Détection de comportements et identification de rôles dans les réseaux sociaux

Jonathan Debure

AIRBUS le cnam

ÉCOLE DOCTORALE EDITE de Paris
Laboratoire CEDRIC

THÈSE DE DOCTORAT

présentée par : **Jonathan DEBURE**
soutenue le : **5 Juillet 2021**

pour obtenir le grade de : **Docteur du Conservatoire National des Arts et Métiers**

Spécialité : **Informatique**

Détection de comportements et identification de rôles dans les réseaux sociaux

THÈSE dirigée par
M DU MOUZA Cédric — *Maître de conférences, HDR, CNAM Paris*

et co-encadrée par
Mme CONSTANTIN Camélia — *Maître de conférences, Sorbonne Université*

RAPPORTEURS
Mme CHIKY Raja — *Professeure, ISEP Paris*
Mme BOUZEGHOUB Amel — *Professeure, Telecom SudParis*

PRÉSIDENT DU JURY
M AMANN Bernd — *Professeur des universités, Sorbonne Université*

EXAMINATEURS
M TRAVERS Nicolas — *Professeur, École supérieure d'ingénieurs Léonard-de-Vinci*

MEMBRES INVITÉS
M Brunessaux Stéphan — *Sénior Expert, Airbus*

Notice

À Augustin,

Remerciements

Je voudrais tout d'abord remercier mon directeur de thèse, Cédric Du Mouza, qui m'a encadré tout au long de cette thèse. Il a su me partager son goût pour la recherche. Sa gentillesse et ses brillantes intuitions ont permis d'élaborer ces travaux dans une atmosphère amicale.

Je remercie Camélia Constantin, qui nous a accompagnés, conseillés et souvent corrigés tout au long de ces trois années.

Je tiens à remercier Airbus et l'ANRT d'avoir pu financer ces travaux et, surtout, Stéphan Brunessaux qui a suivi les travaux avec rigueur, a su me motiver dans les moments de doutes et qui m'a poussé jusqu'au bout.

Je remercie Nicolas Travers et Bernd Amann d'avoir effectué le suivi des travaux.

Je tiens à remercier ceux sans qui tout cela n'aurait pas était possible : mes parents, qui m'ont toujours poussé dans les études. A mon frère et à ma soeur, Céline Boudet (qui a toujours souhaité que son nom apparaisse dans une thèse), merci de m'avoir supporté.

Merci à ma femme, pour toutes les corrections et le support moral. Merci à mon fils pour ses câlins.

Il m'est impossible d'oublier mes collègues, avec qui j'ai partagé de l'expérience, des bons moments, des cafés et des bières. Un merci tout particulier à Antoine, Nicolas, Paul et Axelle. Merci aux jeunes chercheurs, Killian, Guillaume et Jacques. Merci à mon ancienne équipe, Franck, Sylvie, Yann, Laurent, Samy, Florentin, Kadiatou, Adelino, Leïla, Khaled, Sylvain, Florian, Souhir, Bruno, Valérian, Vincent C., Vincent P., Fabien. Et merci aussi à ma nouvelle équipe, François, Bénédicte, Olivier, Jean-Baptiste et Bernard.

Résumé

Les réseaux sociaux sont devenus des outils de communication primordiaux et sont utilisés quotidiennement par des centaines de millions d'utilisateurs. Tous ces utilisateurs n'ont pas le même comportement sur ces réseaux. Si certains ont une faible activité, publient rarement des messages et suivent peu d'utilisateurs, d'autres, à l'opposé, ont une activité importante, avec de nombreux abonnés et publient très régulièrement. Le rôle important de ces utilisateurs influents en font des cibles intéressantes pour de nombreuses applications, comme pour la surveillance ou la publicité. Après une étude des méta-données de ces utilisateurs, afin de détecter des comptes anormaux, nous présentons une approche permettant de détecter des utilisateurs devenant populaires. Notre approche s'appuie sur une modélisation de l'évolution de la popularité sous la forme de motifs fréquents. Ces motifs décrivent les comportements de gain en popularité. Nous proposons un modèle de matching des motifs permettant d'être utilisé avec un flux de données et, nous montrons sa capacité à passer à l'échelle en le comparant à des modèles classiques. Enfin, nous présentons une approche de clustering basé sur le PageRank. Ces travaux permettent d'identifier des groupes d'utilisateurs partageant le même rôle, en utilisant les graphes d'interactions qu'ils génèrent.

Mots-clés : réseaux sociaux, clustering , motifs fréquents , comportements, Twitter, PageRank, détection de la popularité.

Abstract

Social networks (SN) are omnipresent in our lives today. Not all users have the same behavior on these networks. If some have a low activity, rarely posting messages and following few users, some others at the other extreme have a significant activity, with many followers and regularly posts. The important role of these popular SN users makes them the target of many applications for example for content monitoring or advertising. After a study of the metadata of these users, in order to detect abnormal accounts, we present an approach allowing to detect users who are becoming popular. Our approach is based on modeling the evolution of popularity in the form of frequent patterns. These patterns describe the behaviors of gaining popularity. We propose a pattern matching model which can be used with a data stream and we show its scalability and its performance by comparing it to classic models. Finally, we present a clustering approach based on PageRank. This work allow to identify groups of users sharing the same role, using the interaction graphs.

Keywords : social network, clustering, patterns mining, behaviours, Twitter, PageRank, popularity detection.

Table des matières

Liste des tableaux

Table des figures

Chapitre 1

Introduction

Contenu

1.1 Contexte : les réseaux sociaux en ligne

Au cours des vingt dernières années, les technologies de communication et d'interactions sociales ont évolué de façon exponentielle. L'avènement des médias sociaux étant une avancée clé. Le réseautage social est une plateforme en ligne permettant aux individus de créer des liens sociaux. Ces derniers créent des relations avec d'autres personnes partageant des intérêts personnels ou professionnels communs, des passe-temps, des antécédents ou des liens dans le monde réel. Les hommes vivent à l'ère de l'information. Ils sont entourés d'appareils électroniques et de plateformes de réseautage social immersives comme Twitter, Instagram, Facebook ainsi que YouTube. Ces médias sociaux sont devenus des éléments centraux de la vie de chacun, notamment chez les jeunes. Les sites web de réseautage social peuvent permettre à un individu de développer ses compétences sociales.

Les médias sociaux sont un concept qui englobe les relations entre les individus et les organisations dans lesquelles ils créent, communiquent et partagent. Il peut s'agir de pensées, d'idées, de photographies, de vidéos, etc. Les médias sociaux sont un moyen de communication ayant lieu sur Internet. Les utilisateurs des sites de médias sociaux peuvent participer à des discussions, échanger des informations et créer du contenu web. Les médias sociaux existent de diverses manières. Il peut s'agir de blogs, de micro-blogs, de wikis, de sites web pour les réseaux sociaux, de sites de partage de photographies, de messageries instantanées, de sites de partage de vidéos, de podcasts, de mondes virtuels, etc. Le but étant d'échanger des connaissances et de tisser des liens à travers le monde entier. Les médias sociaux nous permettent de nous connecter avec nos amis et notre famille, d'acquérir de nouvelles compétences, d'indiquer nos préférences et d'être fasciné au niveau personnel. En ce qui concerne le milieu professionnel, LinkedIn remplit différentes fonctions telles que la mise en relation avec d'autres professionnels de notre industrie. Nous pouvons utiliser les médias sociaux pour élargir ou augmenter nos connaissances dans un domaine spécifique et développer notre réseau professionnel. De plus, les médias sociaux sont utilisés par les entreprises pour faire du marketing. Les médias sociaux aident les entreprises à parler à leur public, à recevoir des commentaires de leurs clients et à promouvoir leur marque [4]. La publicité dans les médias sociaux est de plus en plus présente et est souvent personnalisée pour chaque utilisateur [5].

Les réseaux sociaux diffèrent dans les tendances et les cadres émergents [6]. Les médias sociaux les plus importants sur le web sont Facebook, WhatsApp, YouTube et Twitter. Tous ces sites ont un impact important sur la société. Facebook a un nombre d'utilisateurs équivalent à un tiers de la population mondiale [1]. Plus les réseaux sociaux améliorent les expériences utilisateurs, plus ils deviennent importants. Les individus confrontés à diverses interrogations peuvent trouver des réponses facilement dans des groupes. Par exemple, Facebook propose un groupe pour les personnes diabétiques dans le but de partager ses expériences personnelles, de poser des questions et d'obtenir des réponses rapidement. Ces groupes peuvent également être utilisés pour vendre des biens et des services. Les réseaux sociaux tels que Facebook, Twitter et WhatsApp sont des exemples de plateformes de communication instantanées en ligne [7]. Les réseaux sociaux et les médias servent à la société de différentes manières : les achats en ligne, la publicité et diverses informations sur des sujets très nombreux. Ainsi, les étudiants profitent grandement des réseaux sociaux, principalement en s'aidant les uns les autres sur des devoirs scolaires et des projets conjoints en dehors de la classe [8]. Les médias sociaux ont un effet énorme sur notre société. Les particuliers et les entreprises peuvent communiquer, créer et entretenir des relations sociales et commerciales grâce à l'échange d'idées et d'opinions via une variété de réseaux de communication en ligne, y compris la communication interpersonnelle (par exemple, Twitter, Facebook), les communautés de contenu (par exemple, Wikipedia) et les plates-formes multimédia (par exemple, YouTube, Instagram)

1.2 Notre Problématique

La plupart des travaux existant proposent des techniques de détection d'utilisateurs déjà populaires ou influents sur les réseaux sociaux. Cependant, les différents exemples d'applications présentés ci-dessus montrent qu'il est important de pouvoir identifier, au plus vite, l'apparition d'utilisateurs populaires sur les réseaux sociaux. Que ce soit dans le domaine du marketing, de la politique ou de la défense.

Nos objectifs sont :

1 **Analyser l'évolution des utilisateurs** : Il s'agit d'étudier les comportements des utilisateurs pour en extraire des informations permettant d'identifier des comptes influents.

2 **Étudier le comportement des utilisateurs** : L'étude du comportement dans les réseaux sociaux

1. https ://blog.digimind.com/fr/agences/facebook-chiffres-essentiels#entreprisesFR

peut nous permettre de comprendre et d'identifier les différents rôles qu'ont les utilisateurs.

1.3 Contributions

Nos travaux sont, à notre connaissance, les premiers à tenter d'identifier les utilisateurs qui sont sur le point de devenir populaires. En détectant des tendances récurrentes dans l'évolution de la popularité des comptes devenant populaires, nous arrivons, avec une bonne précision, à détecter les utilisateurs plusieurs semaines avant qu'ils ne deviennent vraiment populaires. De plus, la structure d'index que nous proposons permet de passer à des centaines de millions d'utilisateurs et donc, de déployer notre solution sur de véritables plateformes de réseaux sociaux. Nous proposons aussi une approche de clustering permettant d'identifier différents rôles d'utilisateurs, basée sur les interactions que ces derniers peuvent générer.

1.4 Organisation de la thèse

Après cette introduction, nous présenterons Twitter (chapitre 2), le réseau social en ligne qui nous a servi de support pour les travaux de cette thèse. Après cela, nous exposerons un rapide état de l'art (chapitre 3) des recherches effectuées sur les réseaux sociaux.
Dans le chapitre 4, nous présenterons une étude des méta-données d'utilisateurs pour détecter des comptes divergents ou anormaux. Cette étude permettra d'introduire nos contributions.
Dans le chapitre 5, nous présenterons une méthode pour détecter, en avance, des utilisateurs populaires. Après une analyse de l'évolution de la popularité chez les utilisateurs populaires et devenant populaires, nous utiliserons une approche basée sur l'extraction et l'identification de motifs fréquents. Enfin, pour permettre l'utilisation de ce modèle directement sur des réseaux sociaux avec plusieurs centaines de millions d'utilisateurs, nous présenterons un modèle de matching de motifs pouvant être utilisé avec d'importants flux de données.
Dans le chapitre 6, nous utiliserons un score de PageRank pondéré calculé sur les graphes d'interactions entre les utilisateurs. Ce dernier, nous permet de regrouper les utilisateurs avec des comportements similaires.
Enfin, le chapitre 7 conclut ce document en rappelant les contributions et présente les différentes perspectives de recherche.

Chapitre 2

Présentation de Twitter

Contenu

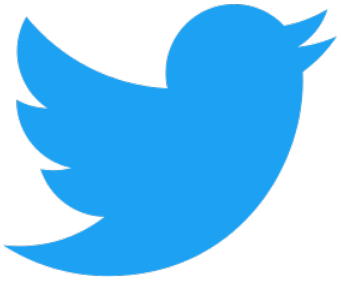

FIGURE 2.1 – Logo de Twitter

Afin de comprendre les différents termes utilisés dans cette thèse, nous pensons qu'il est nécessaire de présenter rapidement Twitter. Ce réseau social en ligne qui nous a permis de construire plusieurs jeux de données sur lesquels nos travaux se sont appuyés.

2.1 Introduction

Twitter est un réseau social en ligne créé en 2006 et facilement identifiable pas son logo bleu (figure 2.1). Il propose aux utilisateurs de partager du contenu "tweet" sous différentes formes (texte, images, vidéos). Ces messages seront affichés sous la forme d'un fil d'actualité souvent ordonné par date. Par défaut, tous les messages postés sur ce site sont publiques et n'importe qui peut les lire. Une option permet de privatiser ses messages et d'autoriser seulement ses abonnés à les lire. Cette plateforme possède 192 millions d'utilisateurs actifs au quotidien et il y a environ 500 millions de messages [1] postés tous les jours. Une des particularités de Twitter est qu'il ne propose pas aux utilisateurs de créer des groupes ou des pages, ce qu'on peut retrouver sur Facebook, par exemple. Sur Twitter, que l'on soit une personne, une entreprise ou n'importe quelle autre entité, le compte est toujours le même. En revanche, il propose un service de certification permettant aux personnalités ou entreprises de faire certifier leurs comptes et, de ce fait, informe les utilisateurs qu'il ne s'agit pas d'un faux compte.
Les utilisateurs de Twitter peuvent interagir avec les contenus ou les autres utilisateurs. Dans la section suivante nous allons lister et expliquer chaque action.

1. `https://www.oberlo.com/blog/twitter-statistics`

2.2 Les fonctionnalités

Sur Twitter, il y a deux groupes d'actions : les actions qui peuvent être effectuées sur les comptes et les actions qui peuvent être effectuées sur les messages.

Actions sur les comptes :

- **L'abonnement** : cette action permet aux utilisateurs de suivre le contenu publié par un autre utilisateur. Avec cette fonctionnalité d'abonnement, il existe deux termes désignant le sens de l'abonnement : Followers, qui désigne les personnes qui suivent un autre compte et Following qui, à l'inverse, renseigne tous les utilisateurs que le compte suit.
- **Listing** : cette action permet de mettre en avant certains utilisateurs sur son fil d'actualité en l'ajoutant à une liste de suivi.

Actions sur les messages :

- **Retweet** : c'est l'action de partage d'un message. Il sera partagé sans modification sur le fil d'actualités de l'utilisateur et sera visible par ses followers.
- **Quote** : le principe est identique au retweet, mais la particularité est, que l'utilisateur peut y attacher un message. Il y aura donc un message suivi du tweet partagé.
- **Mention** : consiste à identifier un autre compte à l'intérieur de son message. Cette action enverra une notification à l'utilisateur mentionné.
- **Like** : signale, sous la forme d'un compteur, le nombre de personnes qui ont apprécié le tweet.
- **Hashtag** : le hashtag est un mot-clé commençant par '#' et, qui souvent, fait référence à un sujet ou à une réaction à propos du message. Les hashtags sont cliquables et permettent d'afficher la liste des messages ordonnés qui utilisent les mêmes mots-clés.
- **Reply** : cette action permet de répondre à un message, la réponse est attachée au tweet et peut être lue par tout le monde.

Les différentes fonctionnalités de Twitter permettent facilement de créer des graphes d'utilisateurs. De plus,il existe une fonctionnalité qui le rend attrayant pour les recherches. Celle-ci met à disposition une API permettant de récupérer des messages que ce soit en effectuant une recherche par mots-clés ou hashtags mais aussi, en permettant de récupérer directement les messages publiés sur la plateforme.

Cette fonctionnalité qui permet d'obtenir gratuitement 1% du flux de données, permet d'accéder aux messages, aux méta-données des messages, aux auteurs et à leurs méta-données ainsi qu'aux acteurs (les utilisateurs effectuant une action). Le nombre de messages reçus peut être augmenté en souscrivant à un contrat payant permettant d'accéder au *decahose* (10% des tweets) ou *firehose* (100% des tweets). Mais cela est très onéreux. Dans le chapitre suivant, nous allons présenter les différents travaux et méthodes utilisées sur les graphes.

Chapitre 3

État de l'art

Contenu

L'analyse de l'influence, la détection de communautés et l'identification de rôles dans les réseaux sociaux sont des sujets de recherche qui, depuis longtemps, intéressent. Bien avant l'arrivée d'internet, l'étude mathématique des réseaux sociaux avait une grande importance. La plupart des algorithmes utilisés aujourd'hui proviennent de recherches effectuées depuis plus de vingt ans.

Ce chapitre a pour objectif de faire l'état de l'art des différents travaux effectués sur les réseaux sociaux. Notamment dans deux domaines : l'identification d'influenceurs et la détection de communautés.

Dans la première section, nous présentons les notions d'influence et de popularité. Nous poursuivons en présentant les différents profils d'utilisateurs présents sur les réseaux sociaux et, nous présentons les différentes mesures et scores mathématiques utilisés dans la plupart des travaux de recherche. Dans la deuxième section, nous présentons les différentes techniques de détection de communautés. Enfin, nous synthétisons ce chapitre.

3.1 L'étude des influenceurs

Identifier les utilisateurs influents au sein d'un réseau social reste un défi colossal que ce soit pour le domaine du marketing (lorsque l'on cherche à promouvoir une marque ou un produit) ou que ce soit pour le domaine de la défense (lorsqu'il faut identifier des acteurs clés).

Dans cette section nous allons définir la notion d'influenceur et présenter les différentes méthodes mises au point pour tenter d'identifier ces utilisateurs.

3.1.1 Définitions d'influence et de la popularité

3.1.1.1 L'influence

L'influence, par sa définition, est le fait d'exercer une action sur quelqu'un ou quelque chose. Dans un cas social, où l'influence s'exerce entre deux individus, l'influence est le fait de faire adopter un point de vue à une autre personne. Dans le cas des réseaux sociaux en ligne, nous avons proposé de définir un influenceur comme :

Définition 1 (Influenceur) *Un influenceur est un individu qui, par ses actions ou absence d'actions (comportements, discours), a la possibilité de modifier le comportement ou le discours d'un autre individu.*

Il ne faut pas confondre l'influence et la popularité que nous allons définir dans le point suivant.

3.1.1.2 La popularité

La popularité est le fait d'être connu par une grande partie de la population, mais être connu n'est pas forcément synonyme d'influent. Dans le cas des réseaux sociaux, une personne populaire est un individu qui a une grande visibilité au sein du réseau. Nous avons proposé cette définition de popularité :

Définition 2 (Popularité) *La popularité d'un individu correspond à sa visibilité, cela se traduit par le nombre de personnes qui lisent, commentent ou propagent les messages écrits par cet individu.*

Dans nos travaux nous avons simplement défini que la popularité d'un compte sur un réseau social était le nombre d'abonnements (followers) que l'utilisateur possède à un instant t.

3.1.2 Profils d'utilisateurs

Les réseaux sociaux en ligne sont devenus des communautés omniprésentes pour avoir des interactions sociales, ce qui a, par conséquent, des impacts sur notre quotidien [9]. Le comportement des utilisateurs sur ces plateformes est un domaine d'étude dans lequel nous avons peu de connaissances. De simples questions sur le comportement de navigation des utilisateurs restent sans réponse malgré la croissance rapide et la pertinence des technologies numériques [10]. Outre l'utilisation croissante des médias numériques, le budget alloué aux publicités en ligne augmente de manière continue. Cependant, nous constatons que, malgré l'augmentation de ces dépenses en marketing numérique, leur efficacité diminue. Les sociétés de marketing ont donc commencé à chercher des méthodes alternatives pour influencer les potentiels clients dans ce marché moderne où la commercialisation des biens et des services n'a jamais été aussi difficile [11]. Ils se servent donc des utilisateurs eux-mêmes pour promouvoir leurs produits. Mais ces derniers ont des rôles bien différents sur les médias sociaux en ligne, nous allons donc, dans cette section, aborder ces différents comportements.

3.1.2.1 L'influenceur

Les influenceurs, sur les médias sociaux, sont des individus ayant acquis une certaine notoriété grâce à leur perspicacité et leur expérience, le plus souvent sur un sujet particulier (cuisine, mode, sciences, etc.). Ils publient régulièrement des articles à ce propos sur leurs plateformes de médias sociaux de prédilection. Cela leur permet d'être suivis par un grand nombre d'utilisateurs passionnés et engagés qui perçoivent leurs opinions et y réagissent. Il existe différents types d'influenceurs. Selon le nombre d'abonnés, le comportement de ces acteurs clés change : les petits influenceurs (ceux avec un nombre d'abonnés faible) ont des taux d'interaction plus élevés que les grands influenceurs. En effet, ayant moins d'abonnés, il leur est plus facile de communiquer directement avec eux. La société Influencer Marketing Hub [12] propose une classification des catégories d'influenceurs à partir du nombre moyen d'abonnés :

- **Le Micro-Influenceur** [1], qui a moins de 15 000 abonnés.
- **L'influenceur du quotidien**, qui possède entre 15 000 et 50 000 abonnés.
- **L'influenceur en devenir**, ce type d'utilisateur possède entre 50 000 et 100 000 abonnés.
- **L'influenceur moyen**, qui a entre 100 000 et 500 000 abonnés.
- **Le Macro-Influenceur**, souvent identifié comme des célébrités locales entre 500 000 et 1 000 000 d'abonnés.
- **Le Mega-Influenceur**, le plus souvent, des célébrités internationales possédant plus d' un million d'abonnés.

Ces différentes catégories peuvent être divisées en plusieurs types d'influenceurs. Selon Gross et al. [13] il y aurait quatre types d'influenceurs :

- **Snoopers** : les influenceurs débutants. Ils sont décrits comme des personnes découvrant les réseaux sociaux, ils sont créatifs et ils aiment partager leurs expériences.
- **Informers** : ce sont des experts. Ils sont souvent influents dans un domaine ou un sujet précis. Ils utilisent les réseaux sociaux pour partager leurs connaissances.
- **Entertainers** : ces influenceurs fournissent du divertissement souvent associé à la musique, le cinéma ou les jeux vidéos. Ils sont là pour profiter et amuser leurs followers.

1. `https://business.twitter.com/en/blog/micro-influencers-and-where-to-find-them.html`

FIGURE 3.1 – Squeezie le YouTubeur le plus suivi en France avec 14.7 Millions d'abonnés.

— **Infotainers** : ce sont principalement des journalistes. Ce type d'utilisateur va favoriser les réseaux sociaux pour diffuser des informations, que ce soit sous la forme d'articles ou de vidéos.

Les figures suivantes permettent de se faire une représentation des différents types et catégories d'influenceurs. Sur la figure 3.1 nous avons l'exemple d'un Mega-influenceur que nous pouvons classer dans Entertainers. Sur la figure 3.2 nous avons un Macro-Influenceur que nous pouvons associer à un Informer et, sur la figure 3.3 nous avons un influenceur moyen qui est classé dans les Infotainers.

3.1.2.2 Le faux influenceur

On appelle faux influenceurs, les utilisateurs qui ont recours à l'achat d'abonnés. La plupart de ces abonnés sont souvent des spammeurs (personnes ou robots, aussi appelés bots, utilisant de faux comptes qui publient automatiquement des messages, des commentaires souvent très répétitifs et ayant peu d'intérêt) ou des abonnés totalement inactifs. Les faux influenceurs sont le cauchemar des annonceurs publicitaires. Très répandus et difficiles à reconnaître au premier abord, les faux influenceurs continuent de tromper les annonceurs et leur coûtent des millions en dépenses inutiles. Mais si le fait d'acheter des followers semble être une méthode coûteuse et complexe, cela est moins cher et plus simple que nous pouvons le penser. Sur le marché de la vente de followers, le site *SupremeBoost* permet à quiconque d'acheter 5 000 d'abonnés Instagram pour un peu moins de 50 € en un seul clic. Les faux influenceurs sur les médias sociaux sont très doués pour créer des profils convaincants autour de sujets

FIGURE 3.2 – Thomas Pesquet Ingénieur et astronaute français partageant de nombreux clichés de son expérience dans l'espace.

FIGURE 3.3 – Martin Weill, reporter français partageant des informations et des vidéos d'interviews

d'intérêts. Ils utilisent des hashtags dans leurs descriptions et leurs messages pour être plus facilement référencés et publient des photos et images de haute qualité qui abordent des sujets tels que la mode, la beauté ou les voyages.

3.1.2.3 L'utilisateur normal

Les utilisateurs normaux représentent la majorité des utilisateurs des réseaux sociaux en ligne. Ceux-ci utilisent ces plateformes pour garder contact avec leurs proches, se faire des amis et interagir avec les contenus publiés. Ils interagissent de différentes façons, créent des relations en s'abonnant, supprimant ou bloquant d'autres utilisateurs. Mais aussi en communiquant par messages, entre utilisateurs, en publiant des messages dans un groupe communautaire ou en partageant des messages qu'il pourra lire sur le réseau.

3.1.2.4 L'utilisateur inactif

Il existe deux types d'utilisateurs inactifs :

— Les utilisateurs totalement inactifs, déconnectés. Ils n'utilisent plus la plateforme à laquelle ils sont inscrits mais leurs comptes sont toujours présents.

— Les rôdeurs. Ces utilisateurs voient et lisent le contenu sans interagir ou publier. Ce sont des utilisateurs silencieux qui représentent généralement une part importante des utilisateurs de plateformes de médias sociaux. Étant donné qu'ils apportent une contribution mineure au contenu en ligne, les analyses actuelles ignorent fréquemment leur participation et leur voix. Les rôdeurs ne contribuent pratiquement à aucun contenu mais ont tendance à en consommer. Dans le domaine de la recherche, il est important de reconnaître les rôdeurs en étudiant leurs caractéristiques, leurs préférences et leurs perspectives. Les rôdeurs (comme les utilisateurs actifs) sont des individus qui ont des désirs et des attentes concernant leur navigation silencieuse en ligne. Ils prêtent attention aux sujets qui les préoccupent ou recherchent des informations pertinentes concernant une futur acquisition. Ils ont des goûts qui peuvent être transmis sous la forme de commentaires et de recommandations sur des biens de consommation. Ils sont également des clients potentiels pour le marketing ciblé [14].

3.1.2.5 Le bot

Les bots ou robots sont des programmes automatiques qui utilisent les réseaux sociaux selon des instructions, sans qu'un utilisateur humain ait besoin d'interagir manuellement. Les bots imitent ou remplacent également les actions d'un utilisateur humain. Ils effectuent généralement des tâches répétées et peuvent les accomplir beaucoup plus rapidement que des utilisateurs normaux. C'est généralement sur ce type de comportements que les travaux d'identification de bots reposent [15]. Les bots s'améliorent de jour en jour. Lorsque l'on regarde les avancées techniques de Siri et Cortana, il ne serait pas étonnant de voir apparaître, si ce n'est pas déjà le cas, ce même type de programme sur les réseaux sociaux. Les bots ont plusieurs objectifs : rassembler et agréger des actualités ou bien transmettre des actualités de différentes sources. Ces types de bots sont souvent utilisés pour transmettre des "Fake News" basés sur des éléments réels de l'actualité. Les bots sont aussi utilisés à des fins marketing : en publiant des annonces publicitaires, par exemple, mais aussi pendant les campagnes électorales afin de manipuler l'opinion publique ou le sentiment des médias sociaux [16]. Les bots sont connus pour être le type d'utilisateurs malveillants le plus courant sur les plateformes sociales. Ils peuvent générer de fausses actualités (*Fake News*), diffuser des rumeurs et influencer les opinions publiques [17]. Les bots peuvent aussi servir à créer de faux abonnés et se mettre à suivre un utilisateur qui aurait payé pour avoir plus d'abonnés.

3.1.3 Les mesures d'influences

Les réseaux sociaux peuvent être modélisés mathématiquement sous la forme de graphes, dans lesquels les noeuds représentent les utilisateurs et, les arêtes, les liens entre chaque utilisateur. L'intérêt de cette modélisation est, qu'elle rend possible l'application d'algorithmes. Dans cette section nous allons aborder trois types d'approches : les mesures de centralité, les scores et les modèles de diffusion.

3.1.4 Les mesures de centralité

La centralité est définie comme une mesure d'importance d'un noeud au sein d'un graphe. On peut trouver différents types de mesures de centralité [18, 19], la mesure de puissance [20, 21] et celle de prestige [22]. Dans la suite de cette section, nous allons nous intéresser à trois variations de mesures de centralité qui sont : le degré de centralité, la centralité de proximité et la centralité d'intermédiarité.

3.1.4.1 Le degré de centralité

Le degré de centralité [18] est une mesure simple qui consiste à compter le nombre d'arêtes d'un noeud. Elle se calcule avec l'équation suivante :

$$C_D(i) = \sum_{j=1, j\neq i}^{n} a_{ij} \tag{3.1}$$

Où i est le noeud sélectionné, n l'ensemble des noeuds du graphe et a_{ij} est la valeur booléenne de la matrice d'adjacence (1 s'il y a un lien entre i et j, 0 sinon).

3.1.4.2 La centralité de proximité

La centralité de proximité [23, 24] d'un noeud est la mesure de la distance de ce noeud aux autres noeuds. Plus le chemin entre chaque noeud est court, plus son score de proximité sera élevé. Pour calculer la centralité de proximité d'un noeud, il suffit de calculer la somme des distances de ce noeud aux autres noeuds du graphe en utilisant un calcul de plus court chemin.

$$C_C(i) = \frac{1}{\sum_{j-i}^{n} d(i,j)} \tag{3.2}$$

Où $d(i,j)$ est le nombre minimum de noeuds intermédiaires pour rejoindre i à partir de j.

3.1.4.3 La centralité d'intermédiarité

La centralité d'intermédiarité [25] a pour but d'identifier les noeuds "pont" ou de passage, c'est-à-dire, les noeuds par lesquels passent le plus de chemins entre toutes les paires de noeuds d'un graphe. Plus un noeud sera traversé, plus son score d'intermédiarité sera élevé.

$$C_B(i) = \frac{\sum_{i\neq j\neq k} d_{(j,k)}(i)}{d_{(j,k)}} \tag{3.3}$$

Où $d_{(j,k)}(i)$ est le nombre de plus courts chemins entre j et k passant par i et $d_{(j,k)}$ est le nombre total de plus courts chemins entre j et k.

3.1.5 La centralité de Katz

La centralité de Katz [26] est une généralisation de la centralité de degré. Cette méthode mesure le nombre de voisins (le degré) et le nombre de noeuds pouvant être atteints en parcourant le graphe

de relations ou Ego Network (qui est un graphe centré sur un noeud).

$$C_K(i) = \sum_{k=1}^{\infty} \sum_{j=1}^{N} \alpha^k (A^k)_{ji} \tag{3.4}$$

Où A est la matrice d'adjacence et $(A^k)_{ji}$ est le nombre de noeuds entre j et i. α est la valeur d'atténuation qui augmente selon la distance des noeuds.

3.1.6 La centralité spectrale

La centralité spectrale [27] d'un noeud est la combinaison linéaire de la centralité de ses noeuds voisins. Le principe de cet algorithme a été repris par l'algorithme de classement PageRank.

$$C_S(i) = \frac{1}{\lambda} \sum_{j \in N(i)} C_S(j) = \frac{1}{\lambda} \sum_{j \in G} a_{i,j} C_S(j) \tag{3.5}$$

Où $N(i)$ est l'ensemble des voisins de i, $a_{i,j}$ est la valeur dans la matrice d'adjacente du lien entre le noeud i et j et λ est la valeur du vecteur propre.

3.1.7 La centralité de percolation

La centralité de percolation de Piraveenan et al. [28] mesure l'impact d'un noeud dans un graphe selon la topologie du graphe. Cet algorithme mesure l'importance du positionnement d'un noeud dans le graphe en analysant son potentiel de diffusion ou de filtrage. Pour cela, cette méthode calcule le nombre de plus courts chemins passant par le noeud sélectionné et considère un état de filtrage indiquant si le noeud sélectionné est filtrant ou non. De plus, cet algorithme prend en compte la dynamicité des graphes et ajoute la dimension de temps.

$$C_P^t(i) = \frac{1}{N-2} \sum_{s \neq i \neq p} \frac{\sigma_{s,p}(i)}{\sigma s, p} \frac{x_s^t}{[\sum x_n^t] - x_i^t} \tag{3.6}$$

Où $\sigma_{s,p}(i)$ est le nombre de plus courts chemins entre le noeud source s et le puits p passant par le noeud i. $\sigma_{s,p}$ est le nombre de plus courts chemins entre s et p. N est le nombre de noeuds filtrants, x_s^t est la fonction retournant 1 si s est filtrant à l'instant t.

3.1.8 Les scores d'influence

L'origine des scores d'influence vient principalement des algorithmes de classement utilisés par les moteurs de recherche dans lesquels on souhaite mettre en avant les documents les plus pertinents.

Pour cela, il faut calculer le nombre d'hyperliens (un élément placé dans une page web qui, lorsqu'il est sélectionné, nous dirige vers une nouvelle page web du même site ou d'un site différent) qui pointent vers cette page et lui attribuer un score par rapport aux autres pages web.

3.1.8.1 PageRank

PageRank [29] est sûrement l'algorithme de classement le plus connu. Cette méthode de classement a été mise au point par Larry Page et Sergey Brin en 1999 [29] et, est l'algorithme associé au moteur de recherche Google. Le score de PageRank est normalisé et propagé d'un noeud vers tous les noeuds auxquels il pointe. Avec ce score, un noeud pourra avoir une valeur élevée s'il est pointé par beaucoup de noeuds ou, s'il est pointé par des noeuds avec un score de PageRank important. Pour parcourir le graphe, PageRank suit le modèle du "Random Surfer" ou internaute aléatoire. Il va donc emprunter aléatoirement les liens sortant de chaque noeud pour visiter d'autres noeuds. Une particularité du PageRank est que la somme des scores du PageRank est égale à 1. On peut donc le voir comme une probabilité d'accéder à un noeud ou à une page.

$$PR(u_i) = (1 - \alpha) + \alpha \sum_{u_j \in In(u_i)} \frac{PR(u_j)}{Out(u_j)} \tag{3.7}$$

Où $In(u_i)$ est l'ensemble des noeuds pour lesquels il existe un lien vers u_i ($c.a.d$, $\{u_j \in \mathcal{U}, (u_i, u_j) \in \mathcal{E}$), $Out(u_j)$ est le dégré sortant de u_j, α est le facteur d'amortissement.

3.1.8.2 TrustRank

TrustRank est une méthode d'identification de SPAM mise au point par Gyongyi et al. [30]. Cette méthode a été mise au point pour identifier, dans un graphe de pages web, les pages de confiance. Elle utilise principalement une version inversée du PageRank pour identifier les pages qui ont été créées dans le simple but de booster d'autres pages. Cela gonfle leurs scores et ces pages sont mises en avant par les moteurs de recherche. Une des particularités de cette méthode est qu'elle fait appel à un oracle. Un oracle est une personne qui valide des éléments choisis par l'algorithme. Ces éléments sont souvent les plus représentatifs d'une classe et pour lesquels l'algorithme a le plus d'incertitudes. Cet algorithme peut donc être utilisé pour classer les pages avec un indice de confiance et il peut être associé avec un score de PageRank pour améliorer les résultats de moteurs de recherche.

3.1.8.3 HITS

HITS (Hyperlink-Induced Topic Search) [31] a pour but d'identifier, à partir d'un score, les pages web les plus importantes sur un sujet. Le calcul de ce score se fait directement lors de la recherche plutôt que d'utiliser un snapshot de la toile. De plus, HITS est basé sur deux scores : un score d'autorité ou d'importance *auth* (qui met en avant les noeuds du graphe les plus pointés) et un score de passage *hubs* (qui a pour but d'attribuer un score aux pages qui pointent vers des pages importantes).

$$auth(i) = \sum_{j \in E_i} hub(j) \tag{3.8}$$

$$hub(i) = \sum_{j \in O_i} auth(j) \tag{3.9}$$

Où E_i est l'ensemble des noeuds pointant vers i et O_i est l'ensemble des noeuds pointés par i.

3.1.8.4 TunkRank

TunkRank [32] est une adaptation du PageRank spécifique à Twitter. Ce score a pour objectif de mesurer l'influence d'un utilisateur de Twitter en estimant le nombre éventuel de lecteurs d'un compte. Ce qui se traduit par la somme des probabilités qu'un follower (abonné) retweet (partage) un message.

$$TR(u_i) = \sum_{u_j \in Followers(u_i)} \frac{1 + p * TR(u_j)}{Following(u_j)} \tag{3.10}$$

Où $Followers(u_i)$ est l'ensemble des utilisateurs abonnés à u_i. $Following(u_j)$ est l'ensemble des utilisateurs auxquels u_j est abonné. p est la probabilité de retweet (partage).

3.1.8.5 TwitterRank

TwitterRank [33] est un autre score d'influence basé sur PageRank et spécifique à Twitter. Il reprend le score de PageRank mais il utilise de sous-graphes d'utilisateurs qui partagent le même sujet de discussion. Il agrège ensuite ces scores pour calculer un score général d'influence d'un utilisateur.

3.1.8.6 Klout Score

Le score Klout [34] est un score d'influence plus complet. il ne se focalise pas seulement sur un seul réseau mais sur les 9 réseaux sociaux majeurs (Twitter, Facebook, LinkedIn, YouTube, Instagram, Google+, Foursquare, Lithium et Wikipedia). Sur chaque réseau social, cette méthode mesure trois scores :

— La portée réelle : les nombres d'abonnés et d'abonnements sans prendre en compte le nombre de bots ou d'utilisateurs inactifs.

— L'amplification : qui réfère à la probabilité de générer des interactions.

— L'impact sur le réseau : qui analyse l'influence des utilisateurs qui interagissent avec le contenu.

En plus de ces trois mesures, cette méthode applique un score de PageRank sur le contenu Wikipedia pour obtenir une mesure d'influence hors-ligne de l'utilisateur. Pour finaliser le score, les mesures effectuées sur les différents réseaux vont être combinées pour donner un score compris entre 0 et 100, où 100 est le score maximal.

3.1.9 Les modèles de diffusion

Les modèles de diffusion ont pour but d'optimiser les chemins de diffusion, *c.a.d* les chemins qu'emprunte l'information pour se propager dans un graphe. Il existe deux grands modèles de diffusion : le modèle linéaire (avec seuil) et le modèle indépendant par cascade. Ils ont tous deux été proposés par Kempe et al.[35].

3.1.9.1 Le modèle linéaire avec seuil

Le principe de ce modèle est d'activer des noeuds par rapport au nombre de noeuds voisins et en fonction d'un seuil. On attribue aléatoirement un seuil θ compris entre $[0, 1]$ à chaque noeud du graphe. Lorsque le nombre de noeuds voisins actifs dépasse ce seuil, alors le noeud courant devient actif lui aussi. C'est un algorithme itératif qui permet d'exprimer l'influence des noeuds d'un graphe. Les auteurs, cités ci-dessus,[35] expliquent que le choix de la valeur de seuil aléatoire est ainsi fait pour exprimer le manque de connaissances sur l'influence que peuvent avoir les noeuds.

3.1.9.2 Le modèle indépendant par cascade

Ce modèle de diffusion est un modèle temporel. A chaque itération de l'algorithme, un noeud aura une probabilité, s'il est actif, d'activer par cascade, ses noeuds voisins. A l'itération suivante, tous les noeuds activés, au pas précédent, auront aussi une probabilité d'activer les noeuds voisins.

Il existe une version asynchrone des deux modèles précédents (asynchronous independant cascade et asynchronous linear treshold [36]) permettant de faire les itérations en parallèle avec un léger délais.

3.1.10 L'identification d'influenceurs dans les réseaux sociaux

L'identification d'influenceurs est l'un des objectifs principaux du web marketing. Mais, de plus en plus, cela devient également important dans le domaine de la défense. En effet, l'identification des acteurs clés lors de conflits ou dans des organisations terroristes reste un problème majeur. Les premiers travaux pour identifier ces utilisateurs font l'analogie entre réseaux sociaux réels et les réseaux sociaux en ligne dans lesquels le bouche à oreille est le meilleur moyen de diffuser des informations. Le rôle du bouche à oreille est, depuis longtemps, un sujet de recherche [37, 38]. Ces travaux montrent l'importance et la crédibilité de l'information lorsqu'elle provient d'une personne autre que la firme qui vend le produit. Matsumura et al. [39] ont proposé un modèle de diffusion de l'influence (IDM) en utilisant le principe de bouche à oreille. Pour cela, ils analysent la diffusion de l'information contenue dans des articles de blogs afin d'identifier les utilisateurs qui réutilisent les mêmes termes sur leur blog. Plus les articles sont similaires, plus l'influence est forte. IDM mesure la somme des termes propagés entre les blogs pour calculer un score d'influence.

Le réseau social Twitter est l'un des supports de recherche les plus utilisés dans les recherches d'identification d'influenceurs. Dans les travaux de Cha et al. [40], la méthode proposée est de mesurer l'influence d'un utilisateur en représentant cette dernière en fonction de trois mesures :

— Indregree influence, qui correspond au nombre d'abonnés et qui indique la taille de l'audience.
— Retweet influence, qui mesure le nombre de messages partagés et qui indique la capacité de cet utilisateur à diffuser ses messages.
— Mention influence, qui mesure le nombre de fois où cet utilisateur est mentionné et qui indique

la capacité de converser.

Ces trois mesures permettent, dans un premier temps, d'identifier les utilisateurs qui sont les plus représentatifs dans chaque mesure. Mais l'intersection de ces mesures permet d'identifier les utilisateurs influents et permet de donner une mesure globale de l'influence. De plus, la corrélation entre ces trois mesures a démontré que le nombre d'abonnés n'avait pas de lien avec le nombre de retweets et de mentions. Dans cette même continuité, Anger et Kittl [41] proposent d'utiliser le ratio Retweet/Mention et le ratio du nombre d'utilisateurs qui retweete ou mentionne divisé par le nombre de followers. Ces deux ratios sont ensuite additionnés et divisés par 2 pour créer un score (SNP). Les résultats ont été comparés avec le Klout score (section 3.1.8.6) et sont proches mais les auteurs ont identifié un biais lorsque le nombre de followers est trop important.

3.2 Détection de communautés

La détection de communautés est l'un des grands axes de recherche sur les réseaux sociaux. Dans cette section nous allons définir ce qu'est une communauté, comment elle est représentée sur les réseaux sociaux et, les différentes techniques mises au point pour pouvoir les identifier.

Définition 3 (Une communauté) *est un ensemble d'individus partageant des liens sociaux ou des intérêts communs.*

Dans le domaine des réseaux sociaux en ligne, la définition de communauté reste la même. Il s'agit également d'individus reliés entre eux par des liens sous la forme d'abonnements, mais aussi, par rapport aux interactions qu'ils peuvent avoir. Ils peuvent aussi être identifiés par rapport aux différentes thématiques qu'ils abordent dans leurs messages. La détection de communautés permet d'identifier des utilisateurs qui forment implicitement des groupes *c.a.d*, qu'ils vont, le plus souvent, interagir entre eux plutôt qu'avec les autres utilisateurs au sein du même réseau. Un utilisateur peut appartenir à plusieurs communautés. L'intérêt de détecter ces communautés permet d'identifier les acteurs clés du réseau, améliorer les systèmes de recommandations et pouvoir faire des actions ciblées telles que des actions de marketing ou d'isolements/bannissements.

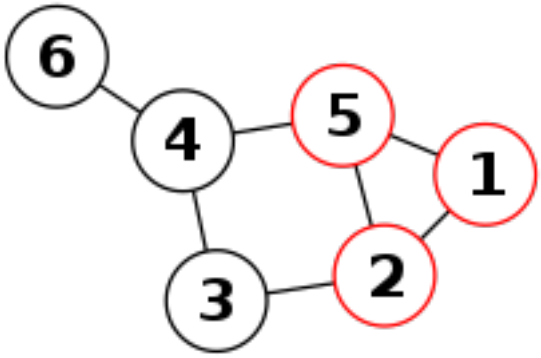

FIGURE 3.4 – Exemple de graphe contenant une clique en rouge. *Source Wikipedia : https://fr.wikipedia.org/wiki/Clique_(théorie_des_graphes)*

3.2.1 Communauté par distance

Cette technique de détection de communauté est très simple. Elle consiste à identifier les noeuds pour lesquels il existe un lien entre eux. Par exemple, dans un graphe, deux noeuds peuvent être considérés comme une communauté s'il existe une arête entre eux. D'après Kumar et al. [42], dans les réseaux sociaux en ligne, nous avons trois structures : une très grande composante connexe, des petites communautés et des utilisateurs isolés. La difficulté de cette approche est de trouver les communautés dans cette grande composante connexe.
Pour identifier ces communautés, nous utilisons principalement les notions de clique et de club. Une clique est un ensemble de noeuds d'un graphe formant un sous-graphe dans lequel, tous les noeuds sont adjacents, *c.a.d*, que chaque noeud possède une arête vers chaque noeud de ce sous-graphe. Un exemple de clique est représenté sur la figure 3.4, sur laquelle les noeuds 5, 2 et 1 forment une clique.

En utilisant la notion de clique, Wasserman et al. [43] proposent deux autres structures : les k-clique et les k-club. Une k-clique est un sous-graphe dans lequel la plus grande distance entre deux noeuds n'est pas plus grande que k.

$$d(u_i, u_j) \leq k \quad \forall u_i, u_j \in V_s \tag{3.11}$$

Où $d(u_i, u_j)$ est le plus court chemin entre u_i et u_j, V_s est l'ensemble des noeuds qui composent le sous-graphe. Par exemple, sur la figure 3.4 les ensemble de noeuds 1, 5, 2, 3, 4 et 6, 4, 5, 3 forment une 2-clique.

Un k-club est, quant à lui, un ensemble de noeuds pour lequel chaque noeud est a une distance maximale de k. C'est une notion plus stricte que le k-clique. Un club est souvent un sous-ensemble d'une k-clique. Dans le même type d'approche, nous avons la notion de k-plex [44], qui a été réutilisée pour détecter des communautés dans un réseau [45]. Un k-plex est un sous-graphe dans lequel tous les noeuds sont au moins connectés à k autres noeuds du même sous-graphe.
Bien que ces notions soient basiques, leur complexité varie entre une complexité exponentielle et une complexité non déterministe polynomiale. Dans le cas d'une utilisation de ces notions sur les réseaux sociaux (qui possèdent un nombre très conséquent d'utilisateurs ou de noeuds), le temps de calcul pourrait engendrer des résultats erronés du fait de la haute dynamicité des réseaux sociaux en ligne. En effet, si un calcul est trop long, les liens entre utilisateurs peuvent changer. A n'importe quel moment un utilisateur peut gagner ou perdre des abonnés.

3.2.2 Communauté par similarité

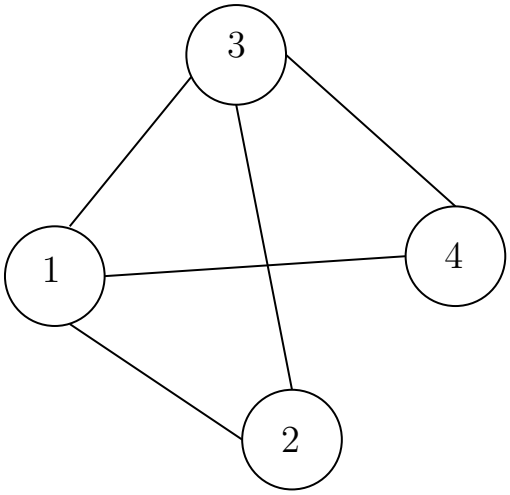

FIGURE 3.5 – Graphe simple avec des noeuds similaires

Noeuds	**1**	**2**	**3**	**4**
1	-	1	1	1
2	1	-	1	0
3	1	1	-	1
4	1	-	1	-

TABLE 3.1 – Matrice d'adjacence du graphe simple correspondant à la figure 3.5

Le but de cette approche est d'identifier des communautés en analysant la similarité des liens entre

utilisateurs. En d'autres termes, il s'agit de regarder le nombre d'amis en commun que partagent deux utilisateurs. Par exemple, sur la figure 3.5 les noeuds 1 et 3 sont identiques. On peut remarquer sur la matrice d'adjacence (tableau3.1) que les lignes de ces deux noeuds sont similaires.
Pour automatiser la détection de noeuds similaires, nous pouvons utiliser des mesures de similarité. Celles qui sont le plus souvent utilisées sont la distance de Jaccard [46] (équation 3.12) et la similarité cosinus [47] (équation 3.13).

$$Jaccard(u_i, u_j) = \frac{|N_i \bigcap N_j|}{|N_i \bigcup N_j|} \tag{3.12}$$

Où N_i est l'ensemble des noeuds voisins de i et, $|N_i \bigcap N_j|$ est la cardinalité de l'intersection des voisins de i et j.

$$Cosine(u_i, u_j) = \frac{|N_i \bigcap N_j|}{\sqrt{|N_i| \times |N_j|}} \tag{3.13}$$

Où $|N_i \bigcap N_j|$ est la cardinalité de l'intersection des voisins de i et, j $|N_i|$ la cardinalité du nombre de voisins de i.
Ces deux mesures de similarité donnent un résultat compris entre 0 et 1, en sachant que $|N_i \bigcap N_j|$ peut être un ensemble de vide.

3.2.3 Communauté par modularité

La modularité introduite par Newman and Girvan [48] propose de mesurer la force d'une communauté. Cette mesure compare les écarts entre le nombre d'arêtes observées de la communauté avec le nombre d'arêtes théoriques. Le nombre d'arêtes théoriques est calculé en plaçant des liens entre les noeuds de façon aléatoire. Plus la force d'une communauté est élevée, plus la communauté est compacte et meilleure est la partition.

$$Q(C) = \sum_{i,j \in C} A_{ij} - \frac{d_i d_j}{2m} \tag{3.14}$$

Où A_{ij} indique le nombre de liens entre le noeud i et j dans la matrice d'adjacence, d_i et d_j sont les degrés de centralité de i et j. m est le nombre d'arêtes observé et $\frac{d_i d_j}{2m}$ représente l'espérance d'avoir une arête entre i et j en suivant une distribution aléatoire. Dans le cas où le réseau est partitionné en

k communautés, le calcul de la modularité est défini comme ceci :

$$Q = \frac{1}{2m} \sum_{k=1}^{k} \sum_{i,j \in Ck} (A_{ij} - \frac{d_i d_j}{2m}) \tag{3.15}$$

On peut aussi écrire cette équation sous cette forme :

$$Q = \frac{1}{2m} Tr(S^T B S) \tag{3.16}$$

Où Tr est la trace, S est la matrice associant chaque noeud à une communauté, S^T est la transposée de S et B est la matrice de modularité pour laquelle $B_{ij} = A_{ij} - \frac{d_i d_j}{2m}$. Pour détecter des communautés à partir de la modularité, nous cherchons à maximiser la modularité Q et, la matrice S devient l'inconnue que l'on souhaite calculer.

La complexité en temps de calcul de cette équation est NP-difficile [49], ce qui rend son utilisation impossible lorsque le graphe dépasse les cent noeuds. Des méthodes d'optimisation de la modularité par déplacement de noeuds ou fusion de communautés [50] ont rendu le calcul plus efficace. Une d'entre elle est l'algorithme de Louvain de Blondel [1] qui permet de passer sur un temps de calcul en $O(m)$. La limite de cet algorithme est qu'il fait disparaître les petites communautés lorsque la taille du graphe devient trop conséquente (La limite de résolution [51]). Dans les travaux de Blondel et al. [1] (section 3.2.3), les auteurs appliquent l'algorithme de Louvain sur un réseau d'échanges téléphoniques en Belgique, comportant 2 millions d'utilisateurs (figure 3.6). Les deux communautés les plus importantes obtenues font apparaître une homogénéité de la langue parlée (le français et le néerlandais). En effet, plus de 85% des utilisateurs de ces communautés parlent la même langue. La communauté la plus hétérogène se trouve à la bordure entre ces deux communautés. Sur la figure 3.7 nous avons un exemple d'application de la méthode Louvain sur un graphe de relation entre les personnages de la série Game of Thrones. Les liens entre les personnages sont générés lorsque deux noms de personnages se trouvent à moins de 15 mots l'un de l'autre dans le troisième livre de la saga. Lorsque l'on connaît l'oeuvre, nous pouvons constater que les résultats obtenus sont de qualité.

3.2.4 Communauté par partitionnement hiérarchique

3.2.4.1 Algorithme divisif

Initialement introduit par Girvan et al. [52] la méthode de partitionnement hiérarchique a pour but d'identifier les communautés en supprimant progressivement les arêtes entre les noeuds. Cela permet

d'isoler rapidement des groupes de noeuds qui forment les communautés. Pour sélectionner les arêtes à supprimer, l'algorithme se focalise sur les liens entre les communautés. La méthode de Girvan et al. [52] repose sur le calcul de la centralité d'intermédiarité (voir section 3.1.4.3). Mais, au lieu de calculer cette mesure pour un noeud, elle est calculée pour les arêtes. On identifie l'arête par laquelle le nombre de passages de plus courts chemins entre les noeuds est maximal et, on supprime cette arête du graphe. À l'itération suivante, la mesure d'intermédiarité des arêtes est calculée seulement sur les arêtes qui ont été affectées par la suppression à l'itération précédente.

3.2.4.2 Algorithme agglomératif

A l'inverse, la méthode agglomérative démarre avec des singletons. Chaque noeud du graphe est isolé et correspond à une partition. Séquentiellement, les clusters de noeuds sont regroupés deux à deux selon la distance qui les sépare. Ce processus s'exécute jusqu'à ce que le graphe soit une partition unique. Pour calculer la distance entre les partitions, nous utilisons souvent deux méthodes : simple lien (Single-linkage) introduit par Sneath [53] et lien complet (Complete-linkage) introduit par Defays [54]. Dans le cas du simple lien, la distance entre deux clusters est calculée en prenant une simple paire de noeuds. Ensuite, la paire de noeuds possédant la plus petite distance est combinée pour former un cluster. Pour la méthode avec les liens complets, nous calculons la distance entre les deux noeuds les plus éloignés de chaque cluster et combinons les clusters possédant le plus court chemin entre ces noeuds. Mathématiquement, nous avons ces deux formules :

$$D(X,Y) = \min_{x \in X, y \in Y} d(x,y) \qquad \text{(Simple lien)}$$

$$D(X,Y) = \max_{x \in X, y \in Y} d(x,y) \qquad \text{(Lien complet)}$$

L'avantage de cette méthode, est qu'il n'est pas nécessaire de renseigner de nombres de clusters. L'inconvénient reste la complexité en temps qui est de $O(n^2)$ pour le simple lien et $O(n^2 logn)$ pour le lien complet.

3.2.5 Communauté avec algorithme génétique

Initialement proposés par Goldberg et al [55], les algorithmes génétiques sont généralement utilisés dans des heuristiques de recherche et d'optimisation dans des systèmes adaptatifs. Le modèle repose

sur des principes d'évolution (génétiques) naturels et principalement sur trois opérations : la mutation, le transfert et la sélection. L'utilisation des algorithmes génétiques dans la détection de communautés essaie de prédire la mutation des utilisateurs et ainsi, les associe à des communautés d'utilisateurs en se servant des liens externes. Les travaux de Pizzuti [56] proposent un algorithme génétique GA-Net. Cet algorithme propose de représenter un réseau sous la forme de gènes possédant plusieurs allèles. Les gènes représentent les noeuds du graphe et, les allèles sont les arêtes entre eux. Les noeuds participant au même composant sont rassemblés dans des clusters. Le gain de cette représentation permet de rendre le calcul des clusters linéaire. Ces travaux ont été repris par Tasgin et al [57] qui en ont amélioré la qualité en remplaçant la fonction d'objectif par une fonction de forme qui permet de ne plus avoir à renseigner le nombre de communautés.

3.2.6 Communauté par propagation d'étiquette

Cette approche consiste à propager une étiquette de noeud à noeud dans un réseau. Introduite par Raghavan et al. [58], cette méthode associe à chaque noeud une étiquette et, à chaque itération, un noeud peut changer d'étiquette par propagation en prenant l'étiquette la plus présente dans son voisinage. L'algorithme s'arrête lorsque tous les noeuds possèdent le label le plus présent dans son voisinage. Cette méthode a une complexité en temps de $O(m)$.

3.3 Conclusion

Dans ce chapitre, nous avons défini les notions d'influence et de popularité qui sont souvent confondues dans les recherches.

Nous avons présenté les différents profils d'utilisateurs présents dans les réseaux sociaux en ligne, ce qui permet de comprendre la complexité des structures de graphes sociaux. On constate que chaque noeud n'est pas toujours un lieu de passage pour l'information. Notamment dans le cas des utilisateurs inactifs et des bots qui partagent de l'information mais ne sont pas nécessairement contrôlés par des humains.

Les méthodes d'analyses de l'influence sont très diverses, allant d'une simple mesure topographique du graphe à des scores complets prenant en compte, à la fois la topographie du graphe mais aussi l'influence de chaque noeud du graphe. Ces mesures sont souvent reprises et adaptées aux types de réseaux.

Pour terminer, les techniques de partitionnement de graphe sont, depuis longtemps, des sujets de recherche. De plus, avec l'apparition des réseaux sociaux en ligne, les approches tentent d'être plus précises ou essaient d'optimiser les algorithmes déjà existants pour pouvoir réduire le temps de calcul. Ces travaux nous permettent donc d'identifier et d'isoler des groupes d'utilisateurs ainsi que de mettre en avant ceux avec une haute autorité ou un haut potentiel de propagation d'informations.

Dans le chapitre suivant, nous présentons une chaîne de traitements de données permettant d'identifier des utilisateurs anormaux en utilisant leurs méta-données.

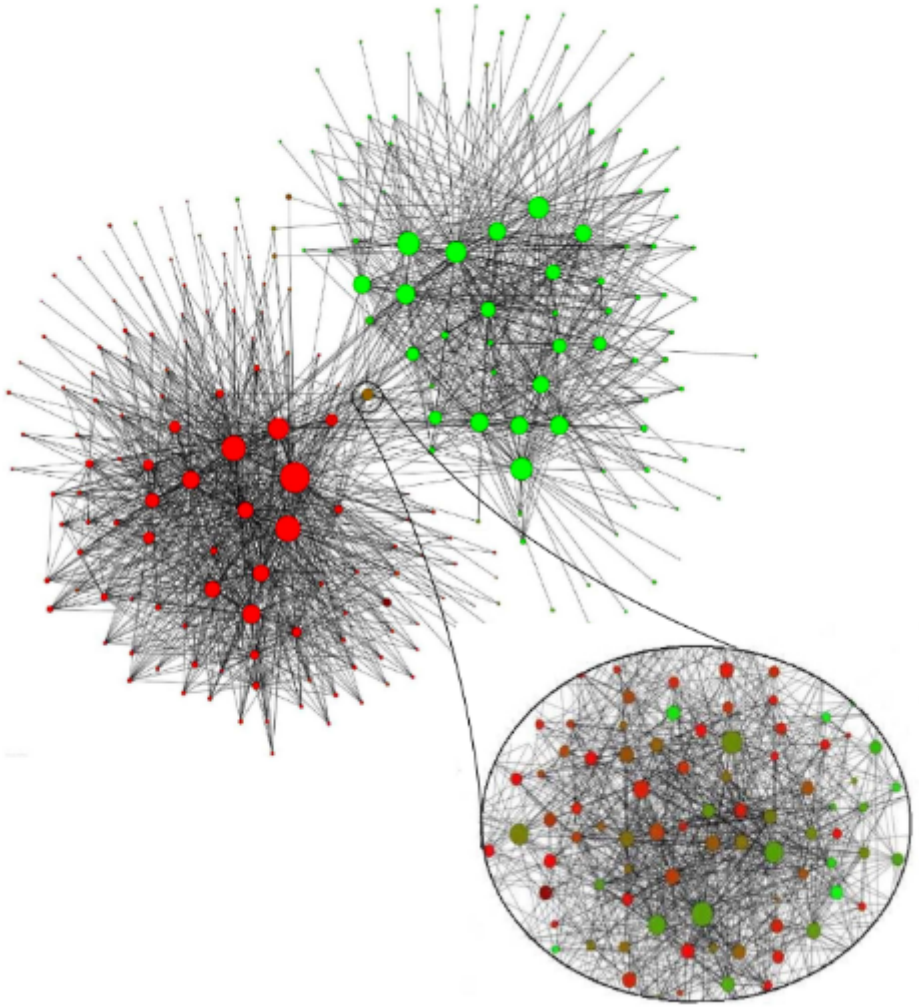

FIGURE 3.6 – Représentation graphique des communautés extraites à partir d'un réseau d'échanges téléphoniques en Belgique. (*Source Blondel et al. [1] Fast unfolding of communities in large networks*)

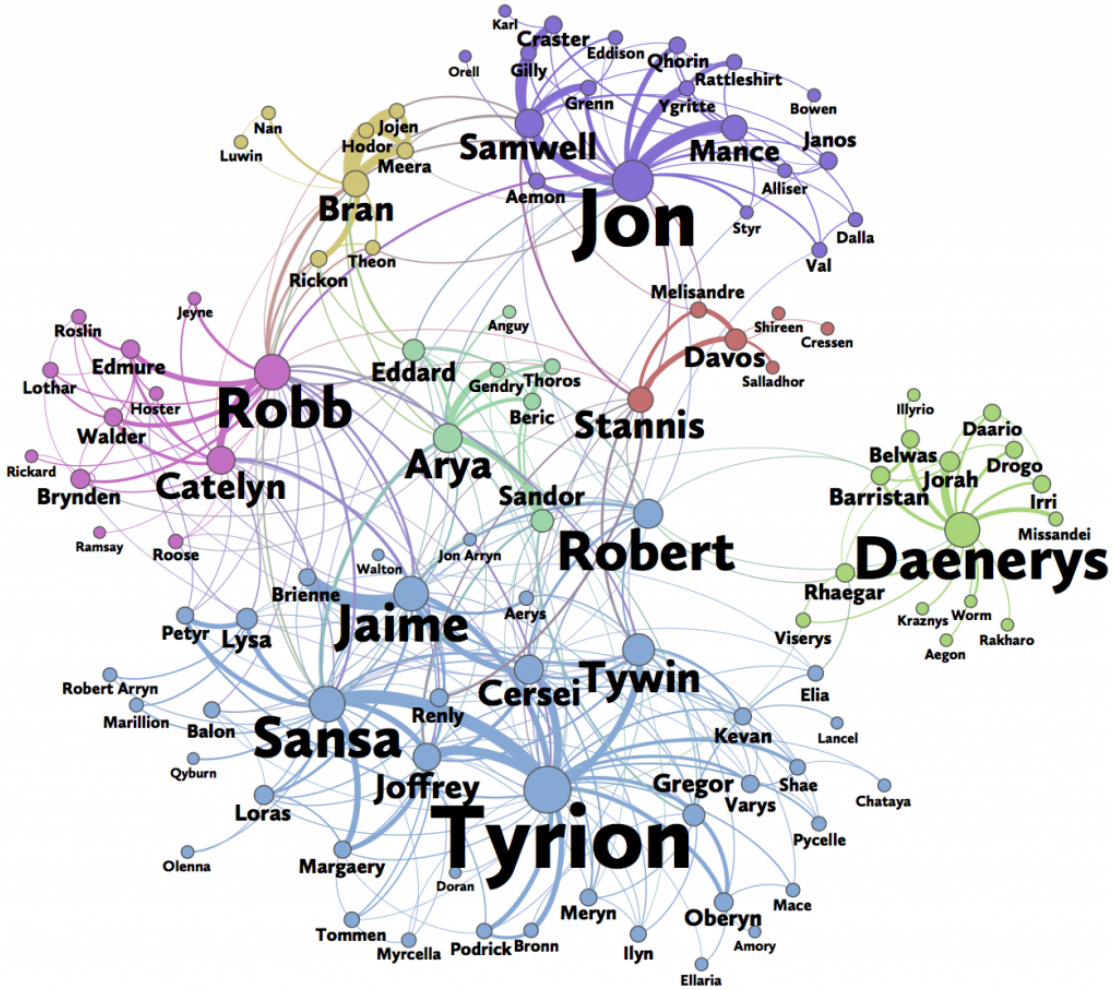

FIGURE 3.7 – Graphe des relations des personnages de Game of Thrones réalisé par Andrew Beveridge and Jie Shan selon la méthode Louvain. `https://www.maa.org/sites/default/files/pdf/Mathhorizons/NetworkofThrones(1).pdf`

Chapitre 4

Analyse des comportements anormaux

Dans ce chapitre, nous allons présenter une courte analyse utilisant le machine learning pour identifier les utilisateurs anormaux sur le réseau social Twitter. Ces analyses se focalisent sur l'utilisation des méta-données des comptes utilisateurs, en utilisant une approche non-supervisée. Ces travaux ont permis de caractériser les comptes de faux populaires comme les bots.

Ce chapitre ne contient pas de contributions théoriques mais fait partie des recherches menées dans le cadre de projets de recherche au sein de mon entreprise et me permet d'introduire les travaux de cette thèse.

4.1 Définition d'anomalie

4.1.1 Qu'est-ce qu'une anomalie ?

Une anomalie est ce que nous n'attendons pas, ce qui ne suit pas la norme et que l'on peut considérer comme anormal . Dans notre cas d'étude, nous allons également utiliser la définition de valeur aberrante donnée par Hawkins [59] :

Définition 4 *An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism.*

Autrement dit, une valeur aberrante est un élément d'un jeu de données qui s'écarte distinctement du reste des données.

4.1.2 Les différentes catégories

Les définitions présentées ci-dessous proviennent de l'article *A Review of Anomaly Detection Systems in Cloud Networks and Survey of Cloud Security Measures in Cloud Storage Applications* [2].

4.1.2.1 Les anomalies ponctuelles

Les anomalies ponctuelles définissent une instance de données individuelles qui peuvent être anormales par rapport au reste des données ou trop éloignées du reste des données. La figure 4.1 illustre cette définition sur laquelle les points marqués d'un **N** sont normaux et ceux marqués d'un **O** sont anormaux.

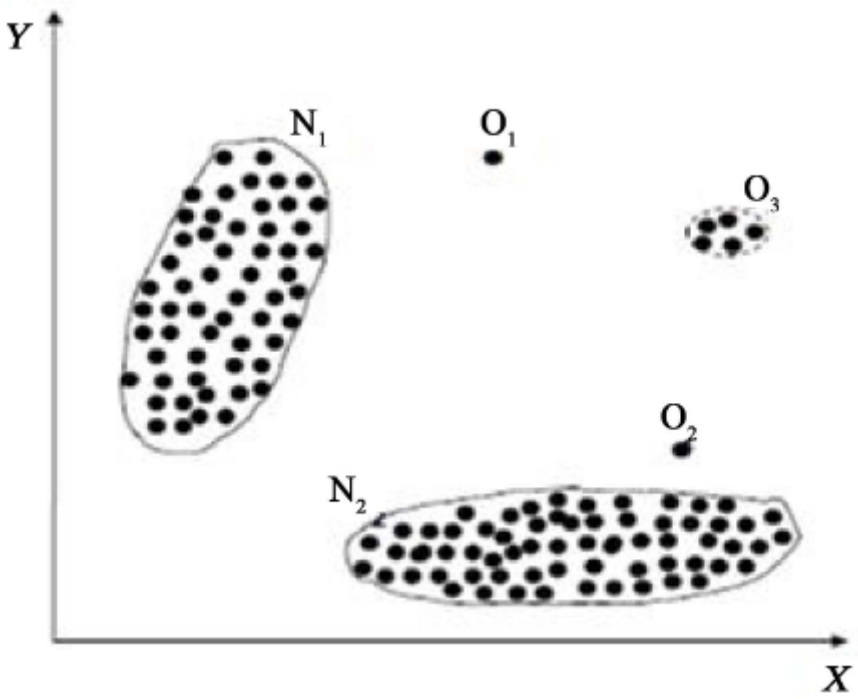

FIGURE 4.1 – Illustration d'une anomalie ponctuelle, Source : *A Review of Anomaly Detection Systems in Cloud Networks and Survey of Cloud Security Measures in Cloud Storage Applications* [2]

4.1.2.2 Les anomalies contextuelles

Les anomalies contextuelles définissent une instance de données individuelles qui sont anormales dans un contexte particulier : si elles s'écartent de manière significative en fonction d'un contexte sélectionné. Par exemple, la température de -5°C à Paris, est une anomalie si nous sommes en été mais

pas si nous sommes en hiver. La figure 4.2 illustre cet exemple. Il existe deux types d'attributs pour définir ces anomalies :

— Les attributs contextuels : qui définissent le contexte, comme le temps, la localisation, etc.

— Les attributs de comportement : qui caractérisent l'instance de données, comme la température ou l'activité.

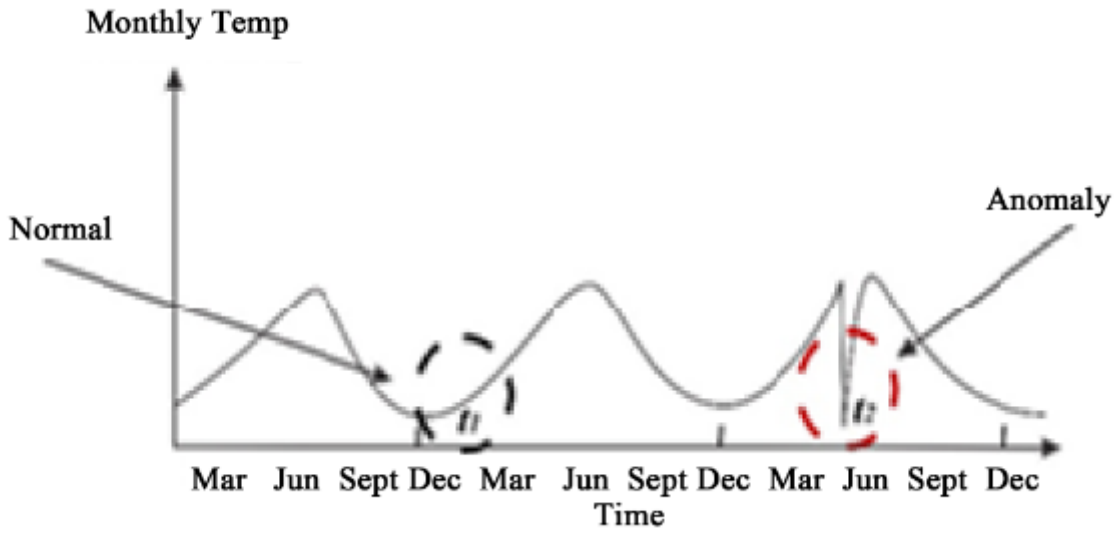

FIGURE 4.2 – Illustration d'une anomalie contextuelle, Source : *A Review of Anomaly Detection Systems in Cloud Networks and Survey of Cloud Security Measures in Cloud Storage Applications* [2]

4.1.2.3 Les anomalies collectives

Les anomalies collectives définissent un comportement collectif inattendu pendant un certain temps en comparaison avec l'ensemble des données. La figure 4.3 illustre un exemple d'électrocardiogramme sur lequel, la région entourée représente une anomalie collective lorsqu'elle est comparée au reste des données. Mais les valeurs successives, elles-mêmes ne sont pas anormales.

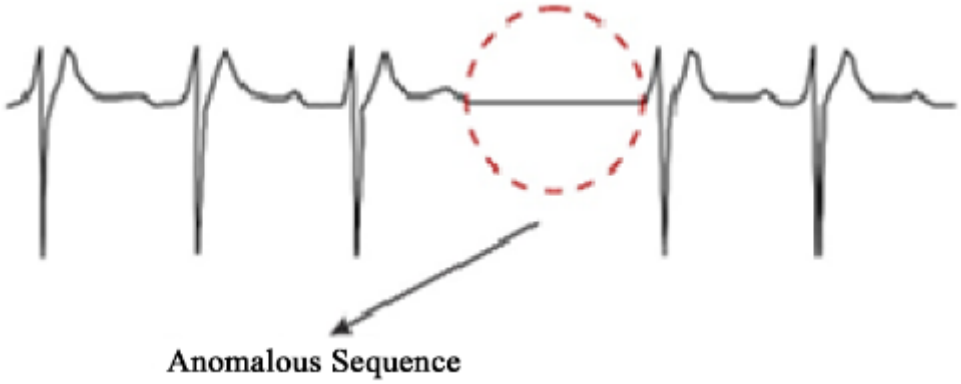

FIGURE 4.3 – Illustration d'une anomalie collective. Source : *A Review of Anomaly Detection Systems in Cloud Networks and Survey of Cloud Security Measures in Cloud Storage Applications* [2]

4.2 Définition de méta-données

Les méta-données sont des données qui décrivent d'autres données. Elles résument les informations. Par exemple, si nous prenons un livre, les méta-données décriraient les caractéristiques du livre (l'auteur, la date de parution, le nombre de pages, le sujet, etc.). Mais cette description ne raconte pas toute l'histoire.

Dans notre cas, nous allons utiliser les méta-données des comptes utilisateurs de Twitter (figure 4.4) :

— La donnée : le compte utilisateur

— Les méta-données : les informations à propos de ce compte

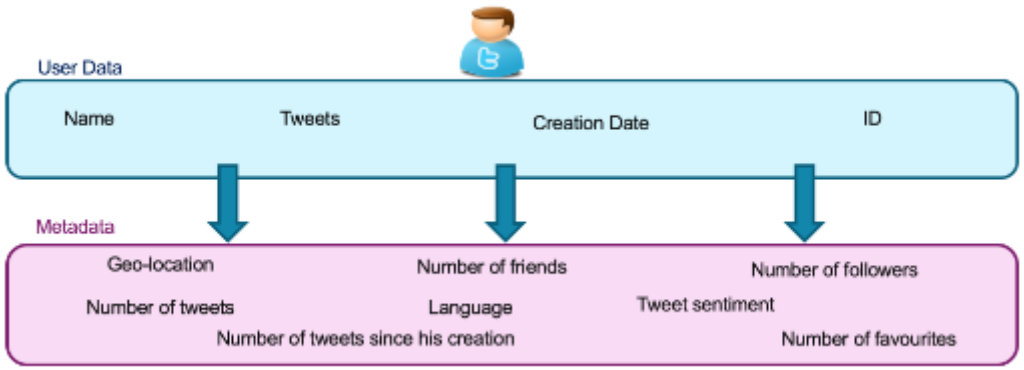

FIGURE 4.4 – Illustration des méta-données d'un compte utilisateur de Twitter

4.3 Jeux de données et traitements

Dans cette section, nous allons expliquer comment le jeu de données est construit et quels sont les traitements effectués pour ajouter de l'information ainsi qu'appliquer des algorithmes de pré-traitement.

4.3.1 Présentation du jeu de données

Le jeu de données est formé à partir de tweets provenant de l'API stream de Twitter, nous avons collecté 50 millions d'utilisateurs sur une période d'un mois en 2018. Nous avons utilisé plusieurs méta-données fournies par Twitter et calculé de nouvelles données à partir de celles-ci.

- **Followers Count** : nombre d'abonnés d'un utilisateur.
- **Friends Count** : nombre d'abonnements d'un utilisateur.
- **Listed Count** : nombre de listes de suivi auxquelles l'utilisateur appartient. Ces listes sont créées par des utilisateurs pour personnaliser, organiser et prioriser les tweets sur leur fil d'actualité.
- **Favourites Count** : nombre de likes que l'utilisateur a fait depuis sa création.
- **Tweets per day** : cette donnée est calculée lors du pré-traitement et, a pour but d'estimer le nombre de messages postés par jour. Elle se calcule en divisant le nombre de messages total de l'utilisateur par le nombre de jours depuis la création du compte.
- **Ration Friends/Followers** : aussi calculée lors du pré-traitement, elle divise le nombre d'abonnements par le nombre d'abonnés.
- **Followers Score** : ration entre le nombre d'abonnés et le nombre d'abonnés de l'utilisateur le plus populaire. Au moment de l'étude, le record de nombre d'abonnés était détenu par la chanteuse américaine Katy Perry avec plus de 100 millions d'abonnés.

	Followers Count	**Friends Count**	**Listed Count**	**Favourites Count**
mean	3.14e+04	2.33e+03	3.17e+04	9458.75
std	6.82e+05	2.31e+04	6.82e+05	26293.45
min	1.00	1.00	1.00	1.00
25%	1.03e+02	1.06e+02	1.03e+02	159
50%	4.01e+02	2.925e+02	4.01e+02	1635
75%	1.67e+03	7.62e+02	1.67e+03	7794.5
max	6.10e+07	1.20e+06	6.10e+07	669716

TABLE 4.1 – Statistiques basées sur un échantillon de 100 000 utilisateurs

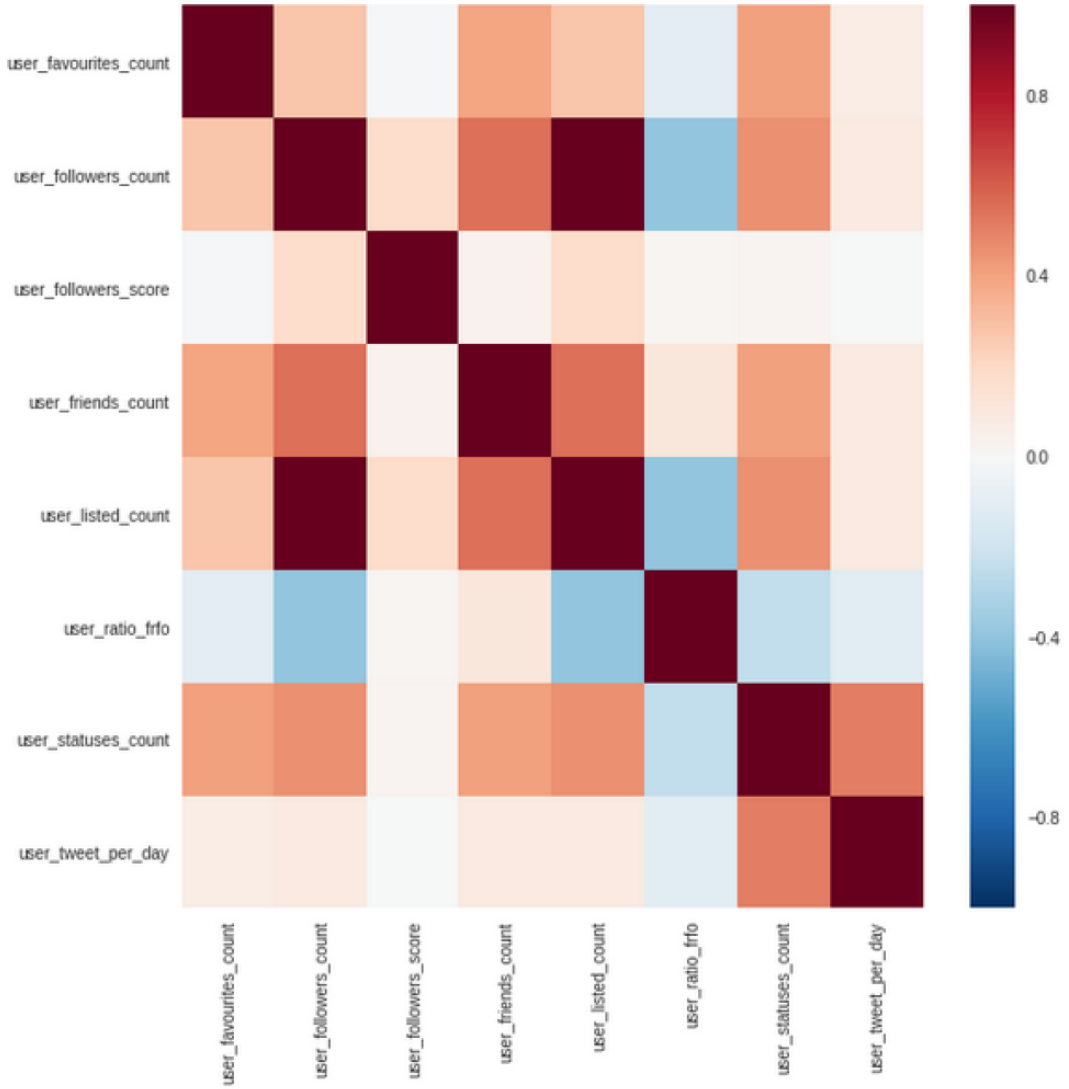

FIGURE 4.5 – Corrélation entre les différentes méta-données

Sur le tableau 4.1, nous pouvons constater que les données brutes provenant de Twitter possèdent une déviation standard (std) très importante. En utilisant les données telles quelles, avec un algorithme non-supervisé, les résultats ne seront pas exploitables. Nous avons donc appliqué un algorithme de normalisation, Robust Scaler [1]. Cet algorithme de normalisation de données est robuste aux valeurs anormales. Il commence par supprimer la médiane et, utilise ensuite les valeurs inter-quantiles (entre le premier et le troisième quantile) pour normaliser les données. Les figures 4.6 et 4.7 montrent le résultat de la normalisation.

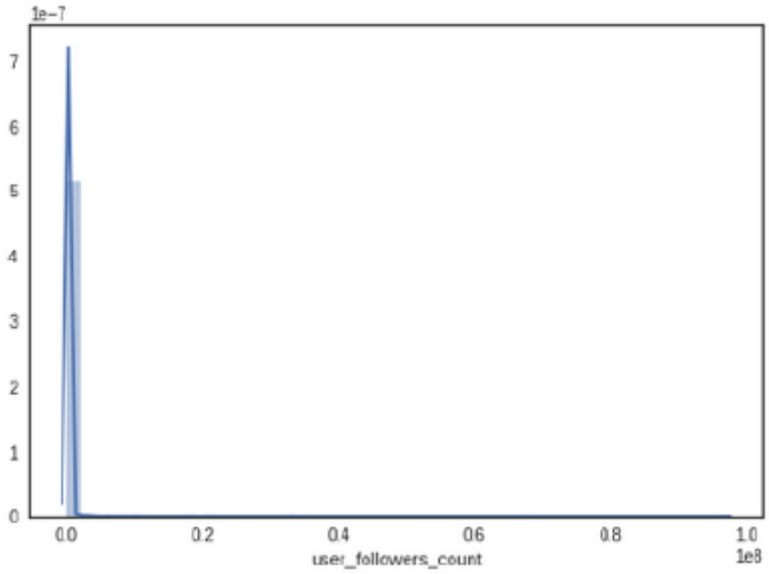

FIGURE 4.6 – Distribution des données pour le nombre de followers avant normalisation.

1. https ://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html

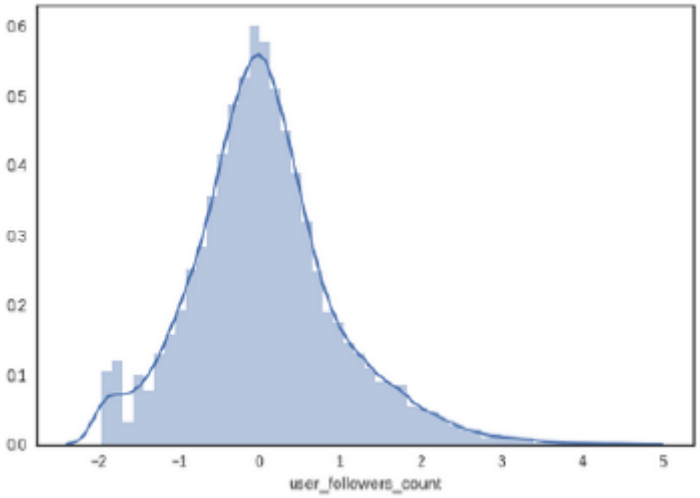

FIGURE 4.7 – Distribution des données pour le nombre de followers après normalisation.

4.3.2 Réduction des dimensions

La réduction des dimensions est un processus permettant de réduire le nombre de variables en construisant des valeurs dérivées, destinées à être informatives et non redondantes pour faciliter l'apprentissage. En plus de simplifier l'entraînement et la complexité, la réduction des dimensions permet de réduire le temps d'entraînement sur de grands jeux de données. Nous utilisons aussi ce processus pour dessiner nos différents graphiques dans le but de perdre moins d'informations et de rendre possible une représentation de nos données en 2D.

4.3.2.1 ACP : L'analyse en composantes principales

L'ACP est un moyen d'identifier des motifs dans les données et, d'exprimer les données de manière à mettre en évidence leurs similitudes et leurs différences [60].Après avoir identifier les motifs dans les données, ces dernières sont compressées. Cela permet de réduire le nombre de variables sans perdre trop d'informations. L'ACP utilise une transformation orthogonale pour convertir un ensemble d'observations éventuellement corrélées en un élément réduit non corrélé.

4.3.3 Estimation du taux de contamination

Dans le but d'intégrer ces travaux dans un projet de R&T, nous avons cherché à automatiser la recherche du taux de contamination d'un jeu de données. Le taux de contamination est une variable à fournir dans la plupart des algorithmes non-supervisés. Il représente l'estimation de données bruitées. Les travaux de Elbatta et Ashour [61] proposent une approche utilisant l'algorithme des plus proches voisins pour identifier la valeur Epsilon pour l'algorithme DBSCAN. Nous avons décidé de réutiliser ces travaux pour estimer le taux de contamination dans notre jeu de données. La première étape est de calculer la distance entre chaque point de notre jeu de test. Il faut ensuite ordonner les distances dans l'ordre croissant. Enfin, identifier le point de décrochage entre la courbe des distances et sa dérivée. La figure 4.8 montre une représentation graphique de cette méthode.
Le taux de contamination identifié avec cette méthode est $\approx 13\%$.

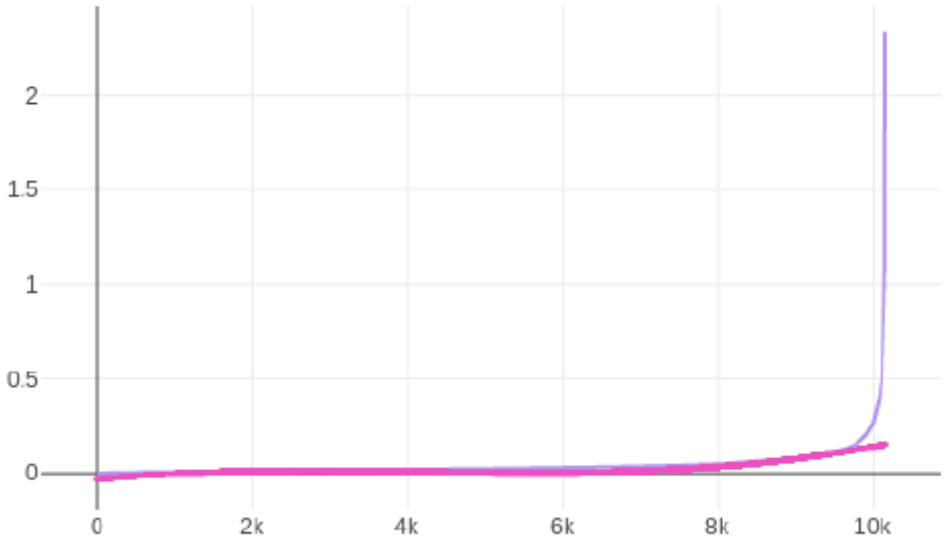

FIGURE 4.8 – Représentation graphique des distances (violet) et de la dérivée (rose)

4.4 Expériences et résultats

Dans cette section, nous présentons les différents résultats obtenus avec différents algorithmes non-supervisés. En utilisant la méthode d'estimation du taux de contamination vue dans la section précédente, nous avons identifié un taux de contaminations proche des 15%. Nous avons gardé ce taux

pour toutes les expériences suivantes. Pour essayer de comprendre ce que nous cherchons à identifier, nous avons tracé un graphique montrant la carte de chaleur du jeu de données (figure 4.9). Nous pouvons observer une zone très dense dans le centre, ce qui est, pour nous, le comportement majoritaire que nous allons classer comme normal. Nous allons donc chercher un algorithme qui respecte ce noyau d'utilisateurs normaux.

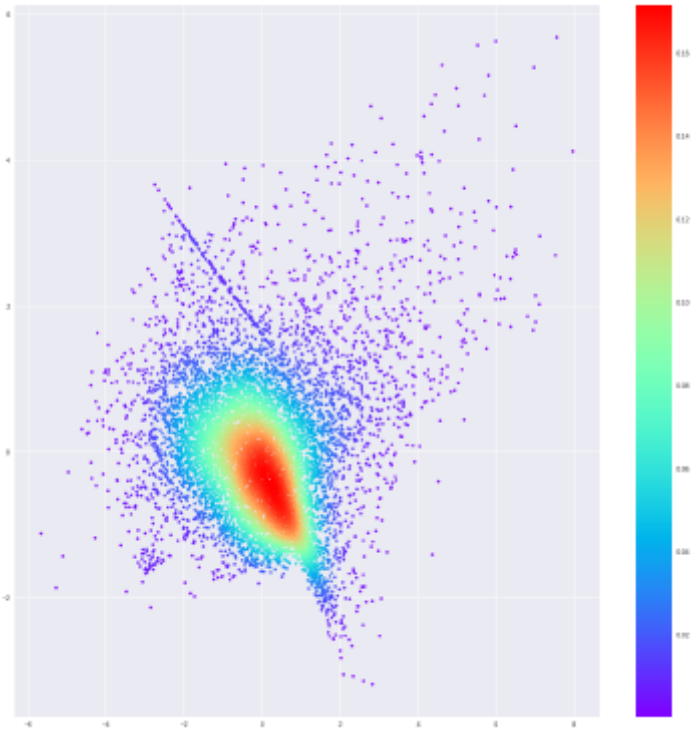

FIGURE 4.9 – Représentation de la densité des données en utilisant le nombre de Followers et le nombre de Friends

4.4.1 K-means

Le clustering K-means [62] est une méthode de quantification vectorielle. Elle vise à diviser n observations en m dimensions et de les répartir dans k clusters dans lesquels chaque observation n appartient au cluster dont la moyenne est la plus proche de la valeur du point. L'algorithme K-means est l'un des algorithmes de regroupement les plus rapides. L'algorithme K-means [62] commence par estimer k centroïde (moyenne des points), qui peuvent être générés ou sélectionnés aléatoirement. L'algorithme comporte deux étapes :

— L'étape d'attribution des données : Chaque centroïde définit l'un des clusters. Dans cette étape, chaque point de données est affecté au centroïde le plus proche, basé sur la distance euclidienne au carré entre les deux. Plus formellement, si c_i est la collection de centroïde dans l'ensemble C alors chaque point de données x est affecté à un cluster basé sur la formule suivante :

$$\arg\min_{c_i \in C} dist(c_i, x)^2 \tag{4.1}$$

Où $dist(c_i, x)$ est la distance euclidienne entre le point x et la centroïde c_i.

— La mise à jour des centroïdes : lorsque tous les points sont attachés à un cluster, on calcule le nouveau centre de gravité du cluster en calculant la moyenne des points appartenant au même cluster.

L'algorithme itère sur ces deux étapes jusqu'à atteindre un critère d'arrêt qui peut être : aucun point change de cluster, le calcul des centroïdes ne change pas ou bien, on a atteint un nombre d'itérations maximales. K-means garantit la convergence vers l'un de ces critères d'arrêt.

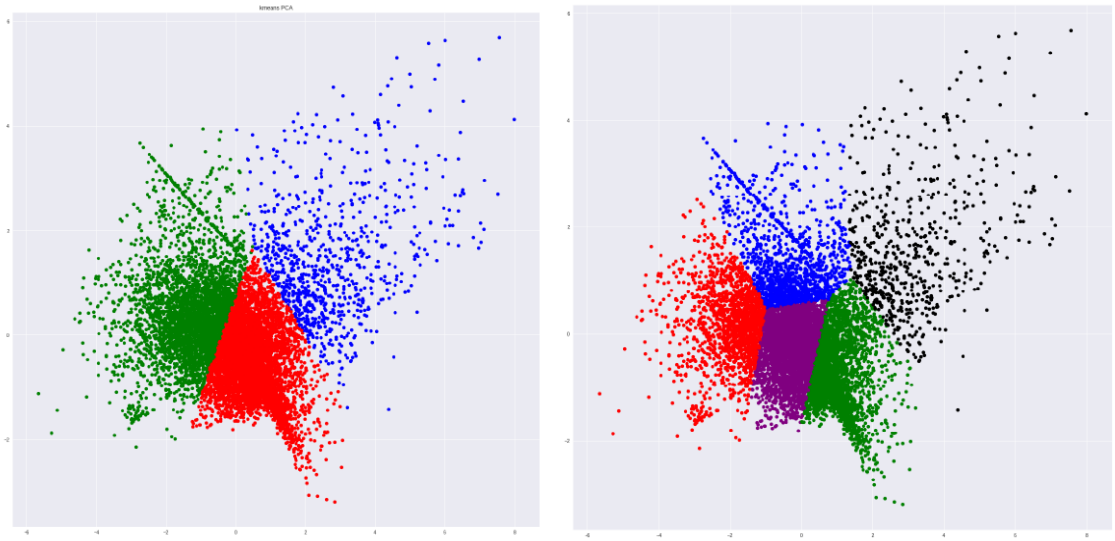

FIGURE 4.10 – Résultats du K-means pour k=3 et k=5

Les résultats obtenus avec K-means (figure 4.10) montrent un bon partitionnement des données, mais nous perdons l'idée d'avoir un cluster normal au centre où il y a la plus forte densité d'utilisateurs et, les utilisateurs sont souvent similaires d'un cluster à un autre.

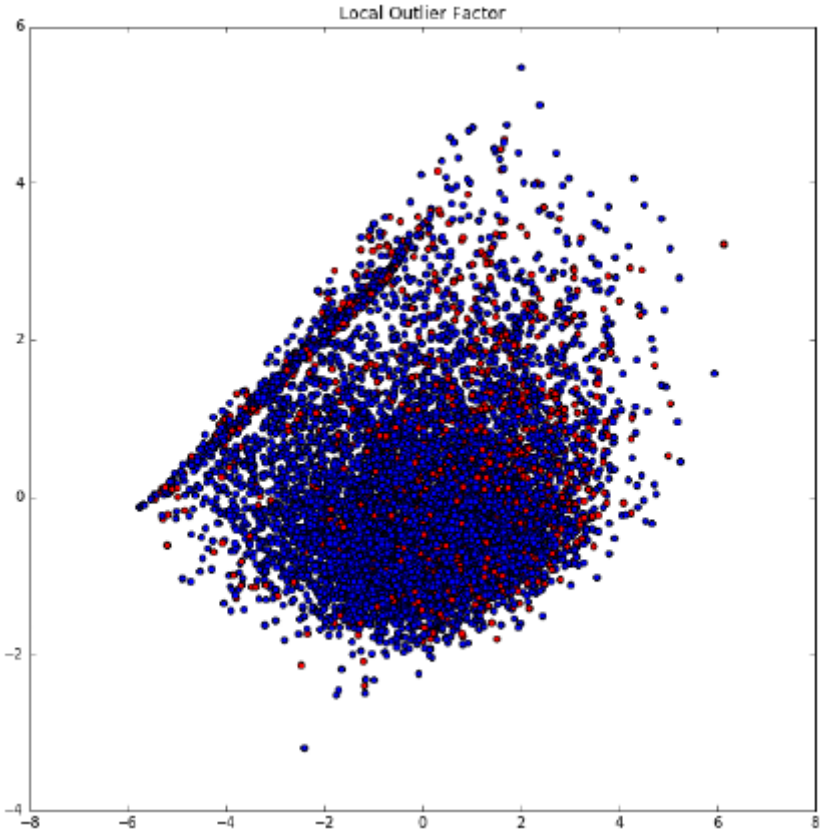

FIGURE 4.11 – Résultat du Local Outlier Factor (en rouge les données anormales)

4.4.2 Local Outlier Factor

Local Outlier Factor [63] est un algorithme conçu pour détecter des données anormales en mesurant la déviation locale d'un point par rapport à ses voisins. La localité d'un point est calculée en comparant la densité locale de ses voisins, en utilisant l'algorithme des *k-nearest neighbour* plutôt que d'utiliser la distribution globale des données. L'intuition est, qu'une valeur aberrante est significativement différente de la densité autour de ses voisins. En entrée de cet algorithme, nous donnons un nombre k correspondant à $k - plusprochevoisin$. De manière générale, $k = 20$ fonctionne, mais dépend du type des valeurs aberrantes que nous voulons détecter. Les résultats obtenus avec LOF (figure 4.11) montrent l'efficacité pour identifier des points déviants dans des zones denses.

4.4.3 Enveloppe Elliptique

L'algorithme d'enveloppe elliptique [64] suppose que si les données suivent une distribution gaussienne (distribution normale), nous pouvons définir la forme des données. Pour cela, l'algorithme calcule tout d'abord la covariance des données et dessine ensuite une ellipse sur les données centrales. Pour mesurer la distance entre chaque point, l'algorithme utilise la distance de Mahalanobis :

$$d_{\mu,\sigma}(x_i)^2 = (x_i - \mu)'\sigma^{-1}(x_i - \mu) \tag{4.2}$$

Où μ est la position et σ la covariance de la distribution gaussienne sous-jacente.

Les distances obtenues sont utilisées pour mesurer l'écart d'un point par rapport aux autres données. Ce qui permet d'ajuster l'ellipse aux données qui sont proches.

L'enveloppe elliptique (Figure 4.12) s'adapte bien aux données, mais sa forme elliptique passe peut-être à côté de certaines valeurs aberrantes ou inclut des valeurs aberrantes dans le cluster normal.

4.4.4 Isolation Forest

Isolation Forest, Liu et al. [65], est une méthode efficace pour détecter les valeurs aberrantes dans les ensembles de données de grandes dimensions. Il isole les observations en sélectionnant, au hasard, une caractéristique, puis sélectionne, aléatoirement, une valeur partagée entre les valeurs maximales et minimales de la caractéristique sélectionnée. Isolation Forest construit des arbres d'isolement à partir des données, puis identifie les anomalies en sélectionnant les arbres dont la longueur des branches est inférieure à la moyenne des longueurs de tous les arbres.

Les résultats obtenus (figure 4.13) montrent qu'il identifie bien la zone dense comme une zone de normalité, malgré quelques points détectés comme anormaux. La zone périphérique, sur laquelle on peut distinguer des points qui s'éloignent, est principalement identifiée comme une zone anormale.

Pour conclure ces résultats, nous avons décidé de garder l'algorithme Isolation Forest qui donne des résultats satisfaisants entre Local Outlier Factor et Enveloppe Elliptique. De plus, il possède un temps de calcul peu coûteux [65] $O(n \log m)$ par rapport à Enveloppe Elliptique qui mesure la distance entre chaque point ce qui est rapidement impossible sur de gros jeux de données.

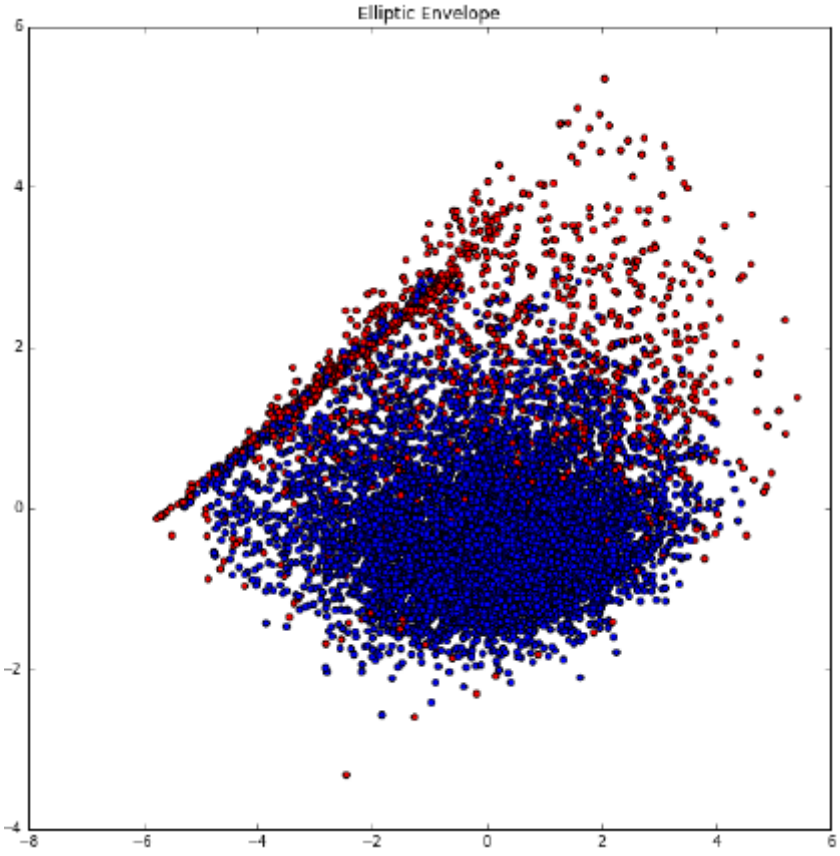

FIGURE 4.12 – Résultats de l'algorithme Enveloppe Elliptique (en rouge les données anormales)

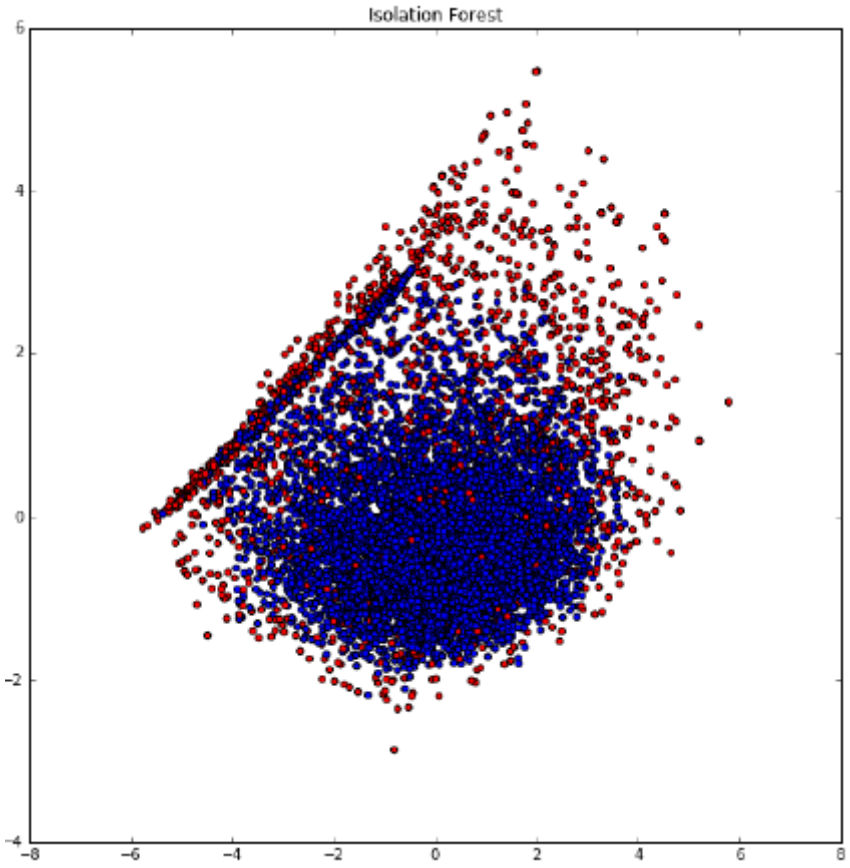

FIGURE 4.13 – Résultats de l'algorithme Isolation Forest (en rouge les données anormales)

Parmi les utilisateurs identifiés comme anormaux, nous avons effectué une vérification humaine sur un échantillon de 100 utilisateurs et nous avons identifié plusieurs types de comportements :

— **Influenceurs** : ce sont des utilisateurs possédant un grand nombre de followers et publiant un grand nombre de messages (exemple sur la figure 4.14).

— **Rebroadcaster** : ils sont souvent contrôlés par des bots. Ces utilisateurs possèdent un grand nombre de followers et repartagent automatiquement le contenu des utilisateurs auxquels ils sont abonnés. En règle générale, ce type d'utilisateur demande à être suivi pour pouvoir bénéficier de ces services (figure 4.15).

— Robot (bot) : ce sont des utilisateurs qui envoient des tweets automatiquement, ils suivent beaucoup d'utilisateurs mais, eux, sont peu ou pas suivis. (exemple 4.16)

— Compte suspendu : des utilisateurs qui sont bannis de Twitter après notre collecte de données (exemple 4.17).

FIGURE 4.14 – Utilisateur influent détecté comme anormal

FIGURE 4.15 – Robot rebroadcaster détecté comme anormal

FIGURE 4.16 – Compte d'un bot news, publiant automatiquement des articles de journaux souvent issus d'autres médias.

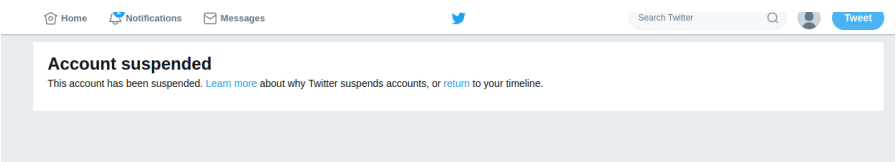

FIGURE 4.17 – Utilisateur banni de Twitter

4.5 Conclusion

Dans ce chapitre, nous avons proposé une approche pour identifier des utilisateurs anormaux. En utilisant les méta-données qu'ils peuvent générer sur le réseau social Twitter. Nous avons mis en place une chaîne de traitement des données : pré-traitement pour normaliser les données, réduction de dimension en utilisant une Analyse de la Composante Principale (un procédé pour mesurer le taux de contamination d'un jeu de données) et enfin, une classification (en utilisant des algorithmes non-supervisés).
Les différents résultats ont été comparés par une étude graphique ainsi qu'une validation humaine et notre choix s'est porté sur l'algorithme Isolation Forest. Ces travaux ont été intégrés dans un projet de recherche au sein de l'entreprise et identifient, dans une chaîne, les utilisateurs anormaux.

Dans le chapitre suivant, nous présentons un modèle d'analyse de l'évolution de la popularité et un modèle d'identification d'utilisateurs devenant populaires.

Chapitre 5

Détection et prédiction d'utilisateurs populaires

Ce chapitre présente notre première contribution, la détection précoce d'utilisateurs populaires. Cela consiste à détecter, le plus tôt possible, des utilisateurs devenant populaires. Pour cela, nous nous sommes intéressés à des techniques souvent utilisées en bio-informatique [66]. Dans ce domaine de recherche, les chercheurs mettent au point des techniques d'analyses de séquences d'ADN ou ARN pour identifier des protéines. Notre approche est basée sur deux algorithmes populaires du patterns mining (APRIORI [67] et FPGrowth [68]), ainsi que des méthodes d'identification de sous-chaînes comme SPADE [69], qui cherche à identifier, dans du texte, des motifs séquentiels, ou FreeSpan [70], qui permet d'extraire des motifs en partitionnant l'espace de recherche. Ces différents travaux nous ont permis de mettre au point le modèle d'analyses présenté dans ce chapitre.

Dans une première partie, nous allons aborder la caractérisation de la popularité, la manière de modéliser la popularité sur Twitter et, le modèle utilisé pour représenter les données. Dans une seconde partie, nous présenterons notre jeu de données et une analyse de la popularité. Dans une troisième partie nous présenterons notre méthode de filtrage, basé sur des motifs fréquents ainsi qu'un modèle d'identification de motifs pouvant être utilisés dans un flux de données continu. Enfin, nous comparerons les performances de notre modèle d'identification avec d'autres solutions et nous présenterons les résultats obtenus.

5.1 Le modèle de données

Nous introduisons dans cette section nos notations et notre modèle de données. Nous considérons la plate-forme Twitter et son graphe orienté sous-jacent $(\mathcal{U}, \mathcal{F})$ où $\mathcal{U}$ dénote l'ensemble de nœuds, c'est-à-dire les utilisateurs, et $\mathcal{F} \subseteq \mathcal{U} \times \mathcal{U}$ est l'ensemble d'arêtes, tel que $(u_1, u_2) \in \mathcal{F}$ signifie l'utilisateur u_2 est abonné à l'utilisateur u_1.

5.1.1 La popularité des comptes utilisateurs

Notre objectif est d'effectuer une détection précoce des (futurs) utilisateurs populaires. En supposant l'existence de la fonction d'abonnement $Follow : \mathcal{U} \times [0, T] \rightarrow \mathbb{N}$ qui renvoie au nombre d'abonnés pour un compte $u \in \mathcal{U}$ à l'instant $t \in [0, T]$, nous adoptons dans ce chapitre la définition suivante :

Définition 5 (Popularité) *La popularité d'un compte correspond à sa visibilité, c'est-à-dire, combien de personnes peuvent lire, commenter, propager un message que ce compte produit. Il est ici simplement estimé par le nombre de followers $Follow(u, t)$ qu'un utilisateur u possède à l'instant t.*

En respectant cette définition, nous proposons la classification de comptes suivante :

- **Les comptes non populaires** : cette classe regroupe les utilisateurs qui ne sont jamais populaires tout au long de la période d'observation $[0, T]$. Donc, en supposant un seuil d'impopularité φ, ces comptes vérifient que : $u \in \mathcal{U}$, $\forall t \in [0, T] : Follow(u, t) \leq \varphi$.
- **Les comptes populaires** : cette classe correspond aux utilisateurs déjà populaires sur notre période d'étude $[0, T]$. En supposant donc un seuil de popularité ε, cela signifie que nous avons à $t = 0$ $Follow(u, 0) \geq \varepsilon$.
- **Les futurs comptes populaires** : cette classe regroupe des utilisateurs qui ne sont pas populaires au début de notre période $[0, T]$ mais qui le sont à la fin. Donc, basé sur nos deux seuils ε et φ, cela correspond à des utilisateurs $u \in \mathcal{U}$ tel que $Follow(u, 0) \leq \varphi$ et $Follow(u, T) \geq \varepsilon$.

5.1.2 L'évolution de la popularité

Nous supposons que la plateforme Twitter met à jour périodiquement ses statistiques pour chaque utilisateur. La période entre deux mises à jour est constante et, est considérée, dans ce qui suit, comme notre unité de temps indivisible. Ainsi, à chaque instant instant $t \in [0, T]$, nous rapportons

pour chaque compte u le nombre de followers. Pour estimer l'évolution de la popularité, nous calculons le gain en nombre de followers entre deux pas de temps. Pour réduire l'impact de la taille des comptes, nous proposons d'utiliser la fonction log sur le gain, car un gain de 10 000 followers doit être considéré comme un gain du même ordre qu'un gain de 20 000 followers lorsque l'utilisateur dispose d'un compte populaire. En raison de la définition du domaine de la fonction log, nous utilisons la définition suivante pour la fonction de gain.

Définition 6 (le gain de popularité) *Considérant un instant $t \in [0, T-1]$, le gain de popularité pour un utilisateur $u \in \mathcal{U}$ est :*

$$gain(u,t) = \begin{cases} \log(Follow(u,t) - Follow(u,t-1)) & if Follow(u,t) > Follow(u,t-1) + 1 \\ 0 & if |Follow(u,t) - Follow(u,t-1)| \leq 1 \\ -\log(Follow(u,t-1) - Follow(u,t)) & if Follow(u,t-1) > Follow(u,t) + 1 \end{cases} \tag{5.1}$$

Par conséquent, l'évolution de la popularité d'un compte sur une période donnée $[0, T]$ correspond à une série temporelle constituée du gain de popularité pour chaque unité de temps. Nous adaptons formellement la définition suivante pour l'évolution de la popularité.

Définition 7 (L'évolution de la popularité) *L'évolution de la popularité de l'utilisateur $u \in \mathcal{U}$ sur la période de temps $[0, T]$ est représentée par une série temporelle :*

$$\pi(u) =< gain(u,1), gain(u,2), \ldots, gain(u,T) > \tag{5.2}$$

Nous dénotons avec Π l'ensemble des évolutions de popularité pour les utilisateurs appartenant à $\mathcal{U}$.

Cette série temporelle brute est intéressante pour extraire des statistiques sur le jeu de données donné, mais n'est pas adaptée pour comparer les utilisateurs dans le but de les regrouper et de détecter des classes d'utilisateurs avec certains comportements spécifiques, ou, à l'opposé, pour identifier les utilisateurs qui ont un comportement divergeant. Pour atteindre ces objectifs, une approche traditionnelle (l'approche SAX [3, 71, 72]) consiste à coder la série temporelle à l'aide d'un ensemble limité de symboles. La figure 5.1 présente un exemple de décomposition d'une série temporelle. Nous présumons donc l'existence d'un alphabet de symboles Ω pour le codage et d'une fonction d'association $mapping : \mathbb{R} \to \Omega$. La définition de la taille de l'alphabet et de la fonction d'association est une tâche difficile qui doit fortement s'appuyer sur les propriétés du jeu de données étudié. La taille de l'alphabet

utilisé, pour les encodages, définit un niveau de raffinement pour la séquence encodée.

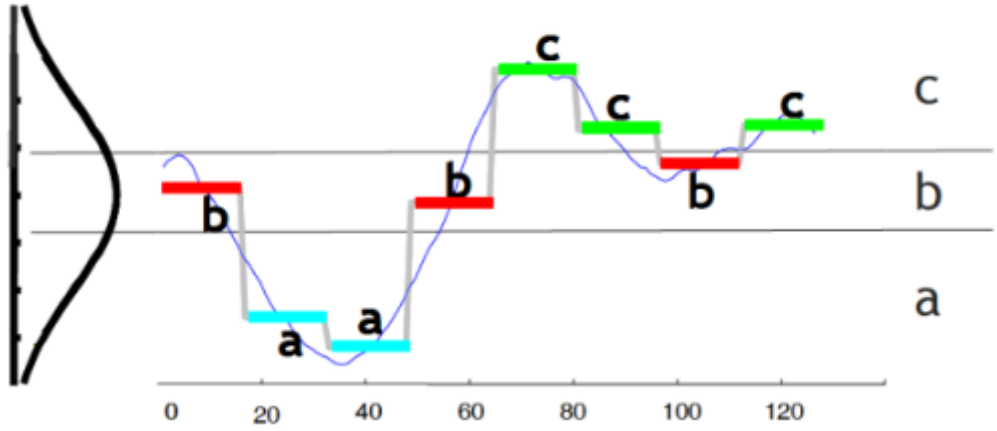

FIGURE 5.1 – Exemple d'utilisation de la méthode SAX pour représenter symboliquement une série temporelle. Source [3]

En effet, un grand alphabet apportera plus de précisions sur l'évolution de la popularité. Mais il réduit le nombre de similitudes entre les séquences extraites. A l'inverse, un petit alphabet conduira à l'extraction de nombreuses sous-séquences similaires, alors qu'elles correspondent en réalité à une évolution relativement différente. De même, la fonction d'association aura un impact important sur la détection de similarité entre les séquences. Lorsque nous associons trop de valeurs de gain différentes sur les mêmes symboles (alors que d'autres symboles correspondent à très peu de gains), il en résulte un problème similaire à l'utilisation d'un petit alphabet : des sous-séquences similaires qui sont détectées peuvent correspondre à des comportements très différents. Le choix de l'alphabet et de la fonction d'association a donc un impact important sur les résultats. Nous ne cherchons pas à approfondir ce problème, mais nous le prenons en considération lorsque nous proposons notre alphabet et notre fonction d'association dans nos analyses et nos expérimentations.

Sur la base de cette représentation, nous voulons étudier si certaines sous-séquences, que nous appellerons motifs ci-après, sont caractéristiques d'une classe de popularité. Si, de tels motifs existent, nous nous attendons à ce qu'ils nous permettent d'effectuer une détection précoce des comptes populaires émergents.

Supposons qu'une fonction d'extraction, *contains* : $2^{\Omega} \times \mathcal{S} \rightarrow \mathbb{N}$, avec 2^{Ω} désignant l'ensemble de

puissance de Ω, où $contains(x, y)$ renvoie 1 si la séquence y contient le mot x, et 0 autrement. Ensuite, nous adoptons la définition suivante pour un motif de popularité.

Définition 8 (Motif de popularité) *Considérons une valeur de taille donnée σ et supposons l'existence d'un seuil de pertinence Γ, un motif de popularité de taille σ est un mot $p \in \Omega^\sigma$ tel que :*

$$\frac{\sum_{s \in \mathcal{S}} contains(p, s)}{|\mathcal{S}|} \geq \Gamma \tag{5.3}$$

Le seuil de pertinence Γ permet de fixer un support minimal pour un motif. En d'autres termes, on ne garde que des motifs qui sont significatifs car ils sont suffisamment présents dans plusieurs séquences.

On désigne $\mathcal{S}_\Pi$ l'ensemble de tous les motifs de popularité pour un ensemble d'évolution de popularité Π. Les ensembles de motifs de popularité *exclusifs*, pour les restrictions aux ensembles d'évolution populaires, impopulaires et devenant populaires, sont notés respectivement $\mathcal{S}_{\Pi^+}$, $\mathcal{S}_{\Pi^-}$ et $\mathcal{S}_{\Pi^*}$. Par exclusif, nous entendons les motifs de popularité qui ne sont présents que dans la classe d'utilisateurs considérée.

5.2 L'analyse de la popularité

Cette section vise à analyser un véritable jeu de données Twitter pour vérifier l'existence de tels motifs d'évolution caractéristiques dans nos trois classes d'utilisateurs. Nous introduisons d'abord notre jeu de données avec ses principales caractéristiques, puis nous extrayons les motifs de popularité pour chaque classe et nous analysons ces différents ensembles de motifs.

5.2.1 Jeu de données

Pour construire notre jeu de données, nous utilisons le flux stream de l'API Twitter qui nous permet de collecter 1% de tous les tweets publiés sur la plateforme. Pour notre jeu de données, nous ne collectons que les méta-données des utilisateurs et nous sélectionnons les utilisateurs qui ont eu une activité suffisante ; c'est-à-dire au moins 1 tweet par semaine sur une période de 3 semaines, durant notre période d'observation de 36 semaines. Nous obtenons un ensemble de données composé d'environ 32,9 millions d'utilisateurs et 150 millions de tweets.

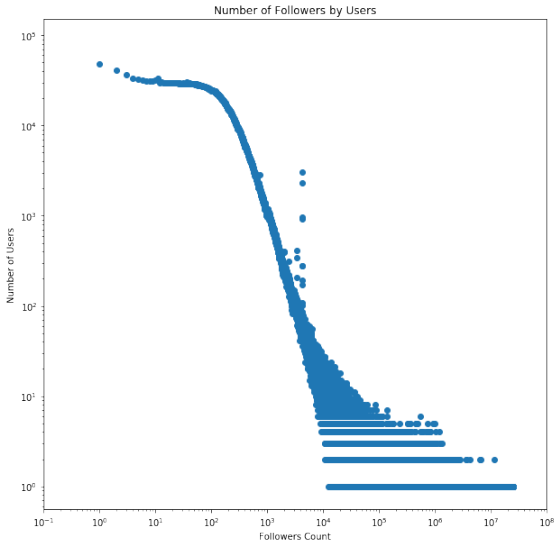

FIGURE 5.2 – Distribution du nombre d'abonnés

Ensuite, nous fixons les deux seuils φ et ε nécessaires pour déterminer nos trois groupes d'utilisateurs en fonction de leur popularité. Sur la base de la distribution des abonnés de notre ensemble de données illustré dans la figure 5.2, nous décidons arbitrairement qu'un compte avec moins de $\varphi = 400$ abonnés est considéré comme impopulaire, tandis qu'un compte avec plus de $\varepsilon = 2000$ est identifié comme un compte populaire. Ainsi, la classe des comptes devenant populaires correspond aux comptes qui possèdent moins de 400 abonnés au début de notre période d'observation et plus de 2000 à la fin. Nous rapportons dans le tableau 5.1 la taille de chaque classe de comptes de notre ensemble de données que nous étudierons ci-dessous.

Comme prévu, la plupart des comptes appartiennent à la classe non populaire et présentent une faible activité avec une moyenne de 3 ou 4 tweets sur la période d'observation. Les 2,1 millions de comptes populaires ont une activité plus importante, mais il existe un écart important entre les comptes comme cela est souligné par un écart-type élevé. En fait, cette classe est assez hétérogène avec, par exemple, des comptes d'agences de presse ou des marques avec une activité importante et, des personnalités populaires (acteurs, chanteurs, etc.) avec de nombreux abonnés mais peu de tweets. Les utilisateurs populaires ne sont pas aussi nombreux, mais ils ont une activité plus élevée. Cela peut, en quelque sorte, expliquer qu'ils deviennent populaires sur un sujet spécifique car ils publient davantage sur ce sujet, ce qui attire les abonnés.

Le tableau 5.2 montre le nombre d'abonnés pour chaque classe d'utilisateurs. Dans nos données,

TABLE 5.1 – Sous-ensembles de données pour chaque classe d'utilisateurs

	global	non-populaire	devenant pop.	populaire
# accounts	3.2289e+07	3.0106e+07	7.7364e+04	2.1056e+06
# tweets	1.4984e+08	9.2258e+07	3.6330e+06	5.3951e+07
tweets : mean	4.64	3.06	46.96	25.62

TABLE 5.2 – Nombre d'abonnés pour chaque classe d'utilisateurs

	global	non-populaire	devenant pop. at t=0	devenant pop. at t=T	populaire
mean	1.4280e+03	1.2737e+02	1.4059e+02	8.2574e+03	2.2431e+04
std	5.2869e+04	1.1283e+02	1.2261e+02	4.9651e+04	2.3761e+05
min	2.4000e+01	0.0000e+00	0.0000e+00	2.0010e+03	2.0010e+03
25%	5.2000e+01	2.8000e+01	2.7000e+01	2.4420e+03	2.9330e+03
50%	1.8800e+02	9.6000e+01	1.1050e+02	3.3160e+03	4.7430e+03
75%	4.8800e+02	2.0800e+02	2.4000e+02	5.9640e+03	1.0699e+04
85%	8.1900e+02	2.7000e+02	3.0100e+02	9.4710e+03	1.8553e+04
90%	1.2040e+03	3.0700e+02	3.3300e+02	1.2800e+04	2.8601e+04
max	7.7129e+07	3.9900e+02	3.9900e+02	6.7735e+06	7.7129e+07

le nombre moyen d'abonnés est de 1428, ce qui est plus que la dernière estimation présentée ci-dessus, en raison de notre seuil de 3 semaines. Les utilisateurs possédant une activité inférieure à ce seuil ne sont pas retenus et sont supprimés du jeu de données. Ces comptes représentent des utilisateurs avec peu d'abonnés et une activité extrêmement faible (d'autant plus que nous utilisons l'API Twitter avec seulement 1% des messages). Nous constatons que les comptes non populaires ont un nombre assez stable de followers. En revanche, nous constatons que les comptes devenant populaires et ceux déjà populaires ont généralement entre 2.10^3 et 3.10^4 abonnés, le dernier décile d'utilisateurs atteint respectivement des millions et des dizaines de millions d'abonnés, avec un écart plus important pour les comptes populaires.

5.2.2 L'extraction de motifs

Afin d'effectuer notre extraction de motifs, nous devons d'abord définir la taille de l'alphabet Ω et déterminer la fonction de *mapping* utilisée pour le codage d'évolution de popularité. Comme expliqué ci-dessus, la fonction d'association doit, autant que faire se peut, répartir uniformément les valeurs de gain sur les différents symboles de Ω.

Nous comparons différentes tailles d'alphabet pour Ω et nous choisissons finalement $|\Omega| = 8$. Comme expliqué, dans la section 5.1.2, un petit alphabet ne nous permet pas d'extraire des motifs

symbol	A	B	C	D	E	F	G	H
range	$]-\infty,-2]$	$[-2,-0.7]$	$[-0.7,0.7]$	$[0.7,1.6]$	$[1.6,2]$	$[2,2.7]$	$[2.7,3]$	$[3,\infty[$

TABLE 5.3 – Table d'encodage des valeurs de gain

significatifs car ils couvrent des comportements très distincts, alors qu'un grand alphabet fournit des motifs très précis mais, ces motifs ne correspondent qu'à un très petit nombre de séquences.

Nous rapportons dans le tableau 5.3 notre implémentation de la fonction d'association pour les valeurs de gain calculées selon la définition 6.

Ensuite, nous calculons le gain d'abonnés pour chaque semaine de la période d'observation de 36 semaines et nous appliquons la fonction d'association pour l'encodage. En raison du faible taux de données de l'API Twitter (seulement 1% des tweets) et du comportement de publication non uniforme des utilisateurs, plusieurs valeurs manquent dans de nombreuses séquences d'évolution de popularité. Par conséquent, nous décidons d'appliquer une interpolation linéaire pour combler les petits espaces de deux valeurs manquantes au maximum. Les utilisateurs, dont les séquences présentent des intervalles vident de plus de 2 symboles, sont supprimés de notre jeu de données. Le tableau 5.4 présente les distributions de symboles pour les différents jeux de données. Nous observons que les comptes non populaires ont, comme prévu, une croissance nulle ou très modérée avec 86,7% de symbole C, soit un gain ou une perte de maximum 5 abonnés. Nous notons qu'ils ont également très peu de pertes d'abonnés (0,6% du symbole B ce qui représente une perte de 5 à 100 abonnés) et, sont donc très stables. Pour les comptes populaires, l'évolution est plus variée : ils peuvent montrer une croissance significative, stagner ou avoir une baisse significative. Les baisses importantes peuvent s'expliquer de plusieurs façons : une impopularité soudaine due, par exemple, à une prise de décision discutable, ou encore une identification comme un faux compte ou un compte ayant acheté des followers. Les faux followers ont tendance à se faire supprimer par Twitter, ce qui supprime leurs abonnements. Enfin, les comptes qui deviennent populaires ont généralement des périodes de croissance significative assez longues, ce qui explique notre observation d'environ 49% des symboles F, G et H, soit un gain de plus de 100 followers.

Les algorithmes d'extraction de motifs traditionnels renvoient un ensemble de symboles qui ne sont pas nécessairement fermés. Pour notre modèle d'appariement, nous devons extraire uniquement les motifs contenant des symboles qui se suivent sans interruption. Pour exécuter un algorithme d'ex-

TABLE 5.4 – Distribution des symboles

symbole	Devenant Pop.	Populaire	Non-populaire	Global
A	3.94%	3.39%	0.0%	1.39%
B	7.20%	16.50%	0.60%	4.94%
C	6.74%	20.27%	86.7%	59.14%
D	14.72%	29.88%	12.30%	16.05%
E	18.36%	12.02%	0.33%	5.91%
F	36.55%	12.08%	0.05%	9.15%
G	6.64%	2.58%	0.0%	1.72%
H	5.82%	3.24%	0.0%	1.69%

traction de motifs, qui ne prend que des ensembles d'éléments fréquents de symboles consécutifs, nous effectuons une extraction de motifs sur nos 3 ensembles de données en utilisant une approche de fenêtres coulissantes et en rapportant toutes les séquences de k-symboles rencontrés. Nous décidons de ne conserver que les motifs avec un support supérieur à 1%. Nous rapportons dans le tableau 5.5 le nombre de motifs que nous avons extraits. Nous observons que la classe non populaire est caractérisée par un faible nombre de motifs. Il y a deux raisons à ce résultat : premièrement, les utilisateurs non populaires présentent un nombre d'abonnés qui ne varie pas de manière importante, de sorte que, la plupart des symboles de leur évolution de popularité sont des symboles C ou D. De plus, la classe des utilisateurs non populaires est très grande, donc un support minimum de 1% implique que ce modèle est présent dans un très grand nombre de séquences de gain de popularité. Avec un support de 0,5%, le nombre de motifs est de 286, 1051 et 2782 pour respectivement les longueurs 3, 4 et 5. Les classes d'utilisateurs populaires et devenant populaires ont environ le même nombre de motifs, soit entre 100 et 200. On peut observer que la taille des motifs extraits a un double impact sur le nombre de motifs. En effet, un motif de taille k avec un support supérieur à 1% peut fournir potentiellement deux motifs ou plus avec $k+1$ symboles et avec un support supérieur à 1%. Mais d'un autre côté, il peut fournir des motifs avec un support inférieur à 1%. Cela explique pourquoi le nombre de motifs augmente de 182 pour la taille 3 à 194 pour la taille 4 pour le jeu de données des comptes devenant populaires, puis diminue à 165 pour la taille 5.

Lorsque nous comparons les motifs extraits dans la classe devenant populaires à ceux des autres classes, nous constatons que plusieurs motifs sont présents dans plusieurs classes. Néanmoins, nous avons environ 70 motifs exclusifs à la classe des utilisateurs devenant populaires : ceux que nous utiliserons pour notre détection.

TABLE 5.5 – Nombre de motifs extraits avec un support minimum de 1%

taille des motifs	taille 3	taille 4	taille 5
non-populaire	11	18	27
populaire	123	144	154
devenant populaire	182	194	165
devenant populaire - (non-populaire + populaire)	70	75	69

Nous rapportons, dans la figure 5.3, le support des 100 premiers motifs les plus fréquents que nous avons extraits pour chaque jeu de données. Nous observons tout d'abord qu'indépendamment du jeu de données ou de la taille des motifs extraits, la distribution de la fréquence des motifs suit une loi de puissance. Les motifs pour la classe des comptes devenant populaires ont un support plus important que ceux des autres classes. Pour une taille 3, le motif le plus fréquent est présent parmi 55% des séquences, le dixième plus fréquent parmi 15% et le cinquantième est toujours présent dans environ 5% des séquences. Il en résulte que l'évolution de la popularité pour les utilisateurs devenant populaires est caractérisée par plusieurs dizaines de motifs qui peuvent être utilisés, par conséquent, pour identifier les utilisateurs de cette classe. Le support des motifs populaires est moins important mais reste significatif : le motif le plus fréquent est présent dans 30% des séquences, le dixième plus fréquent dans 8% et le cinquantième est toujours présent dans environ 3% des séquences. Pour cette classe, on observe, par conséquent, l'existence d'un grand nombre de motifs caractéristiques. Cependant, comme le montre le tableau 5.4, la distribution des symboles est moins biaisée que celle des utilisateurs devenant populaires, ce qui conduit à plus de motifs mais avec moins de supports. Enfin, la classe des comptes non populaires se caractérise par un petit nombre de motifs avec un support important : le motif le plus fréquent est présent dans 30% des séquences, le dixième plus fréquent dans 1%. Enfin, nous observons une longue queue de motifs avec un support inférieur à 1%.

Deux paramètres expliquent cette observation : premièrement, les séquences des utilisateurs non populaires sont plutôt courtes car ils ont une faible activité, ainsi que le fait que nous arrêtons les séquences d'évolution s'il manque deux symboles ou plus (ce qui est assez rare chez les utilisateurs populaires ou devenant populaires). Deuxièmement, leur nombre d'abonnés est assez stable, comme on peut le voir dans le tableau 5.4 avec 86,7% de symbole C. Les résultats pour une taille 4 de motifs sont similaires, sauf que nous observons que les courbes suivant une loi de puissance sont un peu lissées par rapport à la taille 3. En fait, comme expliqué ci-dessus pour le tableau 5.5, nous avons pour les différentes classes, moins de motifs avec un support important mais plus avec un support moyen.

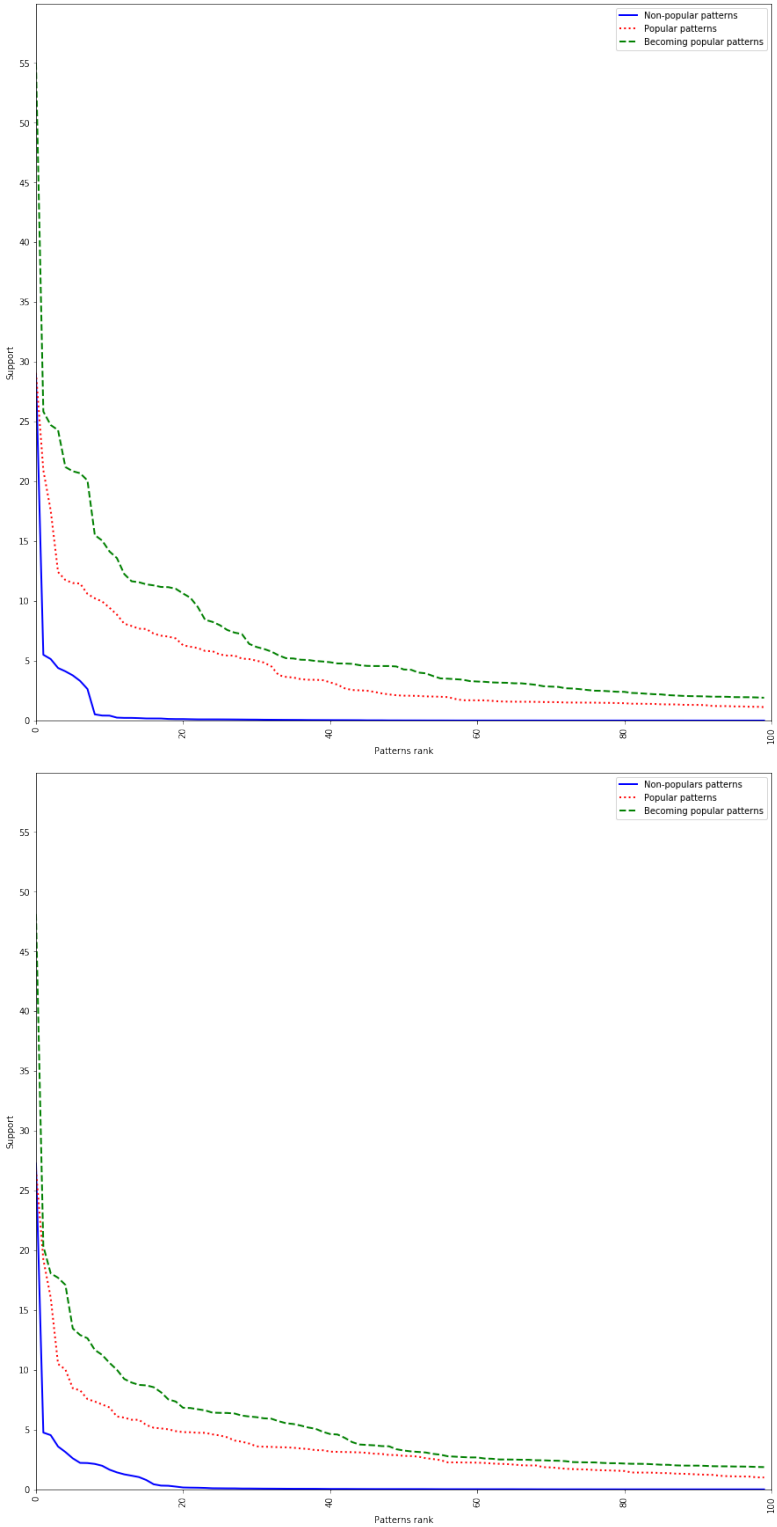

FIGURE 5.3 – Support des motifs pour une taille égale à 3 (à gauche) et une taille égale à 4 (à droite) pour les différentes classes d'utilisateurs (Bleu : Non populaire, Rouge : Populaire, Vert : Devenant Populaire

5.3 L'utilisation de motifs pour détecter de futurs comptes populaires

Une fois que nous avons identifié les différents ensembles de motifs exclusifs de popularité pour chacun des ensembles d'évolution : populaire, non populaire et devenant populaire, que nous noterons respectivement $\mathcal{S}_{\Pi^+}$, $\mathcal{S}_{\Pi^-}$ et $\mathcal{S}_{\Pi^*}$, nous avons l'intention de les utiliser pour identifier les utilisateurs devenant populaires avant qu'ils n'atteignent le seuil de popularité.

5.3.1 Matching de motifs de comptes populaires

Notre objectif est de tester le matching de n'importe quel motif $p \in \mathcal{S}_{\Pi^*}$ avec n'importe quelle séquence d'évolution de popularité $s \in \mathcal{S}$ chaque fois qu'elle augmente avec un nouveau symbole.

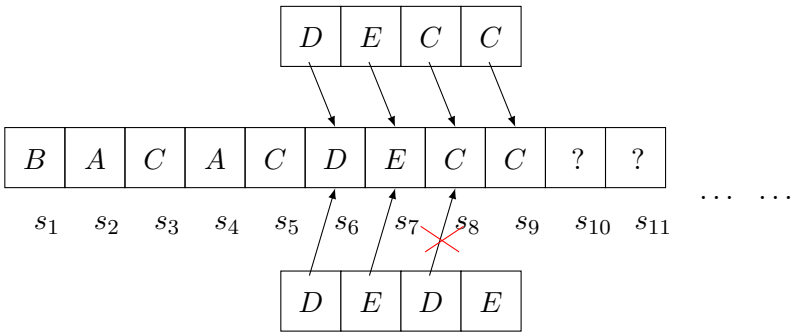

FIGURE 5.4 – Exemple de tentative de matching pour un motif DECC et DEDE sur une séquence d'évolution de la popularité

Example 1 *La figure 5.4 illustre un exemple de tentative de matching. Pour notre utilisateur, nous rapportons un nouveau symbole C à la semaine* 9 *correspondant à son gain de popularité entre les semaines* 8 *et* 9*. Nous devons alors essayer de faire correspondre chaque motif de notre ensemble de motifs d'utilisateurs devenant populaires avec le suffixe de la séquence d'évolution de la popularité. Le motif, DEDE ne correspond pas au suffixe, contrairement au modèle DECC. Nous rapportons donc un match et nous considérons l'utilisateur correspondant à cette séquence d'évolution de popularité comme un éventuel futur utilisateur populaire.*

Donc, fondamentalement, notre problème est un problème de correspondance multi-flux multi-motifs. La différence avec les solutions traditionnelles de correspondance de motifs est que nous devons traiter des centaines de millions d'utilisateurs et des centaines de motifs, ce qui soulève d'importants

problèmes de complexité et d'extensibilité. Notre objectif, dans ce qui suit, est de proposer une structure pour faire du matching multi-motifs plus rapidement sur un très grand ensemble de flux.

5.3.2 L'index H^2M

Pour effectuer le matching d'un ensemble $\mathcal{S}_\Pi$ de motifs sur une séquence π de symboles, plusieurs approches sont proposées dans la littérature. Les plus efficaces reposent sur l'automate à états finis (FSA Finite State Automata). Cependant, un FSA traditionnel présente des boucles qui vérifient pour chaque état atteint, s'il correspond à l'état final d'un motif donné ou non. Pour limiter ce nombre de tests, nous proposons de s'appuyer sur une représentation Trie (arbre préfixe), c'est-à-dire un automate fini déterministe en forme d'arbre. Puisque nous faisons l'hypothèse que nos modèles de popularité ont une taille fixe σ, cela signifie que tous (1) les chemins de la racine à une feuille ont une longueur de σ et (2) seules les feuilles correspondent au symbole final d'un motif de popularité.

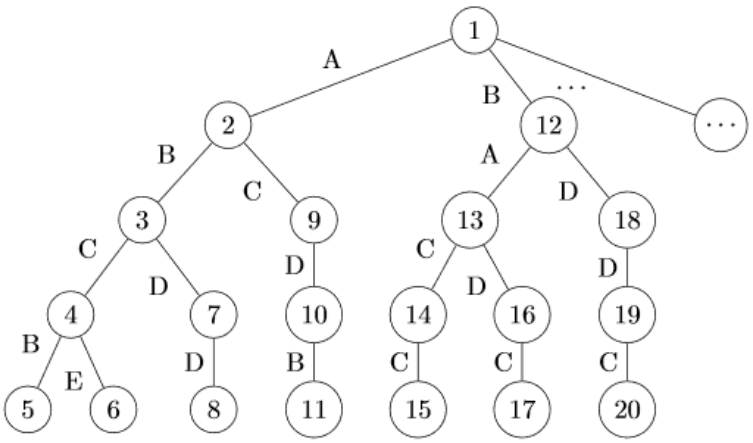

FIGURE 5.5 – Structure en forme d'arbre pour $D_{pop} = \{ABCB, ABCE, ABDD, ACDB, BACC, BADC, BDDC\}$

Example 2 *La figure 5.5 est un exemple de l'arborescence pour l'ensemble des motifs ABCB, ABCE, ABDD, ACDB, BACC, BADC, BDDC. Ici $\sigma = 4$, nous pouvons vérifier que tous les chemins vers une feuille ont une longueur de 4 et que chaque feuille correspond à un motif.*

Cette taille fixe des motifs nous permet d'envisager une fenêtre glissante sur les différentes séquences avec uniquement les derniers symboles σ qui sont utilisés pour les tentatives de matching.

Puisque nous considérons des applications avec des centaines de millions d'utilisateurs, nous devons

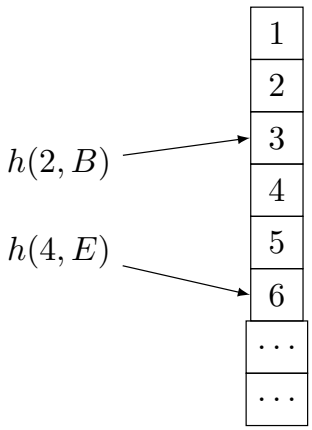

FIGURE 5.6 – Notre implémentation basée sur du hashage

évaluer les transitions de manière très efficace. Ainsi, nous proposons de choisir l'implémentation basée sur le hachage pour notre structure arborescente de motifs. Donc, formellement, notre arbre de motifs est défini grâce à notre *Pattern-Tree Hash index* (PTH-index) comme :

Définition 9 (Pattern-Tree Hash index) *Un* Pattern-Tree Hash index *PTH est défini sur un ensemble de motifs $\mathcal{S}_\Pi$ comme un couple (V, h_{trie}) où V est l'ensemblde de noeuds $v = (id, isLeaf) \in V$ avec id correspondant à l'id du noeud et $isLeaf$ un booléen mis à true quand le noeud est une feuille, et $h_{trie} : V \times \Omega \to V$ est la fonction de hash qui représente les arêtes respectant les propriétés suivantes :*

i) h_trie injectif, donc chaque noeud ne peut avoir qu'un seul parent (à l'exception du noeud racine),

ii) si $\forall v \in V, v.isLeaf = true \Rightarrow \exists(x_1, x_2, \ldots, x_\sigma) \in \Omega^\sigma$,

$h_{trie}(h_{trie}(\ldots(h_{trie}(root, x_1), x_2), \ldots, x_\sigma) = v \wedge x_1.x_2.\ldots.x_\sigma \in \mathcal{S}_\Pi$

Example 3 *La figure 5.6 représente la structure basée sur le hachage correspondant à la structure arborescente de la figure 5.5. Par exemple avec le symbole E, le nœud dont l'identifiant est* 4 *mène au nœud dont l'identifiant est* 6.

Pour tout symbole entrant pour une séquence donnée π, on utilise cet arbre et on détermine les nouvelles positions atteintes dans celui-ci. La proposition suivante détermine le nombre de positions dans l'arbre que nous devons stocker pour tout utilisateur.

Proposition 1 (Nombre de positions stockées) *Puisque la profondeur de l'arbre correspond à la longueur des motifs, σ, les différents suffixes de longueur compris entre $[1, \sigma]$ doivent être pris en compte*

lorsqu'un nouveau symbole est ajouté et, chaque suffixe pourrait atteindre une position dans l'arbre. Donc, à tout moment nous avons :

- *enregistré pour la prochaine tentative de matching, les différentes positions atteintes avec les suffixes de longueur compris entre* $[1, \sigma - 1]$,
- *rapporté potentiellement une correspondance en atteignant une feuille. Par conséquent, l'espace requis pour stocker les informations utilisateur est* $O(\sigma)$.

Pour récupérer efficacement les différentes positions d'un utilisateur dans l'arbre, nous proposons de s'appuyer sur une seconde structure de hachage. Considérez donc $\mathcal{I}$ l'ensemble des identificateurs d'utilisateur. Nous définissons ensuite notre index de positionnement d'automate ($AP - index$) qui se compose d'entrées.

Définition 10 *Soit* $id \in \mathcal{I}$ *un identifiant d'utilisateur et* V *désigne l'ensemble des nœuds ids de l'arbre. L'entrée indexant* P, *notée* $P(id)$, *est un couple* (id, pos) *avec* $pos \in 2^{V^\sigma}$ *(l'ensemble de puissance de* V^σ*) un ensemble de positions dans l'automate.*

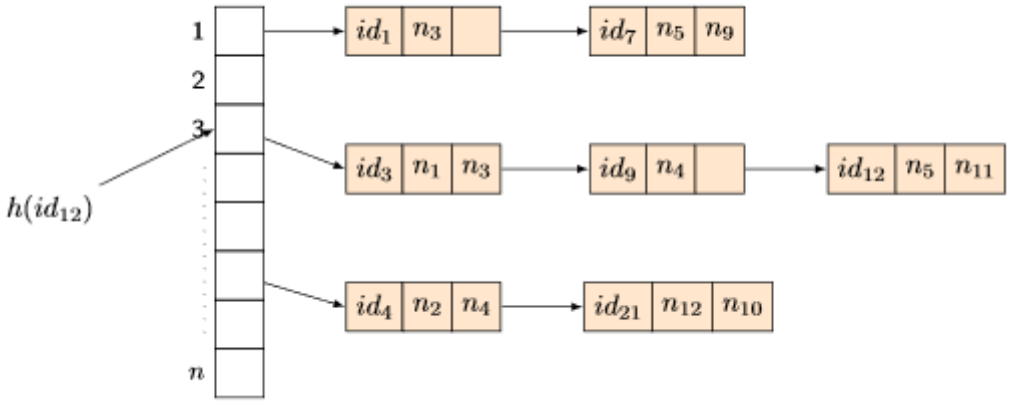

FIGURE 5.7 – Notre implémentation H^2M basée sur du hashage

L'indexation est «dense», c'est-à-dire, que chaque séquence d'évolution de popularité dans la base de données est indexée, dès qu'un utilisateur apparaît dans le système, et par une entrée différente. $AP-Index$ est un fichier de hachage, noté $AP[0..L-1]$, avec une longueur de répertoire L (Figure 5.7). Construire un fichier de hachage avec une entrée de hachage pour chaque utilisateurs conduira à un index extrêmement volumineux, car il nécessite des blocs de mémoire $|\mathcal{S}|$ et donc la structure d'index

ne peut pas tenir en mémoire. Par conséquent, les éléments de AP se réfèrent à des buckets ou des lignes, contenant chacun une liste d'entrées. Chaque $AP[i]$ contient l'adresse du i-ème bucket. Étant donné que les entrées ont une taille similaire, c'est-à-dire, un ensemble user id et de positions avec un nombre d'éléments compris entre 1 et $\sigma - 1$, nous pouvons stocker dans un bloc de taille B entre $B/(|id| + (\sigma - 1).|p|)$ et $B/(|id| + |p|)$ entrées ($|id|$ and $|p|$ désignent respectivement la taille d'un identifiant d'utilisateur et d'un noeud id d'automate). Nous définissons donc la taille de la liste en conséquences. Nous considérons, dans ce qui suit, l'approche pessimiste où toutes les entrées à l'intérieur d'un bucket ont $\sigma - 1$ positions stockées. Nous supposons également que nous avons $|\mathcal{S}|$ utilisateurs à indexer. Nous expliquons ci-dessous comment gérer la dynamicité du système, avec des utilisateurs qui rejoignent ou quittent fréquemment. Avec ces paramètres, nous pouvons définir la valeur de L :

$$L = |\mathcal{S}| \times \frac{|id| + (\sigma - 1).|p|}{B} \tag{5.4}$$

Des statistiques sur la taille de l'entrée (autrement dit, quel est le nombre moyen de positions d'automates stockées pour un utilisateur) pourraient permettre de proposer une valeur L avec un gain d'espace plus élevé. Dans l'ensemble, la structure $AP - Index$ est similaire à une liste de publication dans un fichier inversé. Étant donné que les clés utilisées pour le hachage sont des identifiants d'utilisateurs, il est facile de concevoir une fonction de hashage qui distribue uniformément les entrées dans les compartiments, du moins, lorsque l'index est construit. Pour gérer la dynamicité de l'ensemble des utilisateurs, nous pourrions avoir plusieurs stratégies qui pourraient d'ailleurs être combinées. Tout d'abord, nous pourrions adopter la stratégie utilisée dans les systèmes de base de données pour stocker les données, c'est-à-dire, ne pas remplir le datablock (le paramètre *PCT-free*). Ainsi, nous pourrions, par exemple, choisir une valeur L plus élevée lors de la création de l'index et avoir par conséquent de l'espace libre dans chaque bloc pour ajouter de nouvelles entrées. De plus, puisque nous utilisons un fichier de hachage, les lignes doivent avoir une méthode de résolution de collision telle que *classical separate chaining* qui utilise des pointeurs vers un espace de débordement [73]. Une telle technique permet une croissance modérée, mais si nous avons besoin d'accommoder une croissance importante, nous avons besoin d'une méthode de hachage dynamique telle que le hachage linéaire [74]. La combinaison de nos deux structures basées sur le hachage pour un matching efficace, $PTR - index$ et $AP - index$, compose notre proposition que nous nommons H^2M -index.

TABLE 5.6 – Qualité de la détection

	precision	recall	F1
Global dataset	0.7260	0.7604	0.7428
Removing popular users	0.9972	0.7604	0.8629

5.4 Expériences et résultats

Toutes les expériences ont été réalisées sur une machine dédiée avec 8x IntelR© XeonR© Processor E7-4830 v2 (80 cœurs) et 512 Go de RAM. Nous avons choisi Python pour nos développements. Pour valider notre approche, nous évaluons d'abord la qualité de notre détection, puis nous avons comparé les performances de notre structure de matching avec d'autres implémentations.

5.4.1 Qualité de notre approche de détection

Pour estimer la qualité de notre approche, nous avons divisé notre jeu de données global de 32 millions d'utilisateurs en un jeu d'entraînement avec 80% des utilisateurs, et un jeu de test avec les 20% restant. Nous divisons notre jeu de test en 3 groupes d'utilisateurs en fonction de leur évolution de popularité comme expliqué dans la section 5.2 et nous effectuons notre processus d'extraction de motifs sur chacun de ces jeux de données. Les tableaux 5.1, 5.2 et 5.5 décrivent les caractéristiques de ces jeux de données ainsi que leur nombre de motifs.

Ensuite, nous essayons de faire correspondre les motifs sur notre jeu de données de test pour détecter les utilisateurs devenant populaires. Nous rapportons les résultats dans le tableau 5.6. Nous observons que nous avons une précision globale de $0,7260$. Cependant, cette précision peut être largement améliorée par un pré-traitement rapide ou un post-traitement en écartant les utilisateurs populaires. En effet, nous pouvons, à tout moment, arrêter le suivi d'un utilisateur identifié comme populaire, c'est-à-dire, ceux ayant à tout moment plus de 4000 followers. Lors de la suppression de ces utilisateurs, nous atteignons une précision de $0,9972$. Le rappel atteint $0,7604$, donc une bonne valeur F1 de $0,8629$. Nous obtenons un rappel plus élevé lors de la réduction du support minimal au moment de la sélection du motif. Si nous conservons des modèles avec un support inférieur, nous obtenons un rappel de $0,85$, mais la précision tombe à $0,75$ car ces modèles sont moins caractéristiques de la classe associée et, par conséquent, certaines séquences appartenant à d'autres classes peuvent également correspondre. Puisque notre objectif est de détecter les futurs utilisateurs populaires avant qu'ils ne

deviennent populaires, à chaque fois qu'un matching se produit, nous mesurons le nombre de semaines entre la semaine du matching et la semaine au cours de laquelle ils deviennent réellement populaires. Nous rapportons le ratio de futurs utilisateurs populaires détectés par rapport au nombre de semaines avant que cet utilisateur ne soit réellement détecté populaire sur la figure 5.8. Nous observons que dans 80% des cas, notre approche nous permet de détecter un utilisateur populaire au moins 1 mois avant qu'ils ne deviennent réellement populaire, et dans 60% des cas, au moins 2 mois avant. Ce taux est toujours d'environ 40% pour la détection au moins 3 mois à l'avance. Cela confirme l'intérêt de notre démarche et son efficacité.

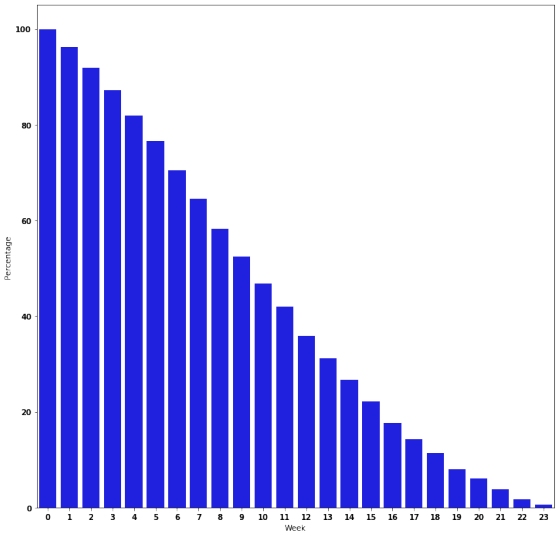

FIGURE 5.8 – Nombre de semaines avant qu'un utilisateur populaire soit détecté

5.4.2 Performances et extensibilité

Enfin, nous effectuons des expériences pour étudier l'extensibilité de notre index et nous le comparons aux structures de matching existantes. En tant que concurrents, nous implémentons :

— **SimpleTree** : implémentation d'un arbre simple provenant d'une bibliothèque de Python avec la possibilité de rechercher si une sous-séquence est dans l'arbre. Pour chaque utilisateur, nous stockons dans une hashmap le $\sigma - 1$ dernier symbole lu (σ est la taille du motif) .

— **FSA** : nous avons utilisé Automata-lib [1] qui implémente les structures et les algorithmes pour les automates finis en Python.

— H^2M : notre solution présentée ci-dessus.

1. Automata-lib 3.1.0 : https://pypi.org/project/automata-lib/

Nous générons en fonction de la distribution des symboles observés dans notre jeu de données réel (voir tableau 5.4) des jeux de données de respectivement 1 million, 10 millions et 50 millions de séquences d'évolution de popularité de 52 symboles (pour simuler 1 an). Nous mesurons le temps pour évaluer un symbole entrant pour un utilisateur avec les différentes structures, ainsi que l'utilisation de mémoire pour stocker les informations des 1, 10 et 50 millions d'utilisateurs. Nous rapportons dans le tableau 5.7 nos résultats. Nous observons, tout d'abord, que l'indice H^2M permet d'obtenir les meilleures performances pour le matching puisqu'un nouveau symbole entrant peut être traité en $0,2ms$ lors de la gestion d'un million d'utilisateurs, donc un gain de 90% et 250% comparé respectivement à *SimpleTree* et *FSA*. Ce temps de matching augmente avec la taille du motif. On constate une augmentation de 40% du temps de matching pour toutes les structures. Le raisonnement est que nous avons une sous-séquence de 4 symboles dans le cas du *SimpleTree* ou *FSA*, ou 3 positions dans notre structure, à considérer au lieu de 3. Concernant le nombre d'utilisateurs à gérer, nous observons un temps de matching constant comme prévu, puisque nous adoptons une extension dynamique du hashmap pour les différentes structures afin de conserver les états actuels de chaque utilisateur pour éviter les collisions. Par conséquent, le traitement d'un symbole entrant est constant. Nous pouvons également observer que notre implémentation nécessite un espace mémoire similaire à celui de *FSA* tout en économisant 24% d'espace par rapport à *SimpleTree*. Le raisonnement est que nous devons conserver les derniers symboles $\sigma-1$ pour tout utilisateur pour le *SimpleTree* alors que nous ne gardons que 1 à $\sigma-1$ états atteignables pour un utilisateur avec H^2M . Le gain en mémoire augmente avec la taille du motif, et nous atteignons 44% de gain pour les motifs de taille 4.

TABLE 5.7 – Performances of the matching

	Patterns size :	Size 3		Size 4	
Type	Nb of users	Time (ms)	RAM (MB)	Time (ms)	RAM (MB)
SimpleTree	1M	0.57	275.05	0.82	319.54
	10M	0.58	2526,00	0.81	3238.62
	50M	0.56	15067.56	0.80	16161.63
FSA	1M	1.05	205.86	1.53	227.55
	10M	1.05	2152.37	1.50	2199.18
	50M	1.03	11570.12	1.48	11962.11
H^2M	1M	0.30	214.97	0.41	222.32
	10M	0.30	2120.89	0.42	2197.58
	50M	0.31	11568.76	0.42	11954.67

5.5 Conclusion

Dans ce chapitre, nous avons présenté une solution qui permet de détecter précocement de futurs utilisateurs populaires basée sur une caractérisation des utilisateurs qui deviennent populaires en utilisant des motifs d'évolution de popularité. Nos analyses montrent l'existence de ces motifs et nos expériences confirment qu'ils permettent d'identifier et détecter de futurs utilisateurs populaires plusieurs semaines avant qu'ils ne soient réellement populaires. Nous proposons également une structure permettant d'étendre le matching à des millions d'utilisateurs, tout en considérant un flux de données constant. Notre modèle est facilement parallélisable pour gagner en temps de traitement et pouvoir faire face à d'importants volumes de données.

Dans le chapitre suivant, nous présentons une méthode permettant d'identifier des groupes d'utilisateurs en utilisant les différentes interactions qu'ils génèrent au sein du réseau social Twitter.

Chapitre 6

Classification d'utilisateurs basée sur les interactions

Dans ce chapitre, nous allons présenter une approche de classification d'utilisateurs de Twitter en utilisant les interactions qu'ils génèrent sur le réseau. L'hypothèse est, que selon le profil d'un utilisateur, les types d'interactions et la quantité d'interactions venant de ses abonnés seront différents. Dans une première partie, nous présenterons les jeux de données et leurs caractéristiques. Dans une seconde partie, nous introduirons les notations et notre modèle de données. Dans une troisième section, nous présenterons une méthode pour calculer un score global d'interactions des utilisateurs. Enfin, nous verrons un score d'interactions spécifique à chaque type d'actions et les résultats obtenus.

6.1 Présentation des jeux de données

Pour ces travaux, nous avons collecté deux jeux de données en utilisant l'API Twitter comme dans les travaux précédents. Nous avons collecté des tweets sur une période de cinq mois. Ces deux jeux de données ont été collectés en utilisant des mots-clés pour filtrer les tweets et, par la même occasion, créer deux communautés d'individus. Le premier, est un jeu de données qui a rassemblé des messages parlant du COVID, et le second jeu est composé de tweets à propos de la NBA (National Basketball Association). Notre jeu de données global contient approximativement 24 millions de tweets

6.1.1 Jeu de données NBA

Le jeu de données sur la NBA est composé de 5 millions de tweets, dans lesquels nous avons identifier 4.9 millions d'interactions (`Retweet`, `Quote`, `Reply` et `Mention`). Il est important de souligner que tous

Table 6.1 – NBA Dataset : Statistiques

	Followers	Friends	# Tweets	# Quotes	# Retweets	# Mentions	# Replies
Value Count	882494	882494	1935124	561041	472376	644758	211985
Mean	4000.47	1096.57	2.49	1.25	1.93	2.27	0.45
Median	328	445	1	1	1	1	0
Std Dev	200434.26	4632.68	10.78	3.21	6.69	7.66	2.32
Min	0	0	1	0	0	0	0
Max	87244738	1480293	9171	1394	1478	2002	748

les tweets ne correspondent pas nécessairement à une interaction, mais certains peuvent contenir plusieurs interactions (par exemple les retweets et les quotes).

6.1.2 Jeu de données COVID

Le jeu de données sur le COVID est composé de 21 millions de tweets dans lesquels nous avons identifié 17 millions d'interactions.

6.1.3 Les graphes d'interactions

Pour représenter nos jeux de données sous la forme de graphes d'interactions pour notre expérience, nous avons sélectionné seulement les utilisateurs ayant suscité au moins deux interactions. Puis, nous avons construit le graphes en gardant, la plus grande composant connexe. Pour cela nous avons utilisé la librairie Python NetworkX. Ce pré-traitement permet d'éviter d'avoir de petits graphes avec des noeuds isolés ce qui peut réduire le score global de PageRank. Le plus grand graphe connexe extrait pour le jeu de données NBA contient 494 noeuds, les statistiques de ces noeuds sont présentés dans le tableau 6.1. Pour le jeu de données COVID, nous avons construit un graphe connexe de 2 789 316 utilisateurs dont les statistiques sont présentées dans le tableau 6.2.

6.2 Le modèle de données

Dans cette section, nous introduisons nos notations et notre modèle de données. Nous considérons la plate-forme Twitter et son graphe d'interactions sous-jacent $\mathcal{G} = (\mathcal{U}, \mathcal{E})$ où $\mathcal{U}$ désigne l'ensemble de noeuds, *c.a.d* les utilisateurs, $\mathcal{E} \subseteq \mathcal{U} \times \mathcal{U}$ est l'ensemble des arêtes, tel que $(u_1, u_2) \in \mathcal{E}$ signifie que l'utilisateur u_2 a effectué une action sur les tweets de l'utilisateur u_1. Nous dénotons $\mathcal{A}$ l'ensemble

TABLE 6.2 – COVID Dataset : Statistiques

	Followers	Friends	# Tweets	# Quotes	# Retweets	# Mentions	# Replies
Value Count	2789316	2789316	6278280	1783237	1699905	1945609	588131
Mean	3832.08	1128.18	2.64	1.31	2.49	2.01	0.37
Median	287	435	8348	1	1	1	0
Std Dev	128751.36	4644.14	8.27	3.09	8.76	6.74	2.37
Min	0	0	1	0	0	0	0
Max	85941911	1907480	8768	929	7910	2823	1480

des interactions possible qu'un utilisateur peut effectuer sur le tweet d'un autre utilisateur. Dans ce qui suit, nous considérons que $\mathcal{A} = \{a_{rt}, a_{qt}, a_{rp}, a_{mt}\}$, qui correspond respectivement aux actions de `Retweet`, `Quote`, `Reply` et `Mention`. La restriction du graphe d'interaction $\mathcal{G}$ à une action donnée $a \in \mathcal{A}$ noté $\mathcal{G}_a$ le graphe $\mathcal{G}_a = (\mathcal{U}_a, \mathcal{E}_a)$ avec $\mathcal{U}_a \subseteq \mathcal{U}$ et $\mathcal{E}_a \subseteq \mathcal{E}$ tel que $(u_1, u_2) \in \mathcal{E}_a$ si u_2 effectue une interaction du type a sur le tweet de l'utilisateur u_1. De toute évidence, toutes les arêtes d'un graphe d'interaction ne représentent pas le même niveau d'interaction entre les utilisateurs. Certaines interactions peuvent se produire fréquemment, tandis que d'autres peuvent se produire rarement. Pour capturer cette notion, nous définissons un poids d'interaction ω comme suit :

Définition 11 (Poids d'interaction global) *Le poids d'interaction global ω est une fonction $\omega : \mathcal{E} \to \mathbb{R}$ qui prend en considération toutes les interactions entre deux utilisateurs.*

Définition 12 (Poids d'interaction spécifique) *Le poids d'interaction spécifique ω pour une action $a \in \mathcal{A}$ est une fonction $\omega : \mathcal{E} \times \mathcal{A} \to \mathbb{R}$. Ce score est principalement basé sur les interactions d'un type $a \in \mathcal{A}$ et donne mois ou pas d'importance aux autres types d'interactions.*

Enfin, nous supposons l'existence d'une fonction $count : \mathcal{E} \times \mathcal{A} \to \mathbb{N}$, tel que $count((u_1, u_2), a)$ est le nombre d'interactions de type a que u_2 effectue sur les tweets de u_1.

Notez que dans la section suivante, nous considérons toutes les interactions des utilisateurs. Étant donné que les graphes sociaux sont très dynamiques et que le comportement des utilisateurs peut varier dans le temps, nous devrions adapter cette approche en prenant en compte seulement les interactions effectuées dans une fenêtre de temps donnée.

6.3 Clustering basé sur un score d'interactions globales

Dans cette section, nous présentons une première approche dans laquelle nous utiliserons le graphe d'interaction global. Cette approche est un peu différente de notre objectif de départ, dont le but était d'identifier des utilisateurs ayant le même rôle au sein du réseau social en utilisant les interactions. Notre intuition est, qu'un clustering basé sur les différentes interactions entre les utilisateurs fournira des clusters plus pertinents.

6.3.1 Clustering basé sur les occurrences d'interactions globales

Dans un premier temps, nous adoptons le score d'interaction suivant pour un utilisateur qui prend en compte le nombre d'interactions que l'utilisateur effectue.

Définition 13 (Score d'interaction global basé sur les occurrences) *Score d'interaction global basé sur les occurrences σ_u^g pour un utilisateur u est défini comme :*

$$\begin{aligned} \forall v \in \mathcal{U}, \omega(v,u) &= \Sigma_{a \in \mathcal{A}}\ count((v,u),a) \\ \sigma_u^g &= \log\left(\frac{\Sigma_{v \in \mathcal{U}}\ \omega(v,u)}{max_{w \in \mathcal{U}}(\Sigma_{v \in \mathcal{U}}\ \omega(w,v)}\right) \end{aligned} \tag{6.1}$$

Notez que la normalisation du score est effectuée par la fonction log pour lisser les différences entre les utilisateurs.

Selon ce score d'interaction (puisque les données ne sont pas labellisées), nous avons décidé d'utiliser un clustering non supervisé. Plus précisément, nous avons choisi l'algorithme de clustering K-Means pour sa scalabilité et parce qu'il est connu pour donner de bons résultats de clustering. Pour déterminer le nombre de K-clusters, nous nous basons sur le score Silhouette [75].

Pour l'approche du score d'interaction globale basée sur les occurrences, nous observons que le score de Silhouette augmente avec le nombre de clusters (voir Figure 6.1). Cela montre qu'aucun nombre de clusters ne semble être meilleur qu'un autre (sauf peut-être des clusters avec un seul utilisateur). De plus, l'analyse manuelle d'un clustering, par exemple avec $K = 4$ ou $K = 5$ révèle que les clusters obtenus contiennent des classes d'utilisateurs très hétérogènes.

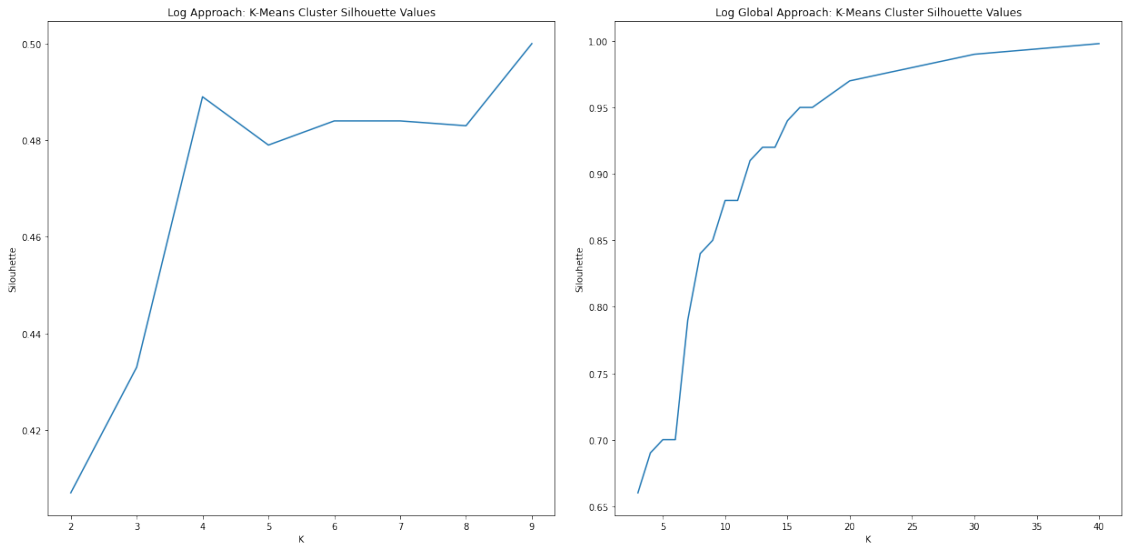

FIGURE 6.1 – Score de silhouette avec l'approche d'interactions globales

6.3.2 Clustering basé sur le PageRank d'interactions globales

Il a été démontré que le PageRank peut calculer avec précision les classements d'influence. De plus, il n'est pas influencé par le nombre de followers mais par les interactions des utilisateurs [76]. Par conséquent, nous nous attendons à ce qu'un score d'interaction global basé sur le PageRank fournisse une meilleure classification des utilisateurs. Les scores PageRank pour un utilisateur u_i *in* $calU$ sont estimés par la formule suivante :

$$PR(u_i) = (1-\alpha) + \alpha \sum_{u_j \in In(u_i)} \frac{PR(u_j)}{Out(u_j)} \tag{6.2}$$

Où $In(u_i)$ dénote l'ensemble des utilisateurs qui ont une interaction avec u_i ($c.a.d$, $\{u_j \in \mathcal{U}, (u_i, u_j) \in \mathcal{E}$), $Out(u_j)$ le degré sortant de l'utilisateur u_j, α le facteur d'amortissement.

Nous prenons en compte plusieurs occurrences de la même interaction entre deux utilisateurs en supposant qu'ils illustrent une forte interaction entre ces utilisateurs. Ceci est modélisé par les poids des arêtes du graphe d'interaction $calG$, le poids d'une arête entre deux utilisateurs est le nombre total d'interactions entre eux.

Afin de calculer un score de PageRank en utilisant le poids des arêtes, nous utilisons l'algorithme Weighted PageRank (WPR) [77]. WPR attribue des scores plus élevés aux nœuds plus importants

au lieu de diviser le score entre leurs voisins. Les nœuds recevront une valeur proportionnelle à leur nombre d'interactions internes (interactions qu'un utilisateur a eues avec ses tweets) et d'interactions externes (interactions qu'un utilisateur a eues avec les tweets d'autres utilisateurs). Par conséquent, nous adoptons la définition suivante :

Définition 14 (Poids d'interaction)
Le poids d'interactions internes $\mathcal{W}^{in}_{(i,j)}$ *et le poids d'interactions externes* $\mathcal{W}^{out}_{(i,j)}$ *pour une arête* $(u_i, u_j) \in \mathcal{E}$ *est estimé comme :*

$$\mathcal{W}^{in}_{(i,j)} = \frac{\sum\limits_{a \in \mathcal{A}} count((u_j, u_i), a)}{\sum\limits_{v \in In(u_i)} \sum\limits_{a \in \mathcal{A}} count((v, u_i), a)}$$
$$\mathcal{W}^{out}_{(i,j)} = \frac{\sum\limits_{a \in \mathcal{A}} count((u_i, u_j), a)}{\sum\limits_{v \in Out(u_i)} \sum\limits_{a \in \mathcal{A}} count((u_i, v), a)} \tag{6.3}$$

Enfin, nous adaptons la formule du Weighted PageRank (WPR) proposé par par Xing et al. [77] pour prendre en considération le poids d'interactions sur les arêtes.

Définition 15 (Score de PageRank d'interactions pondérées) *En utilisant la formule de PageRank précédente et les poids d'interactions défini ci-dessus, nous estimons :*

$$WPR(u_i) = (1 - \alpha) + \alpha \sum_{p_j \in In(u_i)} WPR(u_j) \times \mathcal{W}^{in}_{(j,i)} \times \mathcal{W}^{out}_{j,i)} \tag{6.4}$$

6.4 Clustering basé sur les profils d'interactions

En considérant toutes les interactions comme similaires, nous masquons les différences entre les comportements des utilisateurs. En effet, l'analyse de quelques comptes semble révéler que certains utilisateurs semblent privilégier certaines interactions par rapport à d'autres et cela pourrait être un critère de classification pertinent. Pour vérifier cette intuition, nous avons construit un profil d'interactions pour les utilisateurs qui est défini comme ceci :

Définition 16 (Profil d'interaction) *Le profil d'interaction d'un utilisateur u est un quadruplet* $\sigma_u^p(\sigma_{rt}(u), \sigma_{qt}(u), \sigma_{rp}$ *où chaque dimension* σ_a *est le score d'interaction spécifique déterminé sur le graphe de restriction* $\mathcal{G}_a$*.*

Comme pour l'approche globale, nous comparons l'approche simple avec des scores d'interactions spécifiques estimés avec le nombre d'interactions de l'action correspondante et le PageRank.

6.4.1 Clustering basé sur les occurrences de profils d'interactions

Pour cette approche, nous considérons qu'un poids d'interactions spécifique pour une interaction a est estimé sur le graphe restreint $\mathcal{G}_a$ comme :

$$\forall(u, v, a) \in \mathcal{U}^2 \times \mathcal{A}, \omega(u, v, a) = count((u, v), a) \tag{6.5}$$

Par conséquent, nos scores de profils d'interactions sont estimés comme suit :

Définition 17 *Scores des profils d'interactions basés sur les occurrences] Les scores sont pour l'approche d'interaction basée sur les occurrences* σ_u^p *pour un utilisateur u est défini comme ce qui suit :*

$$\forall a \in \mathcal{A}, \sigma_a(u) = \log\left(\frac{\Sigma_{v\in\mathcal{U}}\omega((v, u), a)}{max_{w\in\mathcal{U}}(\Sigma_{v\in\mathcal{U}}\omega((w, v), a)}\right) \tag{6.6}$$

Une fois ces scores calculés, nous effectuons le clustering non supervisé K-Means. Pour évaluer la qualité du clustering, nous avons effectué une validation humaine qui consiste à analyser manuellement un échantillon de 50 comptes choisis aléatoirement à l'intérieur de chaque cluster.

Nous observons qu'avec l'approche des profils d'interaction basée sur les occurrences, nos clusters restent hétérogènes, ainsi qu'avec l'approche globale basée sur les occurrences : toutes sortes d'utilisateurs sont présents dans chaque cluster. Ce phénomène peut s'expliquer par le fait que cette méthode n'utilise que des valeurs en degrés. Cependant, nous visons à classer les utilisateurs en fonction des interactions sur leurs messages. Il a été démontré que les messages partagés par les utilisateurs populaires ou centraux du graphique peuvent être diffusés efficacement [78].

6.4.2 Profils d'interactions basés sur le PageRank

Au lieu d'un simple calcul des scores d'interaction, basé sur le nombre d'occurrences, nous les estimons en utilisant une approche de PageRank. Par conséquent, nous avons considéré le graphe de restriction $\mathcal{G}_a$ de chaque interaction et avons exécuté l'algorithme de PageRank pondéré (WPR) pour

calculer la dimension associée du profil d'interaction. Notre intuition est, qu'identifier "l'influence" d'un utilisateur sur une interaction donnée (*c.a.d* sa capacité à générer une interaction donnée sur le réseau) caractérise mieux le comportement d'un utilisateur.

Définition 18 (Scores des profils d'interactions basés sur le PageRank) *Les scores des profils d'interactions basés sur le PageRank σ_u^p pour un utilisateur u sont définis comme :*

$$\forall a \in \mathcal{A}, \sigma_a(u) = WPR_{\mathcal{G}_a}(u) \tag{6.7}$$

Où $WPR_{\mathcal{G}_a}(u)$ est le calcul du score de PageRank pondéré pour u sur le graphe $\mathcal{G}_a$, le graphe de restriction $\mathcal{G}$ pour l'interaction a.

Une fois que ces scores sont calculés, nous appliquons aussi l'algorithme non-supervisé K-means. Nous avons choisi d'utiliser K-means après avoir effectué une comparaison de plusieurs algorithmes non-supervisés. Les différents algorithmes ont été comparés en utilisant la mesure de silhouette et en faisant varier les tailles des clusters. Sur la figure 6.2, nous présentons les différents résultats avec les algorithmes K-means [62], Gaussian Mixture [79], Agglomerative Clustering [54]. Le clustering produit des clusters aux caractéristiques très différentes (voir tableau 6.4).

TABLE 6.3 – Weighted PageRank : Clusters composition

	Size	**Composition**	**Types**
Cluster 1	92.63%	100% composed from common users	Common users
Cluster 2	5.44%	55% composed from common users and 45% popular users (more than 4000 followers)	Moderately popular users, local celebrities, doctors, media specialists and active community users
Cluster 3	0.59%	55% composed from entities and 45% human users but mainly above 10 000 followers	Entities, professional users, brands, hospital, city and feed/news accounts
Cluster 4	0.66%	60% composed from popular user more than 4000 followers and 35% users with more than 10 000 followers	Influencers, writers, journalist, attorneys
Cluster Outliers	0.68%	60% human users, 40% entities. With 45% users with more than 100 000 followers and 40% with more than 10 000 followers	Celebrities, international news, politicians and brands

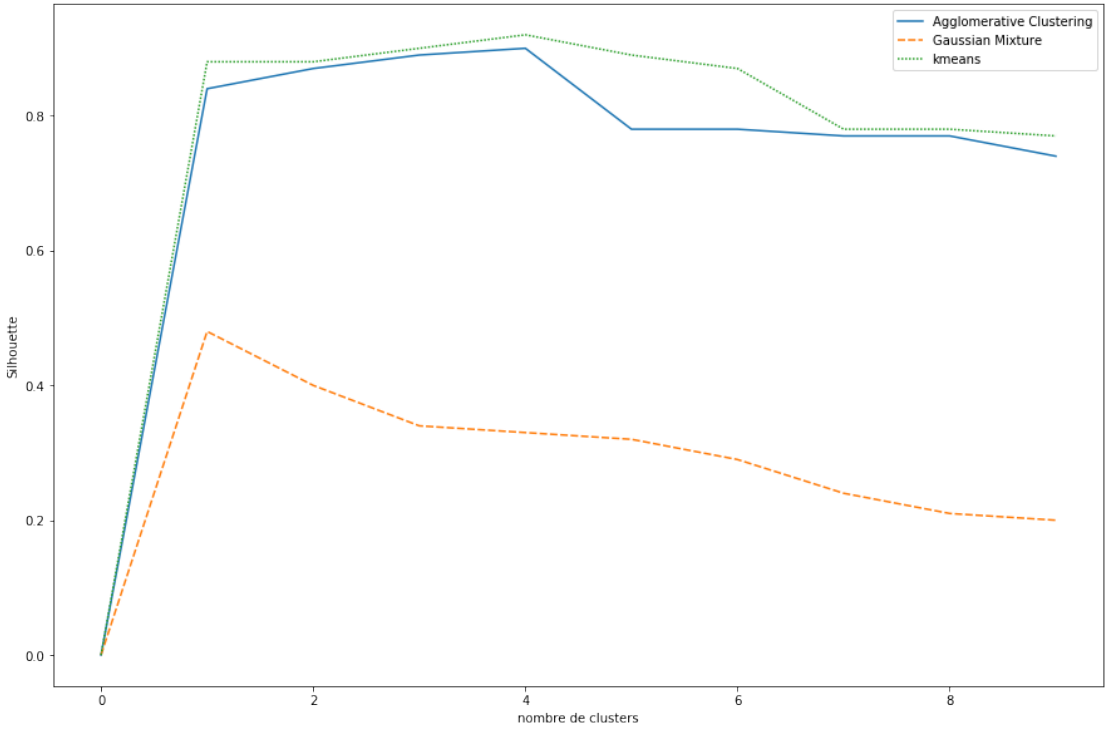

FIGURE 6.2 – Comparaison des différents scores de silhouette

Globalement, nous voyons que les actions `Reply` sont, ce qui est principalement fait par de vrais utilisateurs. A l'inverse, les entités (entreprises, médias, etc.) génèrent plus d'actions `Retweet`. Les

Table 6.4 – Weighted PageRank : Clusters summary

	Weighted PageRank Value
Cluster 1	Reply PageRank is in average **9.10% smaller**, Retweets PageRank is in average **26.56% smaller**, Quote PageRank is in average **29.39% smaller**.
Cluster 2	Reply PageRank is in average **70.84% greater**, Mentions PageRank is in average **20.30% smaller**, Quote PageRank is in average **13.99% smaller**.
Cluster 3	Retweet PageRank is in average **520% greater**, Quote PageRank is in average **230% greater**, Mention PageRank is in average **204% greater**.
Cluster 4	Reply PageRank is in average **182% greater**, Mention PageRank is in average **181% greater**, Retweet PageRank is in average **84.71% greater**.
Cluster Outliers	Reply PageRank is in average **493% greater**, Retweet PageRank is in average **3155% greater**, Quote PageRank is in average **3857% greater**.

`Mentions` peuvent être généré à la fois par des humains et des entités. Comme il existe également une corrélation entre la popularité et les actions `Retweet`, nous pouvons considérer `Quote` comme une sorte de `Retweet`. L'utilisation du PageRank pondéré sur les différents graphes d'interaction pour estimer les scores d'interaction, permet de démontrer l'importance des nœuds qui interagissent avec l'utilisateur. C'est pourquoi nous obtenons des clusters homogènes avec une grande majorité d'utilisateurs similaires.

En ce qui concerne l'approche du profil d'interactions basée sur les occurrences, nous avons évalué la qualité du clustering avec une validation humaine en analysant manuellement (c'est-à-dire lire les fils d'actualités des utilisateurs, les descriptions, les photos) un échantillon de 50 comptes choisis au hasard pour chaque cluster. Les clusters étant plus homogènes, il a été possible de qualifier les différentes classes d'utilisateurs que nous avons identifiées.

Le tableau 6.3 présente les résultats de notre analyse dans lequel les types sont définis par notre analyse manuelle de chaque utilisateurs.

Enfin, nous effectuons une dernière expérience pour valider notre clustering. Nous avons pris 100 nouveaux utilisateurs que nous avons labellisés manuellement avec un numéro de cluster, selon la composition des clusters que nous avons observés avec notre jeu de données initial. Ensuite, nous utilisons notre algorithme de clustering pour les affecter dans un cluster. Pour ces nouveaux utilisateurs, on obtient le résultat suivant : 96 % d'entre eux ont été tagués avec le bon ID de cluster, ce qui signifie que les clusters que nous avons obtenus correspondent à des classes d'utilisateurs bien identifiées.

6.5 Conclusion

Dans ce chapitre nous avons présenté une méthode pour clusteriser les utilisateurs de Twitter en fonction des interactions qu'ils pouvaient générer sur leurs tweets. Sur la base des graphes d'interactions et du calcul du PageRank pondéré, nous avons déterminé les profils d'interactions des utilisateurs. Ensuite, nous avons effectué un clustering non supervisé en utilisant l'algorithme K-mean qui regroupe les utilisateurs avec des profils d'interactions similaires. Nos expériences et nos validations manuelles confirment que cette approche fournit des clusters pertinents.

Chapitre 7

Conclusion

Dans ce chapitre nous allons conclure ce document. La première partie revient sur nos différentes contributions théoriques et une seconde section abordera les différentes perspectives de ces travaux.

7.1 Contributions

Dans nos travaux nous avons construit plusieurs jeux de données à partir du réseau social Twitter. Mais nous avons toujours essayé de créer des approches pouvant être appliquées sur d'autres types de réseaux en utilisant les méta-données ou la topologie du graphe.

Dans notre première étude, nous avons étudié les différentes méta-données des utilisateurs de Twitter. L'étude statistique du jeu de données nous a montré des écarts entre une petite partie des utilisateurs. Cela nous a amené à étudier l'anormalité de ces données pour identifier des comportements anormaux. Notre approche basée sur une classification non-supervisée, nous a permis d'identifier des utilisateurs qui, dans la plupart des cas, étaient des faux influenceurs, des bots ou des comptes ne respectant pas la charte d'utilisation de Twitter (comptes bannis). Cette étude basique nous a permis de nous focaliser sur une donnée des comptes utilisateurs, les followers (abonnés).

Les followers traduisent la visibilité d'un compte sur Twitter. Ce réseau social peut être représenté comme un graphe dirigé, dans lequel les liens avec les followers sont des arêtes entrantes et les followees sont des arêtes sortantes. Nous avons constaté qu'à la différence de Facebook, ces deux valeurs sont indépendantes et ne progressent pas à la même vitesse. Nous nous sommes focalisés sur l'évolution du nombre de followers pour identifier l'évolution de la popularité d'un utilisateur. Dans un premier temps, nous avons modélisé l'évolution de followers en série temporelle en chiffre brut, puis nous avons transformé la série temporelle en gain de followers. Une approche basique portant sur la distance entre deux séries temporelles ne donnait pas de résultats satisfaisants et, notre machine limitait la taille de la matrice des distances. De plus, le temps de calcul était trop important. Nous avons donc discrétisé les séries temporelles des gains de followers en s'inspirant de la méthode SAX [72]. Nous avons transformé l'évolution du gain de followers sous la forme d'une séquence de symboles. Nous avons divisé notre jeu de données en trois groupes, les utilisateurs populaires, les utilisateurs non-populaires et les utilisateurs devenant populaires. A partir de ces trois groupes, nous avons extrait des sous-séquences fréquentes et avons gardé seulement celles qui sont propres à chaque groupe. L'identification de ses sous-séquences nous a permis de valider cette approche et, pour permettre le passage à l'échelle de notre approche,

nous avons mis au point un modèle de stockage et d'identification de motifs sur la base de table de hashage. La comparaison des performances de notre modèle avec des modèles plus classiques montre son efficacité. Les résultats obtenus ont montré qu'il était possible d'identifier des utilisateurs devenant populaires avec des semaines d'avance. [80].

Notre seconde approche, consiste à identifier des groupes d'individus partageant le même rôle basé sur la topologie du graphe d'interactions. Sur Twitter, les utilisateurs peuvent interagir sur les différents contenus avec différentes actions. Nous avons construit les graphes d'interactions sur deux jeux de données pour lesquels les utilisateurs discutent du même sujet (COVID et NBA). Nous avons émis l'hypothèse que tous les utilisateurs de Twitter n'agissent pas de la même manière selon le profil de l'auteur du tweet. En utilisant un PageRank pondéré, nous avons mesuré un score sur le graphe d'interactions globales dans lequel les différentes actions sont réunies sur une seule arête. Le poids de ses arêtes est le nombre d'interactions globales entre deux noeuds. Les résultats de cette approche n'ont pas permis d'identifier des groupes d'utilisateurs similaires. Dans une deuxième partie de cette approche nous avons réutilisé un PageRank pondéré mais, cette fois-ci, nous avons pris en considération les différentes actions. Dans ce cas, il peut y avoir plusieurs arêtes entre deux noeuds et le poids de ces arêtes est la somme des actions du même type entre les deux utilisateurs. Les résultats obtenus dans cette approche ont permis de valider que les utilisateurs n'agissaient pas de la même manière par rapport au type de compte. Nous avons aussi proposé l'identification de ces différents groupes d'utilisateurs en utilisant un algorithme de clustering (K-means). Les groupes obtenus étaient composés d'utilisateurs avec des rôles similaires.

Dans la prochaine section nous allons aborder les différentes perspectives envisageables pour la suite de ces travaux.

7.2 Perspectives

Les réseaux sociaux en ligne ne cessent de changer, que ce soit sur la forme ou le type de contenu chaque réseau social essaie de se démarquer. La popularité de ces plateformes varient et, chaque année, de nouveaux réseaux apparaissent. Une des premières perspectives de ces travaux serait d'appliquer nos différentes approches sur d'autres plateformes ou bien d'augmenter la quantité de données en faisant appel au service payant de Twitter pour obtenir 100% des données.

L'identification de comportements anormaux : dans ces travaux nous avons seulement dépoussiéré la surface de ce problème. Dans un premier temps, nous pouvons construire un jeu de données labellisé. Nous avons présenté une approche utilisant des algorithmes non-supervisés. La plupart de ces algorithmes fonctionnent à l'aide de calculs de distance, ce qui reste encore, aujourd'hui, une méthode coûteuse et pas vraiment adaptée aux gros jeu de données comme les réseaux sociaux. Un jeu de données labellisé permettrait l'utilisation d'algorithmes supervisés, d'évaluer la précision et de comparer l'approche non-supervisée. L'approche supervisée permettrait aussi de ne plus utiliser de taux de contamination qui peut s'avérer non précis. Enfin, il faudrait construire un modèle d'analyse dynamique permettant d'identifier le moment auquel un utilisateur devient anormal.

Améliorer le modèle de données de l'évolution de la popularité : dans les travaux d'identification d'utilisateurs devenant populaires, nous nous sommes focalisés sur une seule dimension. Il peut être intéressant d'ajouter d'autres dimensions permettant de comprendre l'élément déclencheur. L'évolution de la popularité, dans la plupart des cas, est gagnée au fil du temps grâce à l'activité sur le réseau social. Pour d'autres, il suffit de faire le "buzz". Comprendre et identifier rapidement ce phénomène permettrait d'améliorer l'identification et d'expliquer ce gain de popularité. Pour cela, il faut ajouter une analyse du contenu des utilisateurs, qui pourrait facilement se faire en parallèle avec notre modèle H^2M . Nous pourrions améliorer notre alphabet, actuellement basé sur une étude de la distribution des données. Cet alphabet pourrait être amélioré avec la réalisation d'un modèle spécifique. De plus, ajouter d'autres variables comme les nombres de Retweets, Replies, Quotes, Mentions, pour transformer ce vecteur en symbole, permettrait de représenter les différents évènements sous-jacents à l'évolution de la popularité. Enfin, nous pourrions trouver d'autres comportements qui pourraient être modélisés comme la popularité. Notamment, en étudiant l'achat d'abonnés et en identifiant les utilisateurs qui font appel à ce service hors-charte.

Amélioration du clustering basé sur les interactions : une fois encore ces travaux utilisent des algorithmes non-supervisés difficiles à évaluer. Le passage par un jeu de données labéllisé permettrait ainsi de mesurer précisément la qualité des clusters. Nous pourrions passer sur une échelle plus petite en utilisant des algorithmes de détection de communauté. Identifier des communautés, réduirait le nombre d'individus, ce qui pourrait permettre d'identifier les acteurs clés d'une communauté. Identifier les différents profils d'utilisateurs dans les communautés, permettrait de les caractériser et de les comparer. Enfin, nous pourrions effectuer une analyse de la dynamicité du graphe d'interactions en

recalculant les scores d'interactions sur une période de temps glissante. Le changement de comportement des utilisateurs pourrait engendrer des changements dans les scores, qui eux-mêmes, pourraient être analysés avec la même méthode que l'évolution de la popularité. Cela qui permettrait de prédire quand un utilisateur va changer de comportement.

Ces différents travaux pourraient être assemblés pour fournir un modèle d'analyse plus complet avec des données pré-calculées et être utilisés dans des applications d'analyse de données provenant des réseaux sociaux. Les utilisateurs pourraient, par ce biais, obtenir des descripteurs caractérisant leurs profils ainsi que leurs comportements.

Bibliographie

[1] V. D. Blondel, J.-L. Guillaume, R. Lambiotte et E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of statistical mechanics : theory and experiment*, vol. 2008, n°. 10, p. P10008, 2008.

[2] A. Sari *et al.*, "A review of anomaly detection systems in cloud networks and survey of cloud security measures in cloud storage applications," *Journal of Information Security*, vol. 6, n°. 02, p. 142, 2015.

[3] J. Lin, E. Keogh, S. Lonardi et B. Chiu, "A symbolic representation of time series, with implications for streaming algorithms," dans *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, 2003, p. 2–11.

[4] W. Akram et R. Kumar, "A study on positive and negative effects of social media on society," *International Journal of Computer Sciences and Engineering*, vol. 5, n°. 10, p. 347–354, 2017.

[5] S. Alavi, I. Mehdinezhad et B. Kahshidinia, "A trend study on the impact of social media on advertisement," *International Journal of Data and Network Science*, vol. 3, n°. 3, p. 185–200, 2019.

[6] M. A. Smith, "Catalyzing social media scholarship with open tools and data," *Journal of Contemporary Eastern Asia*, vol. 14, n°. 2, p. 87–96, 2015.

[7] A. O. Kemi *et al.*, "Impact of social network on society : A case study of abuja," *American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)*, vol. 21, n°. 1, p. 1–17, 2016.

[8] J. Amedie, "The impact of social media on society," 2015.

[9] D. Mican, D.-A. Sitar-Tăut et I.-S. Mihuţ, "User behavior on online social networks : Relationships among social activities and satisfaction," *Symmetry*, vol. 12, n°. 10, p. 1656, 2020.

[10] J. J. Gillen, M. Freeman et H. Tootell, “Human behavior in online social networks,” dans *2017 IEEE International Symposium on Technology and Society (ISTAS)*, 2017, p. 1–6.

[11] J. Wielki, “Analysis of the role of digital influencers and their impact on the functioning of the contemporary on-line promotional system and its sustainable development,” *Sustainability*, vol. 12, n°. 17, p. 7138, 2020.

[12] I. Marketing, “What is an influencer ?” [En ligne]. Disponible : https://influencermarketinghub.com/what-is-an-influencer/

[13] J. Gross et F. V. Wangenheim, “The big four of influencer marketing. a typology of influencers.” *Marketing Review St. Gallen*, vol. 2, p. 30–38, 2018.

[14] H.-L. Wei, K.-Y. Lin, H.-P. Lu et I.-H. Chuang, “Understanding the intentions of users to ‘stick’to social networking sites : a case study in taiwan,” *Behaviour & Information Technology*, vol. 34, n°. 2, p. 151–162, 2015.

[15] V. S. Subrahmanian, A. Azaria, S. Durst, V. Kagan, A. Galstyan, K. Lerman, L. Zhu, E. Ferrara, A. Flammini et F. Menczer, “The darpa twitter bot challenge,” *Computer*, vol. 49, n°. 6, p. 38–46, 2016.

[16] I. Pozzana et E. Ferrara, “Measuring bot and human behavioral dynamics,” *Frontiers in Physics*, vol. 8, p. 125, 2020.

[17] C. Cai, L. Li et D. Zengi, “Behavior enhanced deep bot detection in social media,” dans *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*. IEEE, 2017, p. 128–130.

[18] L. C. Freeman, “Centrality in social networks conceptual clarification,” *Social networks*, vol. 1, n°. 3, p. 215–239, 1978.

[19] P. Bonacich, “Factoring and weighting approaches to status scores and clique identification,” *Journal of mathematical sociology*, vol. 2, n°. 1, p. 113–120, 1972.

[20] J. S. Coleman, *Mathematics of collective action*. Transaction Publishers, 2017.

[21] P. Bonacich, “Power and centrality : A family of measures,” *American journal of sociology*, vol. 92, n°. 5, p. 1170–1182, 1987.

[22] R. Burt et R. S. Burt, *Toward a structural theory of action : Network models of social structure, perception, and action*. Academic Press, 1982.

[23] G. Sabidussi, "The centrality index of a graph," *Psychometrika*, vol. 31, n°. 4, p. 581–603, 1966.

[24] S. L. Hakimi, "Optimum locations of switching centers and the absolute centers and medians of a graph," *Operations research*, vol. 12, n°. 3, p. 450–459, 1964.

[25] L. C. Freeman, "A set of measures of centrality based on betweenness," *Sociometry*, p. 35–41, 1977.

[26] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, n°. 1, p. 39–43, 1953.

[27] P. Bonacich, "Some unique properties of eigenvector centrality," *Social networks*, vol. 29, n°. 4, p. 555–564, 2007.

[28] M. Piraveenan, M. Prokopenko et L. Hossain, "Percolation centrality : Quantifying graph-theoretic impact of nodes during percolation in networks," *PloS one*, vol. 8, n°. 1, p. e53095, 2013.

[29] L. Page, S. Brin, R. Motwani et T. Winograd, "The PageRank citation ranking : Bringing order to the web." Stanford InfoLab, Rapport technique, 1999.

[30] Z. Gyongyi, H. Garcia-Molina et J. Pedersen, "Combating web spam with trustrank," dans *Proceedings of the 30th international conference on very large data bases (VLDB)*, 2004.

[31] J. M. Kleinberg, M. Newman, A.-L. Barabási et D. J. Watts, *Authoritative sources in a hyperlinked environment*. Princeton University Press, 2011.

[32] D. Tunkeland, "A twitter analog to pagerank." [En ligne]. Disponible : https://thenoisychannel.com/2009/01/13/a-twitter-analog-to-pagerank/

[33] J. Weng, E.-P. Lim, J. Jiang et Q. He, "TwitterRank : finding topic-sensitive influential twitterers." ACM Press, 2010, p. 261. [En ligne]. Disponible : http://portal.acm.org/citation.cfm?doid=1718487.1718520

[34] "Klout score : Measuring influence across multiple social networks," dans *IEEE Intl. Conf. on Big Data (Big Data)*, 2015, p. 2282–2289.

[35] D. Kempe, J. Kleinberg et E. Tardos, "Maximizing the Spread of Influence through a Social Network," *Theory of Computing*, vol. 11, p. 43, 2015.

[36] K. Saito, K. Ohara, Y. Yamagishi, M. Kimura et H. Motoda, "Learning diffusion probability based on node attributes in social networks," dans *International Symposium on Methodologies for Intelligent Systems*. Springer, 2011, p. 153–162.

[37] J. Arndt, “Role of product-related conversations in the diffusion of a new product,” *Journal of Marketing Research*, vol. 4, n°. 3, p. 291–295, 1967. [En ligne]. Disponible : http://www.jstor.org/stable/3149462

[38] J. F. Engel, R. J. Kegerreis et R. D. Blackwell, “Word-of-mouth communication by the innovator,” *Journal of Marketing*, vol. 33, n°. 3, p. 15–19, 1969. [En ligne]. Disponible : http://www.jstor.org/stable/1248475

[39] N. Matsumura, “Finding Influencers and Consumer Insights in the Blogosphere,” p. 8, 2008.

[40] M. Cha, “Measuring User Influence in Twitter : The Million Follower Fallacy,” p. 8, 2010.

[41] I. Anger et C. Kittl, “Measuring influence on Twitter.” ACM Press, 2011, p. 1. [En ligne]. Disponible : http://dl.acm.org/citation.cfm?doid=2024288.2024326

[42] R. Kumar, J. Novak et A. Tomkins, “Structure and evolution of online social networks,” dans *Link mining : models, algorithms, and applications.* Springer, 2010, p. 337–357.

[43] S. Wasserman, K. Faust *et al.*, “Social network analysis : Methods and applications,” 1994.

[44] S. Seidman et B. Foster, “A graph-theoretic generalization of the clique concept*,” *Journal of Mathematical Sociology*, vol. 6, p. 139–154, 01 1978.

[45] Y. Wang, X. Jian, Z. Yang et J. Li, “Query optimal k-plex based community in graphs,” *Data Science and Engineering*, vol. 2, n°. 4, p. 257–273, 2017.

[46] D. Gibson, R. Kumar et A. Tomkins, “Discovering large dense subgraphs in massive graphs,” dans *Proceedings of the 31st international conference on Very large data bases.* Citeseer, 2005, p. 721–732.

[47] J. Hopcroft, O. Khan, B. Kulis et B. Selman, “Natural communities in large linked networks,” dans *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2003, p. 541–546.

[48] M. E. Newman, “Modularity and community structure in networks,” *Proceedings of the national academy of sciences*, vol. 103, n°. 23, p. 8577–8582, 2006.

[49] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski et D. Wagner, “On modularity clustering,” *IEEE transactions on knowledge and data engineering*, vol. 20, n°. 2, p. 172–188, 2007.

[50] O. Gach et J.-K. Hao, “Maximisation de la modularité sous condition de fusion.”

[51] S. Fortunato et M. Barthelemy, "Resolution limit in community detection," *Proceedings of the national academy of sciences*, vol. 104, n°. 1, p. 36–41, 2007.

[52] M. Girvan et M. E. Newman, "Community structure in social and biological networks," *Proceedings of the national academy of sciences*, vol. 99, n°. 12, p. 7821–7826, 2002.

[53] P. H. Sneath, "The application of computers to taxonomy," *Microbiology*, vol. 17, n°. 1, p. 201–226, 1957.

[54] D. Defays, "An efficient algorithm for a complete link method," *The Computer Journal*, vol. 20, n°. 4, p. 364–366, 01 1977. [En ligne]. Disponible : https://doi.org/10.1093/comjnl/20.4.364

[55] D. E. Goldberg et J. H. Holland, "Genetic algorithms and machine learning," 1988.

[56] C. Pizzuti, "Ga-net : A genetic algorithm for community detection in social networks," dans *International conference on parallel problem solving from nature*. Springer, 2008, p. 1081–1090.

[57] M. Tasgin, A. Herdagdelen et H. Bingol, "Community detection in complex networks using genetic algorithms," *arXiv preprint arXiv :0711.0491*, 2007.

[58] U. N. Raghavan, R. Albert et S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Physical review E*, vol. 76, n°. 3, p. 036106, 2007.

[59] D. M. Hawkins, *Identification of outliers*. Springer, 1980, vol. 11.

[60] L. I. Smith, "A tutorial on principal components analysis," 2002.

[61] M. T. Elbatta et W. M. Ashour, "A dynamic method for discovering density varied clusters," *International Journal of Signal Processing, Image Processing and Pattern Recognition*, vol. 6, n°. 1, 2013.

[62] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," dans *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, n°. 14. Oakland, CA, USA, 1967, p. 281–297.

[63] M. M. Breunig, H.-P. Kriegel, R. T. Ng et J. Sander, "Lof : identifying density-based local outliers," dans *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, p. 93–104.

[64] P. J. Rousseeuw et K. V. Driessen, "A fast algorithm for the minimum covariance determinant estimator," *Technometrics*, vol. 41, n°. 3, p. 212–223, 1999.

[65] F. T. Liu, K. M. Ting et Z.-H. Zhou, “Isolation forest,” dans *2008 eighth ieee international conference on data mining*. IEEE, 2008, p. 413–422.

[66] M. Abouelhoda et M. Ghanem, “String mining in bioinformatics,” dans *Scientific Data Mining and Knowledge Discovery*. Springer, 2009, p. 207–247.

[67] R. Srikant, “Fast algorithms for mining association rules and sequential patterns,” Thèse de doctorat, Citeseer, 1996.

[68] J. Han, J. Pei et Y. Yin, “Mining frequent patterns without candidate generation,” *ACM sigmod record*, vol. 29, n°. 2, p. 1–12, 2000.

[69] M. J. Zaki, “Spade : An efficient algorithm for mining frequent sequences,” *Machine learning*, vol. 42, n°. 1-2, p. 31–60, 2001.

[70] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal et M.-C. Hsu, “Freespan : frequent pattern-projected sequential pattern mining,” dans *Proc. ACM Intl. Conf. on Knowledge Discovery and Data mining (KDD)*, 2000, p. 355–359.

[71] I. N. Junejo et Z. A. Aghbari, “Using SAX representation for human action recognition,” *Journal of Visual Communication and Image Representation*, vol. 23, n°. 6, p. 853–861, août 2012. [En ligne]. Disponible : https://linkinghub.elsevier.com/retrieve/pii/S1047320312000806

[72] J. Lin, E. Keogh, L. Wei et S. Lonardi, “Experiencing SAX : a novel symbolic representation of time series,” *Data Mining and Knowledge Discovery*, vol. 15, n°. 2, p. 107–144, août 2007. [En ligne]. Disponible : http://link.springer.com/10.1007/s10618-007-0064-z

[73] D. Köppl, “Separate chaining meets compact hashing,” *arXiv preprint arXiv :1905.00163*, 2019.

[74] W. Litwin, “Linear Hashing : A New Tool for File and Table Addressing,” dans *Intl Conf. on Very Large Data Bases (VLDB)*, 1980, p. 212–223.

[75] P. J. Rousseeuw, “Silhouettes : A graphical aid to the interpretation and validation of cluster analysis,” *Journal of Computational and Applied Mathematics*, vol. 20, p. 53 – 65, 1987. [En ligne]. Disponible : http://www.sciencedirect.com/science/article/pii/0377042787901257

[76] B. Hajian et T. White, “Modelling influence in a social network : Metrics and evaluation,” dans *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing*. IEEE, 2011, p. 497–500.

[77] W. Xing et A. Ghorbani, "Weighted pagerank algorithm," dans *Proceedings. Second Annual Conference on Communication Networks and Services Research, 2004.* IEEE, 2004, p. 305–314.

[78] T. R. Zaman, R. Herbrich, J. Van Gael et D. Stern, "Predicting information spreading in twitter," dans *Workshop on computational social science and the wisdom of crowds, nips*, vol. 104, n°. 45. Citeseer, 2010, p. 17 599–601.

[79] G. J. McLachlan, S. X. Lee et S. I. Rathnayake, "Finite mixture models," *Annual review of statistics and its application*, vol. 6, p. 355–378, 2019.

[80] J. Debure, S. Brunessaux, C. Constantin et C. d. Mouza, "A pattern-based approach for an early detection of popular twitter accounts," dans *Proceedings of the 24th Symposium on International Database Engineering & Applications*, 2020, p. 1–9.

Annexe A

Listes des publications

— A pattern-based approach for an early detection of popular Twitter accounts, *J. Debure, S. Brunessaux, C. Constantin, C. Du Mouza*, In Proceedings of the 24th Symposium on International Database Engineering & Applications (pp. 1-9).

— An interaction profile-based classification for Twitter users, *J. Debure, S. Brunessaux, C. Constantin, C. Du Mouza*, In The Thirteenth International Conference on Advances in Databases, Knowledge, and Data Applications DBKDA 2021.

Annexe B

Outils et architectures systèmes utilisés

Dans cette annexe, nous allons présenter les différents outils et architectures mis en place pour effectuer les travaux de cette thèse.

B.1 Architecture système analyse d'utilisateurs anormaux

Dans cette section, nous allons présenter l'architecture permettant de collecter, stocker et traiter les méta-données des utilisateurs Twitter. Cette architecture est celle mise en place pour les travaux du chapitre 4.

B.1.1 Architecture d'une plateforme de traitement de données

Pour pouvoir répondre aux exigences du projet auquel les travaux du chapitre 4 sont attachés, nous avons dû mettre en place une chaîne de traitement de données depuis l'API Twitter jusqu'à des outils de visualisation et des bases de données. La figure B.1 présente une vue globale de la plateforme d'analyse. Elle montre les différents outils permettant de recevoir les tweets provenant de l'API Twitter. L'utilisation de Flink et Kafka permet de créer une file de messages facilement parallélisables et réutilisables. Le bloc Tweet Processing fait référence à notre chaîne de traitement du tweet présenté dans le chapitre 4. Ensuite, les données sont stockées dans une base de données NoSQL Cassandra et indexées dans ElasticSearch. Enfin, les données indexées dans ElasticSearch peuvent être visualisées grâce à Kibana et, les différentes statistiques (concernant l'utilisation de la pile Kafka ainsi que les résultats de la classification) peuvent être visualisés sur une interface graphique codée avec Angular.

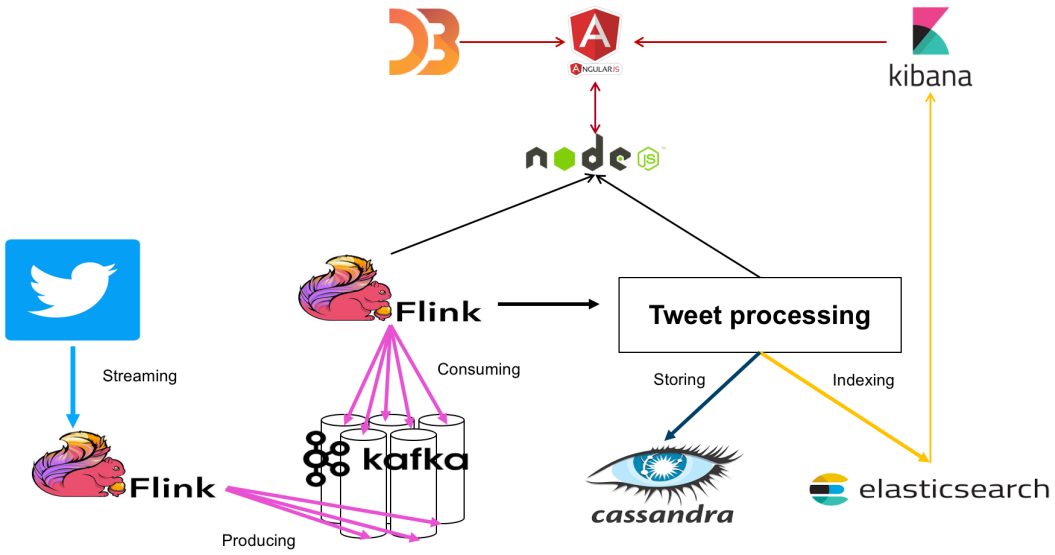

FIGURE B.1 – Plate-forme de traitement de données Twitter

B.2 Outils techniques

Dans cette section, nous présentons les différents outils nous ayant permis de réaliser les expériences des différents travaux.

B.2.1 Dataïku

Dataïku est une plate-forme de Data Science conçue pour les entreprises. Elle permet d'effectuer des chaînes de traitements sur les données. Cette plate-forme propose des outils qui permettent de préparer des données rapidement en utilisant leurs interfaces. De plus, les connecteurs de base de données font gagner un temps considérable. Dataïku est complet et permet de parcourir un échantillon de données, de créer des graphiques et de synthétiser tout dans un tableau de bord facilement éditable. Mais, elle laisse aussi beaucoup de liberté aux utilisateurs car nous pouvons facilement ajouter du code dans différents langages comme Python, Scala, R, SQL, Bash, Spark... Une des fonctionnalités qui nous a beaucoup servi est le laboratoire. Le laboratoire permet de créer ou d'utiliser des algorithmes de machine learning et deep learning en proposant une interface de visualisation des résultats et des performances.

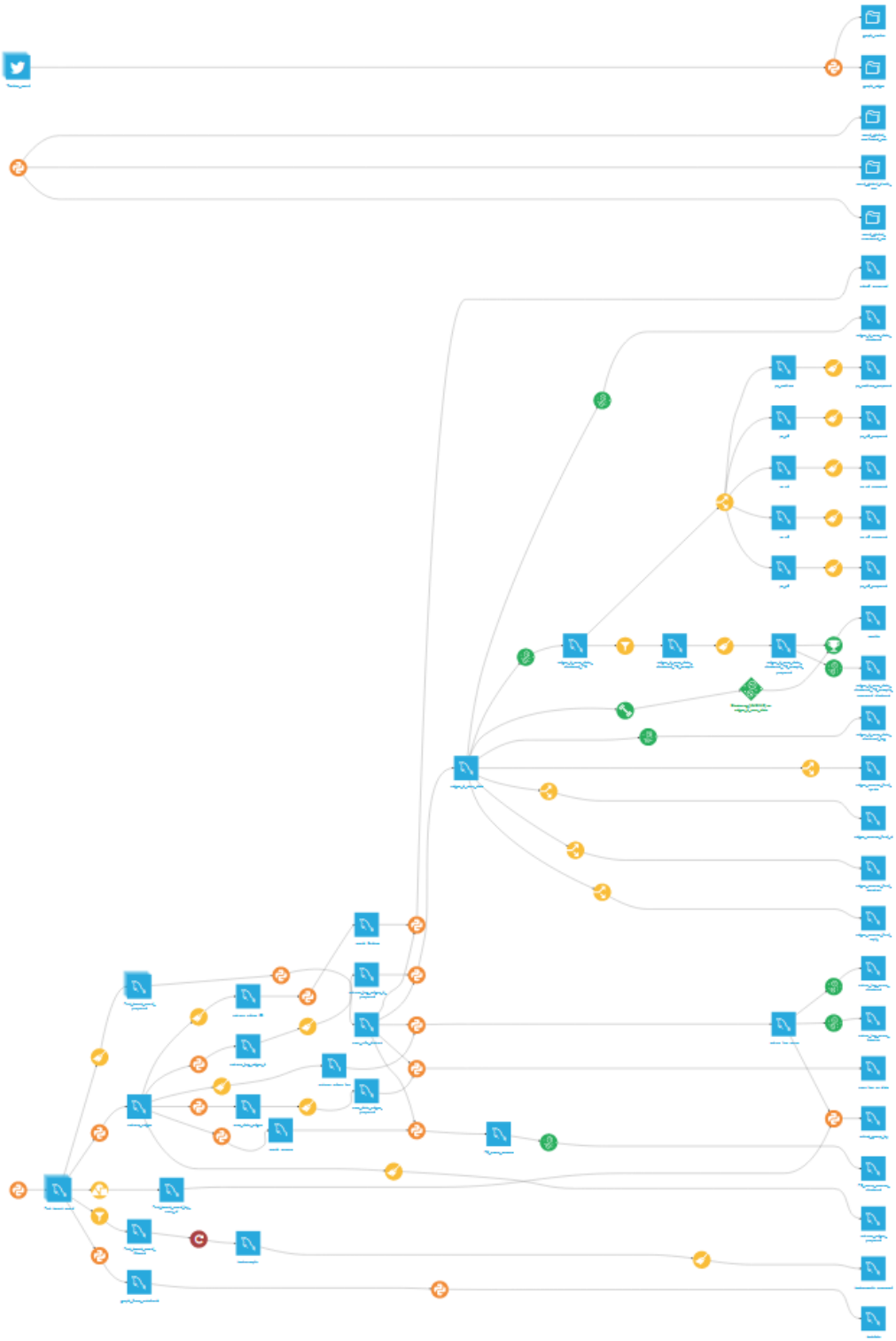

FIGURE B.2 – Projet Dataïku des travaux d'identification de rôles dans les graphes d'interactions

Sur la figure B.2 nous présentons la chaîne de traitement des expériences des travaux sur le clustering d'utilisateur en utilisant les graphes d'interactions. En bleu, ce sont les différents jeux de données intermédiaires à chaque traitement, ils sont principalement stockés dans une base de données SQL (MariaDB) mais peuvent facilement être exportés en CSV. En jaune, les différentes opérations sur les jeux de données, nettoyage, sélection, filtrage. En orange, les différents scripts python utilisés pour créer les graphes et mesurer les scores. Enfin, en vert, ce sont les algorithmes de machine learning entraînés et appliqués sur les données. Le fait d'avoir un flux de traitement de données, nous permet de garder différents états des jeux de données et, sans perdre de temps, nous pouvons refaire de nouvelles expériences à n'importe quel endroit de la chaîne.

Pour préparer nos différents scripts python, nous avons utilisé Jupyter Notebook que nous présentons dans la section suivante.

B.2.2 Jupyter Notebook

Jupyter Notebook (figure B.3) est une interface de programmation permettant d'en-capsuler des lignes de code dans des cellules. Ces cellules sont exécutables dans n'importe quel sens et permettent de garder en mémoire des variables pouvant être exécutées dans d'autre cellules. Elle permet d'afficher directement des graphiques et des zones de texte. Ces deux fonctionnalités sont idéales pour ajouter des commentaires et même pour réaliser un rapport sous la forme d'un PDF ou de diapositives. Il s'agit d'un environnement idéal pour effectuer des expériences.

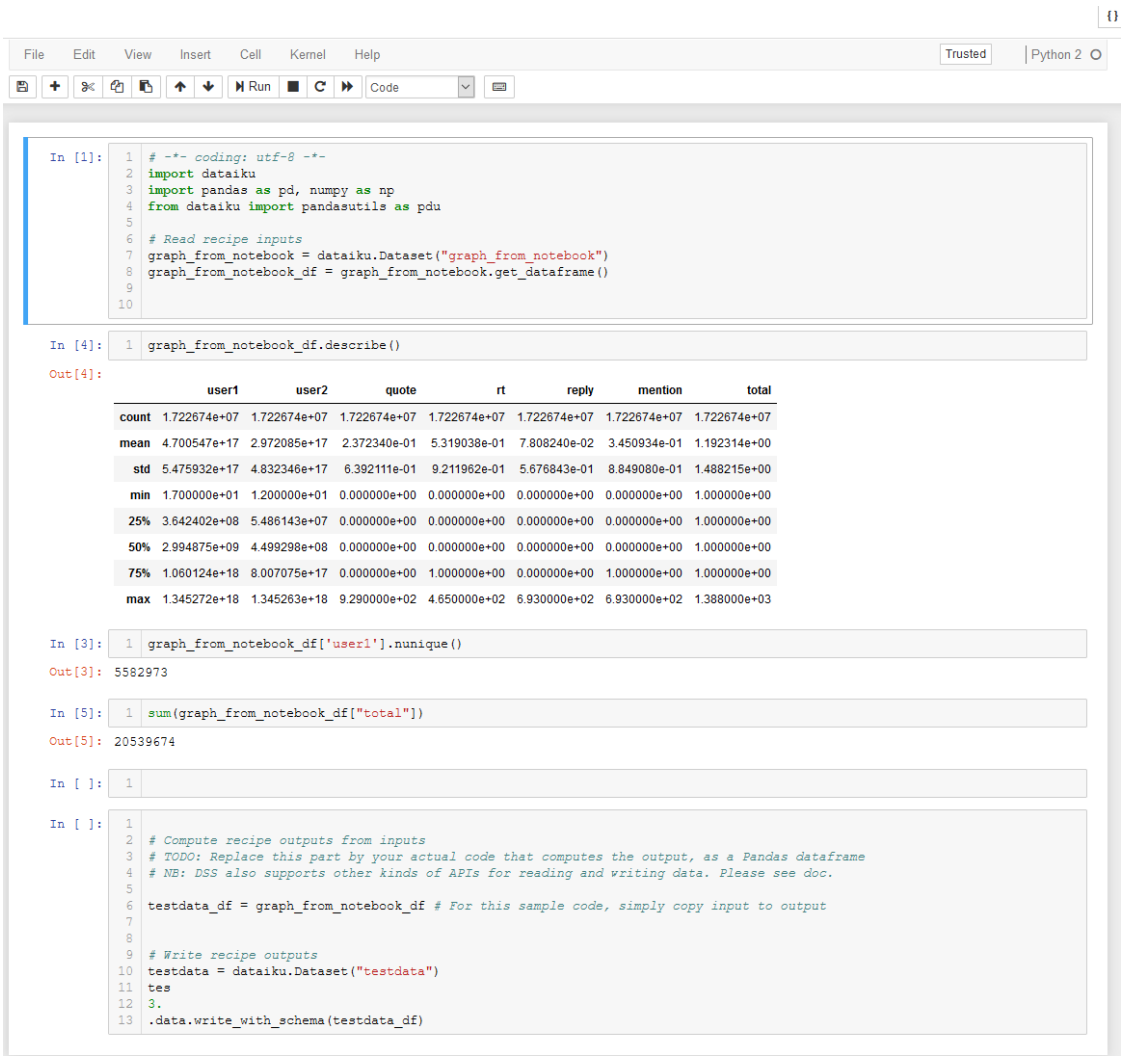

FIGURE B.3 – Jupyter Notebook avec quelques cellules permettant de charger et d'afficher des statistiques sur un jeu de données.

Annexe C

Travaux d'ingénieries

Dans cette annexe, je présente les différents travaux d'ingénierie effectués durant cette thèse CIFRE. Dans mon contrat CIFRE, un tiers de mon temps de travail était consacré à des projets d'ingénierie. Bien que l'entreprise tente toujours de mettre en relation les projets de développement avec les travaux de thèse, il n'est pas toujours possible de le faire. Dans mon cas, j'ai très peu travaillé sur des thématiques proches de mes recherches. C'est pour cette raison que je souhaite, tout de même, présenter brièvement les différents projets sur lesquels j'ai travaillé.

C.1 Surveillance maritime

Dans un projet de surveillance maritime, le but était de proposer des méthodes d'analyses de trajectoires de bateaux en utilisant les émissions AIS. Les AIS sont des données d'identification provenant des balises des bateaux, permettant de connaître leurs positions, vitesses, caractéristiques, types et tailles. A partir de ces données, l'objectif était de proposé une méthode d'analyse de trajectoire afin d'identifier des bateaux ayant des comportements anormaux. Le modèle déjà existant effectuait une comparaison des trajectoires en calculant des distances euclidiennes point à point entre chaque trajectoire de bateau pour les regrouper avec un algorithme de clustering. Le problème de ce modèle, est, qu'il n'est pas possible de traiter de grands nombres de données. Ma principale contribution a été de mettre au point un algorithme de partitionnement de trajectoires basé sur une discrétisation des trajectoires en centroïdes puis, d'appliquer une méthode de partitionnement dichotomique sous forme d'arbres en séparant en deux, à chaque itération, le plus gros ensemble de centroïdes. Cet algorithme nous a permis de regrouper les trajectoires proches afin d'éviter de calculer une distance entre des trajectoires qui ne sont pas dans la même zone de navigation.

C.2 DataLake et droit d'en connaître

Dans ce projet, j'avais pour objectif d'effectuer un état de l'art sur les différents travaux pour la mise en place d'un DataLake. Un DataLake est une structure permettant de regrouper plusieurs base de données de différents types. Ce type de structure est généralement accompagné d'une Federated Search. Une Fereated Search ou, recherche fédérée, est un connecteur permettant d'interroger différentes bases de données en utilisant une seule requête. Après avoir proposé un état de l'art et, après avoir fait une proposition d'architecture micro-services permettant de mettre en place un DataLake, j'ai effectué un

dérisquage du droit d'en connaître du plugin X-Pack d'ElasticSearch. Le but était de pouvoir gérer des droits sur la consultation de documents et sur le masquage de différents champs de documents ElasticSearch selon les profils d'utilisateurs en utilisant le module X-Pack proposé par ElasticSearch.

C.3 Extraction et prédiction de coordonnées d'utilisateurs Twitter

Dans le cadre d'un projet de recherche, j'ai mis au point une chaîne de traitement de données pour affecter, à tous les utilisateurs d'un graphe, une position GPS. La chaîne se décompose en plusieurs parties. Dans la première partie, nous identifions, parmi les utilisateurs, ceux qui ne possèdent pas de champs "location". Dans la seconde partie, nous identifions les utilisateurs qui ont renseigné, dans leur description (ou dans d'autres champs) des noms de villes ou pays en utilisant un algorithme d'extraction d'entités. Ensuite, nous convertissons les différentes entités associées à des lieux en position GPS. Lorsque tous les utilisateurs sont passés par ces différentes étapes, nous créons un graphe de relations entre utilisateurs Nous utilisons ensuite un algorithme de propagation de labels pour associer aux utilisateurs ne possédant pas de coordonnées GPS celles des utilisateurs de son réseau. Tout cela, en calculant la moyenne des coordonnées et en favorisant les utilisateurs avec lesquels les liens sont plus forts. En utilisant également le même principe de graphe et de poids sur les arêtes du chapitre 6.

le cnam

Jonathan DEBURE

Détection de comportements et identification de rôles dans les réseaux sociaux

le cnam

Résumé : Les réseaux sociaux sont devenus des outils de communication primordiaux et sont utilisés quotidiennement par des centaines de millions d'utilisateurs. Tous ces utilisateurs n'ont pas le même comportement sur ces réseaux. Si certains ont une faible activité, publient rarement des messages et suivent peu d'utilisateurs, d'autres, à l'opposé, ont une activité importante, avec de nombreux abonnés et très publient régulièrement. Le rôle important de ces utilisateurs influents en font des cibles intéressantes pour de nombreuses applications, par exemple pour la surveillance ou la publicité. Après une étude des méta-données de ces utilisateurs, afin de détecter des comptes anormaux, nous présentons une approche permettant de détecter des utilisateurs devenant populaires. Notre approche s'appuie sur une modélisation de l'évolution de la popularité sous la forme de motifs fréquents. Ces motifs décrivent les comportements de gain en popularité. Nous proposons un modèle de matching des motifs permettant d'être utilisé avec un flux de données et, nous montrons sa capacité à passer à l'échelle en le comparant à des modèles classiques. Enfin, nous présentons une approche de clustering basé sur le PageRank. Ces travaux permettent d'identifier des groupes d'utilisateurs partageant le même rôle, en utilisant les graphes d'interactions qu'ils génèrent.

Mots clés : réseaux sociaux, clustering, motifs fréquents, comportements, Twitter, PageRank, détection de la popularité

Abstract : Social networks (SN) are omnipresent in our lives today. Not all users have the same behavior on these networks. If some have a low activity, rarely posting messages and following few users, some others at the other extreme have a significant activity, with many followers and regularly posts. The important role of these popular SN users makes them the target of many applications for example for content monitoring or advertising. After a study of the metadata of these users, in order to detect abnormal accounts, we present an approach allowing to detect users who are becoming popular. Our approach is based on modeling the evolution of popularity in the form of frequent patterns. These patterns describe the behaviors of gaining popularity. We propose a pattern matching model which can be used with a data stream and we show its scalability and its performance by comparing it to classic models. Finally, we present a clustering approach based on PageRank. This work allow to identify groups of users sharing the same role, using the interaction graphs.

Keywords : social network, clustering, patterns mining, behaviours, Twitter, PageRank, popularity detection