvons ajouter les liens en faisant attention à 2 choses : que la liste des candidats ne soit pas nulle et que le nombre de liens à rajouter ne soit pas supérieur aux nombres de candidats possibles. Dans le premier cas, aucun ajout de lien ne peut être fait. Cela peut être le cas lorsque la communauté est composée du sommet cible relié aux autres sommets. Dans le deuxième cas, on effectue le nombre d’ajouts maximum possible.

L’ensemble des étapes décrites précédemment sont codées dans la fonction ajout_lien qui fait appel à Borda_ER

Voici un résumé de la méthode de recommandation de liens :

Entrée :

- Un graphe g (étant issu d’un certain nombre d’ajouts de liens depuis G2)
- Le sommet cible considéré v
- La communauté égo-centrée en le sommet cible v
- Un vecteur de similarité similarities
- Le nombre d’ajouts de liens nécessaires nb_ajout : cette valeur est nécessaire pour le test. Un sommet candidat peut avoir perdu plusieurs liens lors de la suppression des liens.

Sortie :

- Un graphe étant une altération de G2 avec un nombre d'ajout égal à nb_ajout de liens jugés les plus pertinents
- Un entier indiquant si la communauté égo-centrée fournie ne contenait que deux sommets dont le sommet cible, ou bien si la liste des candidats est nulle (ce qui arrive si le sommet cible est relié à tous les autres sommets de la communauté)

Algorithme :

- Pour chaque sommet v' dans la communauté égo-centrée de v ($v' \neq v$) :
  - Pour chaque similarité sim dans similarities :
    - Calcul de sim(v,v')
    - Classement de chaque v' selon leur valeur de similarité dyadique avec v
- Combinaison des classements selon l'algorithme de Borda
- Filtrage des sommets candidats en supprimant les voisins directs de v
- Sélection des nb_ajout meilleurs sommets selon le vote de Borda
- Ajout des liens à g
- Retour du graphe g modifié et de 1 si (la communauté fournie ne contient que deux sommets) ou (la liste des sommets candidats est vide)

4 Méthode de vérification des résultats de prévision

Après avoir ajouté les liens manquants, nous pouvons procéder à la vérification des liens. Pour cela, on itère sur les sommets cibles en comparant leur voisins dans le graphe connexe de wikipédia avec leur voisin dans le graphe reconstitué.
Deux précisions sont calculées :

1. Une précision standard, dégradée par des prédictions impossibles pour notre modèle (sommets isolés lors de suppressions d'arcs)
2. Une précision excluant les prédictions impossibles mentionnées auparavant

.

L'objet du test est le calcul de la performance d'une prédiction pour les sommets qui seront ciblés en pratique par l'algorithme, et non l'efficacité de la recherche de nœuds cibles. Ainsi, comme il y a autant de prédictions que de suppressions, le rappel se confond avec la précision.

Voici un résumé de la vérification de nos résultats de prévision :

— À partir du graphe G2, application de notre méthode de recommandation sur les sommets cibles présents dans G2 en prenant en compte le nombre de liens perdus

— Après ajout de liens dans G2, comparaison des voisins des sommets cibles dans G2 avec ceux dans G1

— Calcul des deux mesures de précision :

  — Une mesure de précision globale égale à $\frac{\text{"nombre de liens bien prévus"}}{\text{nombre de liens retirés}}$

  — Une mesure de précision ne prenant pas en compte l'erreur systématique sur les sommets cibles n'appartenant pas à G2 égale à $\frac{\text{nombre de liens bien prévus dans G2}}{\text{nombre de liens retirés dans G2}}$

5 Analyse des résultats obtenus

Les essais menés sont réalisés pour un pourcentage de liens de supprimer de 0.5
Précisions avec toutes les modularités et toute les similarités utilisées :

— sans sommets isolés : 0.0049

— avec sommets isolés : 0.048

Précisions avec modularité R seulement et toutes les similarités :

— sans sommets isolés : 0.0025

— avec sommets isolés : 0.0024

Précisions avec toutes les modularités et similarité de Katz seulement :

— Précision nulle

Ces résultats sont évidemment très peu satisfaisant. Nous avons quelques explications possible pour expliquer les niveaux de précision faibles :

— Les essais ont montré que les voisins qui auraient dû être créés n'ont pas pu l'être car dans 80% des itérations, ces voisins supprimés n'étaient pas présents dans la communauté égo-centrée. Le sommet qui aurait dû être relié n'était pas dans la communauté égo-centrée. Le problème viendrait donc probablement de la création de communauté. Pour améliorer notre algorithme, il faudrait trouver une autre façon que la communauté égo-centrée pour considérer les sommets candidats

— 1/4 des liens n'ont pas pu être créés car la communauté comprenait le sommet cible relié au reste des sommets. Il n'y avait donc pas de sommet candidat pour effectuer la prévision de liens

— La distribution des degrés en loi de puissance pourrait suggérer une grande probabilité de créer des composantes connexes lorsqu'on retire un arc. Ainsi il est très probable lors de la suppression de liens de ne plus avoir de chemin pour retrouver le sommet à relier

— Le test proposé n'était peut-être pas adéquat à notre but. Un test adéquat aurait peut-être été de tester notre algorithme sur des sommets de petit degré, d'observer les liaison effectuées puis de comparer thématiquement les labels entre le sommet cible et le sommet relié

Conclusion

Avec ce projet, nous avons développé une méthode de prévision de liens multicritère dans le sens où elle prend plusieurs modularités en compte dans la construction de la communauté égo-centrée, et plusieurs similarités pour le choix du sommet à relier. Malgré ces résultats très insatisfaisant, nous avons pu ressortir grandis de cette expérience. Nous avons pu approfondir nos connaissances dans la prévision de liens, nous améliorer dans l'utilisation du langage R, et apprendre à collaborer de manière plus organisée, notamment à travers Github que nous n'avions jamais utilisé avant. Nous avons également profité de ce projet pour rédiger un rapport en LateX, chose que nous n'avions pas faite auparavant. Ainsi, nous finissons ce projet avec des résultats peu satisfaisants, mais également avec une expérience valorisante pour la suite de notre parcours.

Références

1. Rushed Kanawati, *Similarités topologiques de nœuds dans les graphes de terrain*, 2014
2. Rushed Kanawati, *Détection de communautés dans les grands graphes de terrain*, 2016
3. Rushed Kanawati, *Local Community Computation*, 2016
4. Rushed Kanawati, *Travaux pratiques et corrections*, 2023