Contribution à la librairie python de recherche CDlib

Amine ITJI

Février 2024

Résumé : Ce projet vise à enrichir la librairie Python CDlib, spécialisée dans l'analyse des communautés dans les réseaux complexes. Les contributions incluent le développement de nouvelles visualisations de communautés, l'extension de fonctionnalités existantes et l'amélioration de la documentation. L'objectif est de consolider CDlib en tant qu'outil pour l'analyse et la visualisation de réseaux dans l'écosystème Python.

Mots-clés : Réseaux, Visualisation, Données, Algorithmes, Contribution.

Table des matières

1. Introduction

Dans un monde où les interactions humaines, biologiques et technologiques s'entremêlent de manière complexe, la visualisation des réseaux nous offre une fenêtre pour comprendre ces dynamiques. Au cœur de cette discipline, le projet s'est concentré sur le développement et l'amélioration des méthodes de visualisation des communautés, avec pour objectif principal d'enrichir la librairie CDlib.

Sous la supervision de Rémy Cazabet, cofondateur de CDlib, le projet a pris racine dans cette expertise établie, avec pour ambition d'apporter une contribution significative à la visualisation de réseaux de données. Ce rapport documente ainsi le processus de collaboration, depuis la conception jusqu'à l'implémentation pratique de nouvelles fonctionnalités. À travers ce projet, j’ai exploré les défis et les opportunités liés à la visualisation de réseaux.

Ce rapport offre ainsi un aperçu détaillé de mon travail ainsi que certaines perspectives au sein de la librairie CDlib.

2. Présentation de CDlib

CDlib est une librairie d'analyse de réseaux et de détection de communautés qui a été développée par une équipe de 4 chercheurs:

- Giulio Rossetti qui supervise la conception globale de la librairie.
- Letizia Milli contribue à l'intégration des modèles de communauté.
- Salvatore Citraro contribue, également, à l'intégration des modèles de communauté.
- Rémy Cazabet responsable de la visualisation des résultats.

Ensemble, ils fournissent des outils puissants pour l'analyse de réseaux, témoignant d'une qualité et d'une fiabilité exceptionnelle. Avec une couverture Codecov de 95%, des réussites dans les tests et la couverture sous Ubuntu et CodeQL, ainsi que des téléchargements mensuels de 10 000 et un total de 405 000 téléchargements, elle est largement utilisée dans la communauté des chercheurs et des praticiens, avec 167 utilisateurs qui dépendent de celle-ci dans leurs projets. De plus, avec sa disponibilité sur SoBigData++, ses 70 Forks, ses 340 Stars et 19 contributeurs, CDlib bénéficie d'une popularité mondiale, affirmant son rôle essentiel dans l'analyse de réseaux et la détection de communautés.

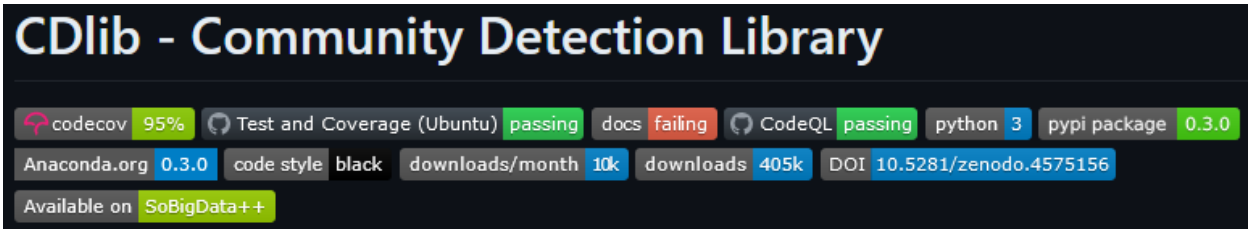

Figure 1 : Démonstration de la fiabilité de CDlib.

Dans le cadre de ma contribution, améliorer l'aspect de la visualisation des graphes est un objectif concret, car cela permettra une meilleure compréhension des résultats de détections de communautés. Une visualisation efficace peut aider à identifier les structures communautaires et à explorer les relations entre les entités, facilitant ainsi l'interprétation des données et la prise de décision.

3. Visualisation de graphes

Actuellement, la librairie propose deux fonctions de visualisation : ***plot_network_clusters*** et ***plot_community_graph***. J'ai décidé d'apporter ma contribution à ces fonctions en y ajoutant des idées nouvelles pour offrir une visualisation plus intuitive et plus complète.

La fonction ***plot_network_clusters*** constitue un outil essentiel pour la visualisation des réseaux complexes. Elle offre une représentation graphique détaillée où chaque nœud du réseau est attribué à une communauté spécifique, ce qui permet une identification visuelle rapide des structures communautaires. En colorant les nœuds en fonction de leurs affiliations communautaires, cette fonction met en évidence les regroupements et les interactions entre les entités du réseau. Les paramètres optionnels, tels que la position des nœuds ou leurs tailles, offrent une flexibilité supplémentaire pour adapter la visualisation aux besoins spécifiques de l'utilisateur.

D'autre part, la fonction ***plot_community_graph*** va encore plus loin en offrant une représentation graphique plus complexe dans laquelle chaque nœud représente une communauté détectée dans le réseau. Les arêtes entre les nœuds indiquent les connexions entre ces communautés. Cette fonction permet, ainsi, une analyse approfondie des interactions entre les communautés, facilitant ainsi la compréhension de la connectivité du réseau dans son ensemble.

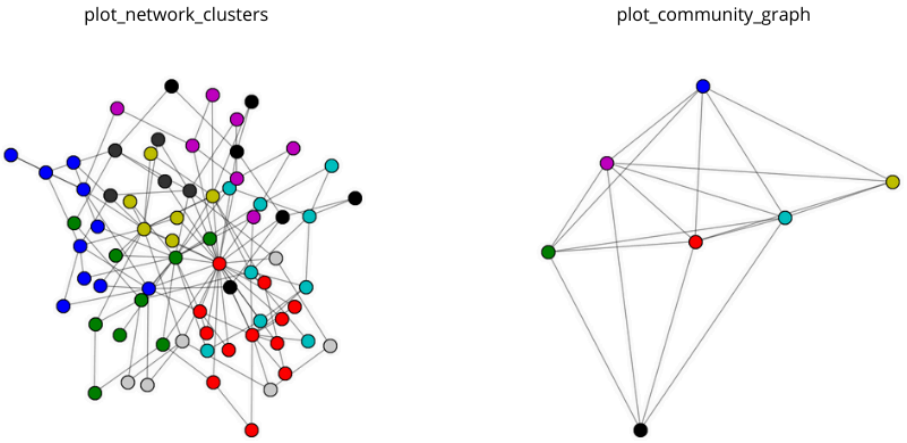

Figure 2 : Démonstration des fonctions de visualisation de réseaux.

4. Nouvelles fonctionnalités

Mes contributions apportées à ***plot_network_clusters*** et ***plot_community_graph*** sont la résultante de l'intégration des nouveaux paramètres *show_edge_widths*, *show_edge_weights* et *show_node_sizes*, ces paramètres offrent une gamme élargie d'options de visualisation, permettant aux utilisateurs d'explorer plus en profondeur la structure et les interactions des réseaux détectés.

La fonction ***plot_network_clusters*** a été améliorée pour permettre l'ajustement de l'épaisseur des arêtes et des nœuds en fonction de leurs poids, ainsi que l'affichage numérique des poids des arêtes directement sur la visualisation. Ceci permet aux utilisateurs de mieux appréhender la force des liens entre les nœuds et d'identifier les connexions clés dans le réseau.

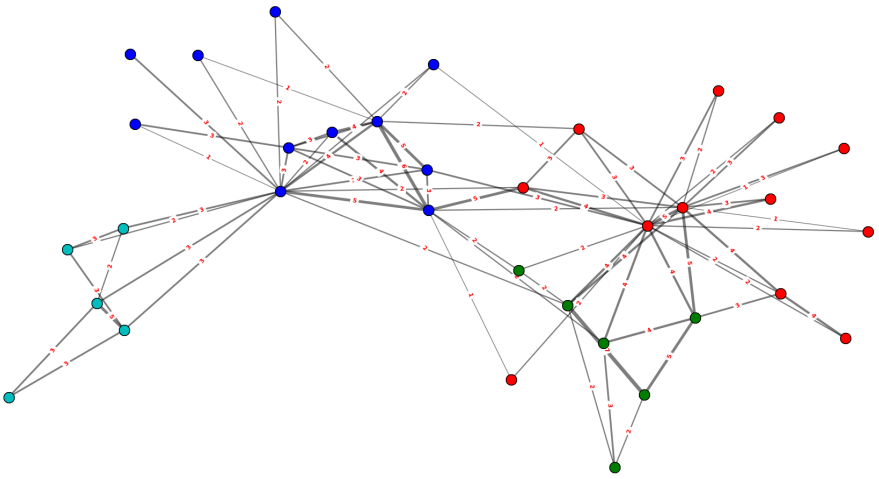

Figure 3 : Démonstration de la fonction plot_network_clusters après les modifications.

Quant à ***plot_community_graph***, elle a été étendue pour inclure les mêmes paramètres. Désormais, elle permet une représentation plus précise des relations entre les communautés en se basant sur la somme des poids des arêtes inter-communautaires, mais aussi la somme des poids de chaque nœud d'une même communauté. Pour une plus grande modularité, j'ai fait en sorte créé un poids tel que *'weight=1'* pour chaque nœud et arête dans le cas où le graphe ne comporte aucun détail sur le poids des nœuds et des arêtes. Ainsi le poids peut, également, symboliser le nombre de nœuds dans une communauté ou le nombre d'arêtes liés entre les communautés.

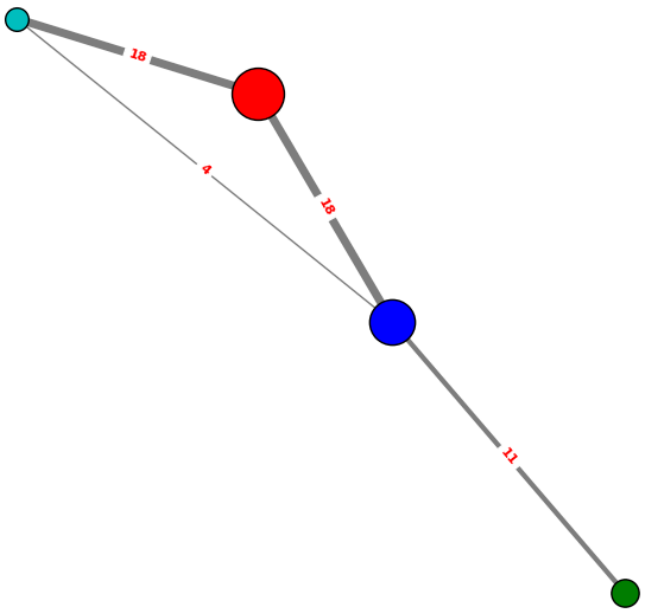

Figure 4 : Démonstration de la fonction plot_community_graph après les modifications.

Pour parvenir à ces améliorations, il était nécessaire d'avoir de nouvelles fonctionnalités sous-jacentes, notamment ***calculate_cluster_edge_weights*** et ***calculate_cluster_sizes*** que j'ai dû ajouter à ma contribution. La fonction ***calculate_cluster_edge_weights*** calcule les poids des arêtes (ou son nombre, cela dépend des paramètres du graphe) inter-communautaires, permettant ainsi de pondérer les connexions entre les différentes communautés. De même, la fonction ***calculate_cluster_sizes*** est cruciale pour déterminer la taille d'un noeud, en fonction de la somme des poids de chaque individu d'une même communauté (ou son nombre, cela dépend des paramètres du graphe) , fournissant ainsi une représentation précise de l'importance relative des communautés ou des nœuds dans le réseau.

Parmi mes contributions, il y a la création d'une nouvelle fonction. Il s'agit de la fonction ***plot_network_highlighted_clusters*** qui positionne stratégiquement les nœuds en fonction des poids. Elle favorise une disposition où les liens forts intra-cluster sont privilégiés, rapprochant ainsi les nœuds au sein d'une même communauté.

J'ai fait appel à la fonction ***spring_layout*** de *NetworkX* qui, généralement, harmonise la position des nœuds et rend le graphe lisible. Cependant, j'ai introduit un paramètre *weight* dans cette fonction, qui, en plus d'harmoniser le graphe, rapproche les nœuds qui partagent des arêtes avec un poids fort et éloigne les nœuds qui partagent des arêtes de poids faible. Pour ce faire, j'ai écrit un code qui cible les liens intra-communautaires et leur attribue un poids élevé, tel que *'weight=200'*, tandis que les autres

liens sont attribués à un poids faible *'weight=1'*. Ensuite, j'ai mis en évidence visuellement les communautés en colorant les nœuds et leurs arêtes appartenant à une même communauté avec la même couleur. Pour renforcer cette représentation, j'ai ajouté un polygone autour de chaque communauté, délimitant ainsi visuellement les frontières de chaque cluster.

Cette approche offre une nouvelle représentation visuelle claire et intuitive des structures communautaires, mettant en lumière les interactions significatives à l'intérieur de chaque cluster.

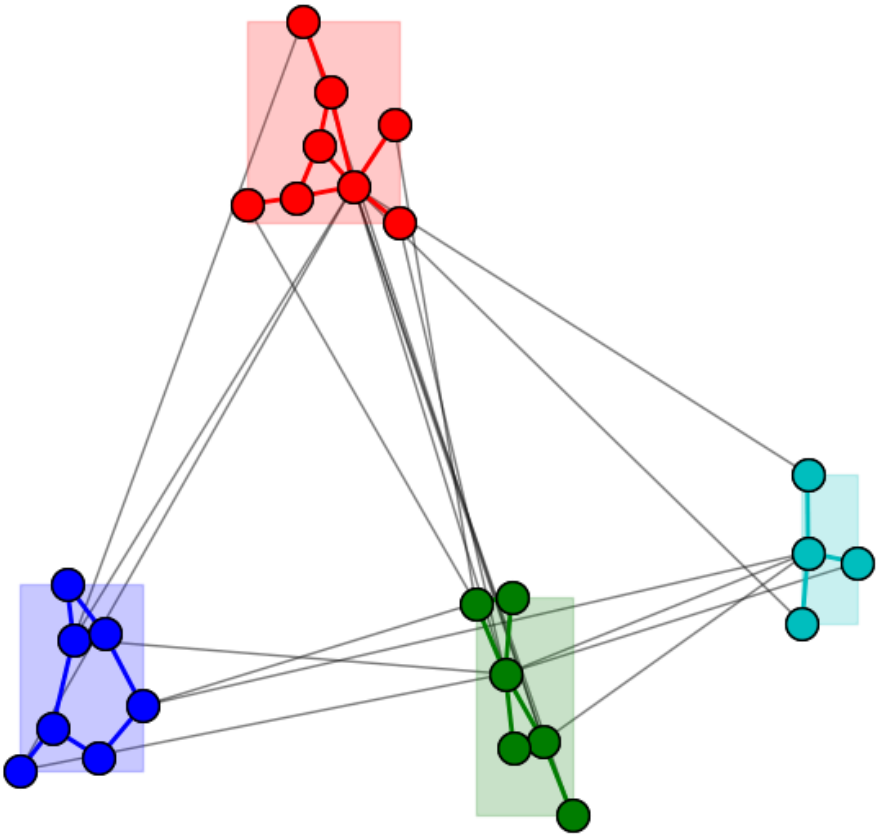

Figure 5 : Démonstration de la fonction plot_network_highlighted_clusters.

Les récentes fonctionnalités ajoutées dû à ma contribution élargissent les capacités de représentation visuelle des structures communautaires des réseaux. Ces ajouts reflètent un engagement continu envers l'amélioration de CDlib et la volonté de répondre aux besoins évolutifs de la communauté de la recherche en réseaux. Il est prévu que ces nouvelles fonctionnalités fournissent des outils plus complets pour explorer et interpréter les réseaux.

5. Analyses et Résultats

Dans le cadre de mon projet, les réunions collaboratives avec Rémy Cazabet ont joué un rôle central dans l'analyse et l'évaluation des besoins en matière de visualisation des graphes. Ces séances de travail ont fourni un cadre propice à l'exploration des idées et des exigences spécifiques. À travers des échanges fructueux, la viabilité et la faisabilité de diverses propositions ont été examinées, telles que l'intégration de nouveaux paramètres au sein des fonctions ***plot_network_clusters*** et

plot_community_graph ou encore l'intégration des fonctions ***calculate_cluster_edge_weights***, ***calculate_cluster_sizes*** et ***plot_network_highlighted_clusters*** pour répondre à de nouveaux besoins fonctionnels. Malgré l'enthousiasme pour certaines idées novatrices, la contrainte temporelle s'est parfois avérée un obstacle, entraînant l'abandon de certaines initiatives en raison de la complexité des dépendances logicielles et des contraintes de temps.

Parmi les idées explorées, figurent notamment la création de graphes interactifs permettant d'explorer le contenu des communautés de manière plus approfondie. De plus, le contrôle des positions des nœuds afin de les placer de manière précise sur une carte géographique a été envisagé, mais sa mise en œuvre s'est avérée complexe dans les délais impartis. Ces exemples illustrent les défis rencontrés lors de l'exploration de nouvelles fonctionnalités et mettent en lumière l'importance de trouver un équilibre entre l'innovation et la réalité des contraintes techniques et temporelles.

En parallèle, une part importante de l'analyse a porté sur la réponse aux questions et aux demandes des utilisateurs de CDlib. En s'appuyant sur les retours de la communauté, les attentes des utilisateurs ont été identifiées et des solutions adaptées ont été proposées, telles que la possibilité d'afficher les *top_n* communautés d'un graphe.

Figure 6 : Exemple du issue Closed durant ma contribution.

Les résultats de cette phase de collaboration se sont traduits par la validation des propositions par l'administrateur du projet Github CDlib, marquant ainsi l'intégration de mes nouvelles fonctionnalités au sein de la librairie.

Les résultats de ce travail sont désormais visibles sur la plateforme CDlib (https://github.com/GiulioRossetti/cdlib), où les changements ont été intégrés via des pull-requests présentés et validés (ou encore en attente). Ce processus a permis d'assurer l'intégration harmonieuse

des nouvelles fonctionnalités au sein de l'écosystème CDlib, conservant ainsi sa robustesse et sa pertinence.

De plus, une partie de mon travail a été de réécrire la documentation de la visualisation. J'ai conclu que les noms des fonctions étaient trop ambigus, ce qui constituait un frein dans ma compréhension de la librairie au début du projet. J'ai donc décidé de revoir et de clarifier la documentation des fonctions existantes, ainsi que de compléter celle des fonctions que j'ai créées. Elle a été automatiquement mise à jour après un Pull-Request et est visible sur https://cdlib.readthedocs.io/en/latest/reference/viz.html, offrant aux utilisateurs une référence complète pour explorer et utiliser les nouvelles fonctionnalités de visualisation.

6. Conclusion

La contribution à la librairie CDlib a abouti à des améliorations significatives dans sa capacité à visualiser les communautés dans les réseaux complexes. L'objectif initial était d'enrichir CDlib en introduisant de nouvelles fonctionnalités de visualisation, en répondant aux besoins de la communauté des utilisateurs et en améliorant la documentation. Dans cette optique, ma contribution a été déployé sur le Git de CDlib, tels que ***plot_network_clusters***, ***plot_community_graph***, et ***plot_network_highlighted_clusters***, qui offrent une représentation claire et détaillée des structures communautaires des réseaux, ainsi que la réponse aux issues sur GitHub, ont renforcé l'utilité de la librairie.

L'évaluation de l'atteinte des objectifs initiaux révèle un succès satisfaisant dans la mise en œuvre des fonctionnalités que j'ai proposé. Les nouveaux outils de visualisation offrent aux utilisateurs des moyens plus efficaces pour explorer et interpréter les communautés détectées. En fin de compte, la contribution à la librairie CDlib représente une avancée dans la facilitation de la visualisation des réseaux complexes, offrant ainsi de nouvelles perspectives pour la recherche et l'analyse dans ce domaine.

7. Perspectives d'avenir

Pour les futures contributions à CDlib, plusieurs pistes d'amélioration et de développement s'ouvrent. L'intégration de fonctionnalités permettant la visualisation des graphes sur une carte géographique constitue une voie prometteuse pour mieux comprendre les structures spatiales des réseaux dans divers contextes, tels que les réseaux sociaux géolocalisés ou les réseaux de transport. De plus, la création d'outils interactifs permettant aux utilisateurs d'explorer les communautés en cliquant sur les nœuds pour accéder à des sous-graphes intra-communautaires offrirait une expérience plus immersive et intuitive.

8. Références

I. *NetworkX. (s.d.). NetworkX. Récupéré sur Introduction — NetworkX 3.2.1 documentation*
   a. *Documentation sur les types de visualisation des graphes: Gallery — NetworkX 3.2.1 documentation*

II. *CDlib Documentation. (s.d.). CDlib Documentation. Récupéré sur https://cdlib.readthedocs.io/en/latest/index.html*

a. *Documentation des fonctions de visualisation: https://cdlib.readthedocs.io/en/latest/reference/viz.html*
b. *Références de CDlib comprenant les algorithmes de détection de communautés: https://cdlib.readthedocs.io/en/latest/reference/reference.html*