Médias sociaux et gestion de communautés - applications dans le domaine de la gestion de la relation client

Ian Basaille Basaille-Gahitte

SPIM

Thèse de Doctorat

école doctorale **sciences pour l'ingénieur et microtechniques**

UNIVERSITÉ DE BOURGOGNE

Plateforme pour la gestion des données issues des réseaux sociaux dans le cadre de la gestion de la relation client

IAN BASAILLE-GAHITTE

SPIM

Thèse de Doctorat

école doctorale **sciences pour l'ingénieur et microtechniques**

UNIVERSITÉ DE BOURGOGNE

N° X X X

THÈSE présentée par

IAN BASAILLE-GAHITTE

pour obtenir le

Grade de Docteur de
l'Université de Bourgogne

Spécialité : **Informatique**

Plateforme pour la gestion des données issues des réseaux sociaux dans le cadre de la gestion de la relation client

Unité de Recherche :
Laboratoire Électronique, Informatique et Image - CNRS FRE 2005
CNRS, Arts et Métiers, Univ. Bourgogne Franche-Comté

Soutenue publiquement le 9 février 2018 devant le Jury composé de :

MARIE-CHRISTINE FAUVET	Rapporteur	Professeur à l'Université de Grenoble Alpes
BERNARD ESPINASSE	Rapporteur	Professeur à l'Université Aix-Marseille
ZOHRA BELLAHSENE	Examinateur	Professeur à l'Université de Montpellier
ALLEL HADJALI	Examinateur	Professeur à l'ENSMA de Poitiers
NADINE CULLOT	Directeur de thèse	Professeur à l'Université de Bourgogne
ERIC LECLERCQ	Co-encadrant	Maître de conférences à l'Université de Bourgogne

Touching from a distance

So this is permanence

Reality is only a term, based
on values and well worn principles,
whereas the dream goes on forever.

- Ian Curtis, handwritten note, circa 1979

Don't ever fade away

REMERCIEMENTS

Je tiens tout d'abord à remercier Nadine Cullot et Eric Leclercq, qui m'ont encadré tout au long de ma thèse. Leur aide et leurs expériences respectives dont ils ont su me faire profiter ont été déterminantes dans l'aboutissement de mes travaux.

Je tiens à exprimer toute ma gratitude à Marie-Christine Fauvet et Bernard Espinasse qui ont accepté d'être les rapporteurs de mon travail, ainsi qu'à Zohra Bellahsene et Allel Hadjali qui ont bien voulu faire partie de mon jury.

Je remercie Marinette Savonnet pour avoir relu ce manuscrit et pour l'aide qu'elle m'a apporté tout au long de ma thèse. Mes remerciements vont également aux membres du Le2i que j'ai pu côtoyer, avant ou pendant ma thèse, et dont j'ai pu apprécier les conseils : Jean-Luc Baril, Albert Dipanda, Elisabeth Gavignet, Thierry Grison, David Gross-Amblard, Marc Neveu, Denis Pellion, Marie-Noëlle Terrasse, Olivier Togni et Kokou Yétongnon.

Je tiens à remercier Emmanuel Mignot de la société eb-Lab de s'être engagé dans ce challenge et de m'avoir soutenu tout au long du projet, et de m'avoir donné la possibilité de le poursuivre après la fin du contrat CIFRE. Je remercie aussi l'ANRT d'avoir financé ce travail dans le cadre d'une bourse CIFRE numéro 2012 / 0261.

Mes remerciements vont également à l'ensemble des équipes d'eb-Lab et de Teletech International, avec qui j'ai pris beaucoup de plaisir à travailler et échanger ; particulièrement Yohann Pansard, Maxime Barreau, Bruno Fernandès, Abderrahim Azmou, Rémy Anceau, Elie Testard, Jean-Michel Barbier, Guillaume Despret, Loïc Lucien, Régis Träger et Kevin Vieille.

Je remercie également mes collègues de la Caisse Primaire d'Assurance Maladie de la Côte d'Or et de la Caisse Nationale d'Assurance Maladie pour m'avoir permis de prendre le temps nécessaire à la finalisation de ce manuscrit et à la prépartion de sa soutenance.

Je tiens à remercier également Sergey Kirgizov, Armen Petrossian et Wahabou Abdou pour l'aide précieuse qu'ils m'ont apportée durant ma thèse, et pour nos nombreuses discussions.

Je remercie aussi chaleureusement Clémence Ménis pour avoir relu des parties de ce manuscrit, et pour toutes nos conversations qui m'ont profondément enrichi.

Merci à Cécile Tugler pour m'avoir soutenu, éclairé, et pour avoir changé ma vie.

Comme des pages sans reliure, cette thèse et tant d'autres choses se seraient disséminées sans Hervé Abdi, que je ne remercierai jamais assez, et sans qui je ne serai pas qui je suis aujourd'hui.

Enfin, j'exprime ma profonde gratitude à mes parents pour leur inspiration, leur soutien, et pour m'avoir permis d'aller aussi loin.

SOMMAIRE

TABLE DES FIGURES

LISTE DES TABLES

1

INTRODUCTION

Sommaire

La gestion de la relation client (GRC, ou CRM pour *Customer Relationship Management*) est un composant fonctionnel essentiel des systèmes d'information des entreprises et a été reconnue comme un facteur critique de leur réussite. Elle consiste en la création, la maintenance et l'amélioration de relations génératrices de valeur et de satisfaction entre les entreprises et les consommateurs. Avec l'explosion, depuis dix ans, de la popularité et de l'usage des réseaux sociaux numériques et des plateformes collaboratives, le métier de la relation client a évolué pour prendre en compte ces nouveaux canaux de communication. De nouveaux métiers sont également apparus, par exemple les gestionnaires de communautés (*community manager*), et les métiers de conseiller client ou télé-conseiller ont évolué. Les entreprises souhaitent pouvoir suivre leurs clients, connaître ce qui est dit d'une marque ou d'un produit sur les réseaux sociaux, identifier les communautés d'utilisateurs liés à une marque, diffuser des offres promotionnelles, améliorer les services avant et après-vente, etc. Par conséquent, un système de gestion de la relation client moderne se doit d'être connecté aux réseaux sociaux numériques afin de collecter des données sur les clients ou prospects, analyser leurs comportements et publier des informations en utilisant des relais ou des mots-clés pertinents.

1.1/ CONTEXTE DE LA THÈSE

L'introduction d'une composante réseau social dans la gestion de la relation client apporte aux entreprises de nouvelles perspectives en matière d'analyse du comportement des consommateurs, d'action et de définition des stratégies. Du point de vue des analyses, la quantité de données disponibles doit permettre de mieux connaître les communautés d'utilisateurs liés à l'entreprise, de suivre les conversations sur une marque ou un produit, d'évaluer l'impact sur les réseaux sociaux de campagnes marketing ou d'événements particuliers. En plus de la connaissance qu'il est possible d'extraire des données des réseaux sociaux, il est également possible d'agir sur des éléments importants du réseau pour leur diffuser des informations ciblées : soit en détectant des personnes influentes capables de mobiliser une communauté, soit en utilisant des mots-clés spécifiques qui vont diriger les informations diffusées vers la communauté la plus pertinente. Cependant, au-delà des algorithmes d'analyse de données, la construction d'outils utilisables par des gestionnaires de communautés pour les aider à cibler les données et faciliter l'interprétation des résultats est un verrou important. Ces outils permettront l'adaptation de la gestion de la relation client aux nouveaux moyens de communication et la valorisation des données issues des réseaux sociaux.

Ma thèse s'est déroulée en contrat CIFRE dans la société eb-Lab [1], en collaboration avec le laboratoire LE2I [2] de l'Université de Bourgogne. eb-Lab a pour activité principale le développement de solutions logicielles innovantes pour la gestion de la relation client. Ainsi, cette entreprise a une composante recherche et développement très importante. Le principal client d'eb-Lab est la société Teletech International [3] (aussi appelée TTKI). Cette dernière intègre la dimension CRM dans ses produits et propose : 1) des solutions de centre d'appels spécialisés, par exemple pour la maintenance de stations-service de groupes pétroliers ; 2) des plateformes logicielles collaboratives incluant des bases de connaissances pour des grands comptes. Dans ce contexte, eb-Lab fournit à Teletech

1. http://eb-lab.com/
2. http://le2i.cnrs.fr/
3. https://www.teletech-int.com/

International deux types de prestations :

— des spécifications détaillées pour des modèles de données, et des algorithmes d'analyse à destination des équipes de développement de TTKI ;
— un framework de plateforme générique pour la collecte, le stockage et l'analyse de données de réseaux sociaux incluant des composants logiciels en mode service Web que TTKI adapte, spécialise ou revend à ses clients.

Durant mes trois années de contrat CIFRE et l'année qui a suivi, j'ai développé ces deux types de prestations au sein de la société eb-Lab. Suite au rachat de TTKI en mai 2016, eb-Lab a cessé son activité de recherche et de développement en juillet 2016.

1.2/ PROBLÉMATIQUES ABORDÉES ET CONTRIBUTIONS

Les problématiques scientifiques abordées durant la thèse concernent la collecte, le stockage et l'analyse de grandes quantités de données. La partie analyse est la plus importante et nous nous concentrons plus particulièrement sur les méthodes d'analyse contextuelle de données à grandes dimensions. Dans le cadre de mes travaux de thèse, les données traitées par l'entreprise eb-Lab sont :

— Des données issues de plateformes collaboratives spécialisées, c'est-à-dire développées spécifiquement pour des clients liés à domaine métier particulier, par exemple pour fédérer les acteurs de la nutrition et de la santé. Ces données ont un volume maximum de quelques centaines de giga-octets mais les utilisateurs et les ressources sont décrits avec plusieurs dizaines de caractéristiques. Ces données peuvent être qualifiées de données à grandes dimensions en raison du nombre de caractéristiques.
— Des données collectées à partir des réseaux sociaux pour un besoin spécifique, par exemple pour suivre les évolutions du co-voiturage et les services de désintermédiation proposés par des sociétés comme BlaBlaCar [4] ou id-Vroom [5]. Ces données ont un volume très variable, allant de quelques giga-octets à quelques tera-octets. Les caractéristiques descriptives de ces données sont moins nombreuses que celles produites par les plateformes collaboratives. En revanche, les liens entre les données, générés par les interactions des utilisateurs, sont très nombreux. Ces données sont qualifiées de données à grandes dimensions en raison de leur volume et du nombre de liens.

Les problématiques associées à ces données sont pour l'entreprise : 1) leur collecte qui doit être la plus exhaustive possible, tolérante aux pannes (comme des interruptions réseaux), et les outils de collecte mis en œuvre doivent être capables d'absorber des flux très importants ; 2) la minimisation, pour ce qui est du temps de développement, de la transformation des données pour l'alimentation des algorithmes d'analyse.

Concernant l'analyse de données, la problématique que nous traitons est relative à l'interprétabilité des données et des résultats. Plus précisément, il s'agit d'exploiter au mieux l'ensemble des données à disposition pour fournir un mécanisme de détection de communautés à partir de données contextualisées, et permettre une caractérisation sémantique ces communautés. En relation avec les communautés, la détection des utilisateurs influents et leur caractérisation est un élément complémentaire important. De même, la notion d'événement ainsi que sa détection permettent, lors d'analyses, exploratoires de

4. https://www.blablacar.fr/
5. https://www.idvroom.com/

cerner une période sur laquelle effectuer des détections de communautés. Cependant, sans une définition et une caractérisation sémantique complémentaire de l'événement, l'apport pour l'utilisateur est faible.

Les contributions de la thèse sont multiples et s'organisent selon trois axes : le premier concerne les modèles de données et leur contextualisation par la sémantique du domaine métier ; le second est relatif aux algorithmes d'analyse des données ; le troisième, plus technologique, concerne le développement d'une plateforme logicielle orientée CRM pour la collecte, le stockage et l'analyse de données issues des réseaux sociaux.

Les deux premiers axes sont fortement liés. Il s'agit tout d'abord d'une méthode de détection de communautés exploitant un modèle de profil thématique d'utilisateur. Cette méthode s'applique pour des plateformes collaboratives, gérées par une entreprise ou hébergées chez un client, à destination de ses employés (de quelques centaines à quelques milliers). Les utilisateurs de la plateforme sont modélisés par de multiples caractéristiques, dépendantes de leur métier. La méthode utilise, en plus des profils, une modélisation de la sémantique du domaine. Cette modélisation prend la forme d'une ontologie d'application, réduite à une hiérarchie de termes dans les expérimentations. Les algorithmes utilisés pour la détection de communauté sont : l'algorithme des k-moyennes pour rechercher des communautés à partir de profils simples ; l'algorithme de Louvain [Blondel et al., 2008] pour l'optimisation de la modularité, appliqué à des graphes regroupant des utilisateurs, des mots-clés (tags) et dans lesquels sont injectés des éléments de la connaissance du domaine. Cette approche permet de caractériser rapidement une communauté par les mots-clés du domaine et fournit un support visuel pour les utilisateurs. Afin de préciser les résultats de l'algorithme de découverte de communautés globales sur le graphe et de contourner la limite de détection des algorithmes d'optimisation de modularité, nous modifions l'algorithme CarOp (*Carry over Opinion*) proposé par M. Danisch dans [Danisch et al., 2013]. À partir des nœuds d'intérêt (utilisateurs ou mots-clés), nous appliquons un algorithme de propagation pour détecter la ou les communautés formées autour de ces nœuds. Dans l'optique de mieux maîtriser la convergence de l'algorithme, nous avons remplacé la méthode de propagation, par une propagation s'inspirant d'une marche aléatoire de type *random surfer* proposée dans l'algorithme PageRank et aboutissant, dans notre cas, à une mesure de proximité des nœuds d'un graphe par rapport au(x) nœud(s) d'intérêt. Les expérimentations de la méthode et des algorithmes proposés se sont appuyées sur la plateforme collaborative développée par l'entreprise TTKI pour Vitagora [6], le pôle de compétitivité "Goût-Nutrition-Santé" de l'ancienne région Bourgogne. La méthode, les algorithmes et les résultats ont fait l'objet des publications suivantes [Basaille-Gahitte et al., 2013a, Basaille-Gahitte et al., 2013b, Basaille-Gahitte et al., 2014]. Ces contributions sont détaillées dans le chapitre 4 et illustrées dans le chapitre 5. Du point de vue appliqué, les spécifications et les algorithmes ont été transférés au pôle développement de TTKI pour être utilisés au sein de leurs plateformes logicielles afin d'améliorer la composante CRM, d'effectuer des recommandations d'articles ou de contacts, et de diffuser des informations pertinentes aux communautés.

La seconde contribution est un algorithme de détection et de caractérisation d'événements à partir de données extraites de Twitter, en relation avec des marques, des produits ou des services. Notre proposition, contrairement aux approches statistiques, ne fait pas d'hypothèse sur la loi associée aux séries temporelles et ne nécessite qu'un seul

6. http://www.vitagora.com/

paramètre facile à appréhender. Cet algorithme permet notamment d'effectuer des analyses exploratoires d'un jeu de données. Il est décrit et illustré dans le chapitre 5, et les résultats ont été publiés dans [Basaille-Gahitte et al., 2016, Basaille-Gahitte et al., 2017].

La troisième contribution relève d'un axe de recherche plus technologique. Il s'agit d'une plateforme pour la collecte et l'analyse de données Twitter, déclinée sous deux formes : un prototype de recherche et une plateforme générique développée selon les standards de l'entreprise eb-Lab. Dans le prototype, nous avons validé les propriétés de reprise sur panne, de passage à l'échelle des mécanismes de collecte de Twitter, ainsi que le stockage des tweets dans un *polystore*. Ce dernier correspond à une couche qui permet d'abstraire les systèmes de stockage NoSQL et les systèmes de gestion de bases de données relationnels, et d'aiguiller les données vers le système de stockage le plus adapté aux flux et aux algorithmes d'analyse. Ce système a été validé sur des grandes quantités de données, par exemple la coupe du monde de football 2014 (pour un volume de 3,2 tera-octets de données) et a servi à plusieurs projets de l'équipe du recherche du LE2I. La plateforme a aussi servi de preuve de concept pour montrer l'intérêt d'une approche itérative et incrémentale d'analyse des données des réseaux sociaux à destination d'utilisateurs non spécialistes de l'analyse de données mais experts métier. Les publications relatives à cette contribution sont [Basaille-Gahitte et al., 2016, Leclercq et al., 2016, Basaille-Gahitte et al., 2017]

1.3/ ORGANISATION DU DOCUMENT

Le document est organisé en cinq chapitres. Après avoir précisé la définition et le périmètre de la gestion de la relation client, le chapitre 2 détaille le contexte et les problématiques abordées dans la thèse dans le cadre de la coopération avec l'entreprise eb-Lab.

Le troisième chapitre présente un état de l'art centré sur la détection de communautés. Dans un premier temps les principaux modèles de réseaux complexes sont définis, puis deux sections abordent la détection de communautés selon l'angle de la classification et selon l'angle de la théorie des graphes. Pour chacune des approches, les bases théoriques sous-jacentes sont exposées afin de montrer les limites des algorithmes et leurs conditions d'application. De même, pour chacune des deux catégories, les principales méthodes d'évaluation des résultats des algorithmes sont passées en revue, que ce soit avec des critères intrinsèques ou bien avec une vérité de terrain. Dans une section discussion, nous montrons l'intérêt d'utiliser plusieurs types d'algorithmes pour éclairer, selon différents points de vue, une analyse.

Le quatrième chapitre détaille notre contribution à la détection des communautés. La sémantique, considérée dans notre cas comme la contextualisation des données ou des résultats des algorithmes par rapport à la connaissance du domaine, ainsi que la modélisation des profils utilisateurs, sont les deux fils conducteurs du chapitre. Après une synthèse des modèles de profil utilisateur et de leur utilisation, nous décrivons notre modèle de profil thématique pour des utilisateurs d'une plateforme CRM. Nous montrons comment un profil thématique détaillé entre dans un mécanisme de détection de communautés. En prenant en compte des données non plus issues d'une plateforme CRM mais aussi celles issues des réseaux sociaux, nous montrons comment la combinaison d'algorithmes de détection de communautés globales et locales permet une caractérisation

sémantique fine en fonction des mots-clés utilisés.

Le cinquième chapitre décrit la plateforme générique pour la gestion de données issues de réseaux sociaux. Cette plateforme permet de gérer la collecte, le stockage ainsi que l'analyse et la visualisation des données issues des réseaux sociaux. Afin de pouvoir exploiter les données, plusieurs outils d'analyse et de visualisation doivent être intégrés ou couplés avec la plateforme. Ils permettent aussi de supporter des analyses exploratoires rapides des données recueillies. La collecte de données pouvant s'étaler sur de longues durées, un système de reprise sur panne doit être intégré à la plateforme afin de prévenir une défaillance. Un *proof of concept* de cette plateforme, nommée SNFreezer, a été développé et a permis de valider plusieurs des fonctionnalités au travers de différents projets. Dans ce chapitre, nous décrivons l'architecture de la plateforme SNFreezer, et détaillons les fonctionnalités principales que sont la collecte et le stockage polyglotte des données (*polyglot storage*), aussi désigné sous le terme de *polystore*. Nous décrivons ensuite comment ces fonctionnalités ont été validées au travers de trois projets, puis nous abordons par des exemples d'utilisation les outils d'analyse avant d'établir un comparatif avec les plateformes concurrentes existantes et de faire une synthèse des leçons tirées de l'implémentation de cette plateforme.

La plateforme SNFreezer a servi de point de départ pour une implémentation plus industrielle décrite dans le sixième chapitre qui précise les objectifs du projet DisCoCRM et les fonctionnalités requises. Le chapitre présente les différents cas d'utilisation de la plateforme, ainsi que son architecture technique et les différences par rapport à SNFreezer. L'organisation et les phases du projet ainsi que l'environnement technique sont décrits. Les contraintes et la réalisation de l'entrepôt de données, l'intégration des algorithmes et outils d'analyse au sein de la plateforme sont détaillés.

Le dernier chapitre résume les contributions de la thèse, effectue un bilan et présente les perspectives de notre travail.

2

CONTEXTE ET PROBLÉMATIQUE

Sommaire

Dans ce chapitre, nous décrivons le contexte métier de la thèse et nous précisons la problématique associée à la gestion de la relation client. Dans une première section, nous définissons la gestion de la relation client, ses enjeux et les outils logiciels sur lesquels elle s'appuie. Dans la deuxième section nous décrivons l'impact des technologies du Web 2.0, c'est-à-dire des média-sociaux et des plateformes collaboratives, sur la gestion de la relation client, tout en identifiant les enjeux quant aux outils logiciels et à l'implication dans les systèmes d'information des entreprises. Les deux dernières sections de ce chapitre sont centrées sur l'entreprise partenaire de la thèse et décrivent respectivement son approche de la gestion de la relation client, les outils existants et les évolutions de ces outils réalisées et prévues dans un contexte Web 2.0 ainsi que les fonctionnalités requises qui en découlent.

2.1/ LA GESTION DE LA RELATION CLIENT

La *gestion de la relation client* (GRC), ou *CRM* pour *Customer Relationship Management* en anglais, est un terme ayant émergé au milieu des années 1990, souvent utilisé pour décrire des outils informatisés de service client offrant des prestations de services à des consommateurs avant, pendant et après un achat [Payne et al., 2005]. Dans les paragraphes suivants, nous précisons la notion de relation client puis nous identifions ses enjeux. Dans la suite du document, nous utiliserons l'acronyme CRM au masculin pour désigner l'ensemble des outils, méthodes et techniques qui contribuent à la gestion de la relation client.

2.1.1/ DÉFINITIONS

La vision que les individus ont de la gestion de la relation client est très variée, que ce soit pour ceux qui sont la cible ou pour ceux qui opèrent la relation client. Ainsi pour certains, cela peut se résumer à un courrier, une carte fidélité, tandis que d'autres la verront comme un centre d'assistance ou un centre d'appels ; ou encore, dans le cadre d'une plateforme de commerce sur le Web, un moteur de personnalisation de contenu Web, ou de recommandation de produits. Du point du vue du système d'information, il est possible de le voir comme un entrepôt de données associé à des techniques de fouilles de données (*datamining*). Ce manque de définition claire et largement acceptée par les acteurs tend à créer de la confusion lors d'échanges sur ce sujet. Ceci peut être un facteur d'échec des projets de mise en place d'un CRM. Nous allons reprendre plusieurs définitions du CRM, chacune éclairant un point de vue particulier, organisationnel, métier ou technique.

[Kumar, 2010] définit la gestion de la relation client comme la construction de relations personnalisées avec les clients d'une entreprise afin de créer de la valeur pour cette dernière. [Knox et al., 2007] la définissent comme un processus, ayant pour champ d'action l'entreprise dans sa globalité, qui se concentre sur le traitement différent de chaque client afin de créer de la valeur, que ce soit pour le client ou pour l'entreprise ; et qui inclut une stratégie d'acquisition, de croissance et de conservation des *bons* clients.

[Payne et al., 2005] présentent la gestion de la relation client de trois manières bien distinctes, qui peuvent être mises dans un continuum d'approches :

1. de manière restrictive et tactique : c'est alors l'implémentation d'une technologie particulière ;
2. de manière centrée sur le consommateur : c'est alors l'implémentation d'un ensemble intégré de technologies orientées autour du consommateur ;
3. de manière large et stratégique : c'est alors une approche sélective de la gestion des relations avec les consommateurs afin de créer de la valeur.

La façon qu'a une entreprise d'aborder le CRM n'est pas qu'une question de sémantique mais affecte aussi la mise en place des outils, les stratégies de création de valeur ainsi que les pratiques développées autour du CRM. D'un point stratégique, le CRM ne se réduit pas à une collection d'outils informatisés, mais nécessite une vision stratégique et une compréhension de la nature de la création de valeur dans un environnement multicanal (ou multi-source d'information) ainsi que l'utilisation des informations et des applications appropriées afin d'obtenir un haut niveau de qualité de service et de satisfaction.

[Swift, 2001] et [Payne et al., 2005] mettent en avant les bénéfices que les entreprises ont à adopter l'approche stratégique du CRM, afin d'assurer son utilisation cohérente au sein de l'entreprise. [Payne et al., 2005] proposent ainsi leur définition du CRM : une approche stratégique qui s'intéresse à la création de valeur à travers le développement de relations adéquates avec des consommateurs clés ainsi que des segments de consommateurs. Il combine le potentiel des stratégies de relation marketing et des technologies de l'information pour créer des relations profitables et de long terme avec les consommateurs. Il fournit une opportunité d'utiliser des données afin de mieux comprendre les consommateurs. Cela nécessite une intégration transverse des processus, des personnes, des opérations et des capacités marketing qui est rendue possible par l'information, la technologie et les applications.

Les travaux présentés dans [Winer, 2001], dans une approche orientée outils, identifient sept composants de base permettant de développer une démarche complète afin de capitaliser et de développer la connaissance qu'une entreprise a des consommateurs :

1. une base de données contenant l'activité des consommateurs ;
2. des analyses sur les données ;
3. en fonction des analyses, des décisions ;
4. des outils pour cibler ces consommateurs ;
5. des méthodes pour créer des relations avec ces consommateurs ciblés ;
6. des méthodes et outils pour garantir le respect de la vie privée ;
7. des métriques pour mesurer l'efficacité du CRM.

Les travaux présentés par [Ryals et al., 2000] décrivent les objectifs du CRM et dressent les grandes lignes des éléments techniques pour y répondre. Ainsi, ils identifient trois objectifs principaux :

1. identifier, satisfaire, garder et maximiser la valeur des meilleurs clients d'une entreprise ;
2. envelopper l'entreprise autour des clients pour s'assurer que chaque contact avec eux soit approprié et basé sur une connaissance approfondie, que ce soit des besoins des clients ou de la rentabilité ;
3. créer une image la plus complète possible de la clientèle.

Ils identifient également cinq composants essentiels pour une implémentation réussie d'une solution CRM :

1. un *front office* qui comprend les ventes, le marketing à travers plusieurs médias (téléphone, plateforme Web, personnes en contact direct ou en face à face, etc.) ;
2. un entrepôt de données pour stocker les données et les informations sur les clients ainsi que des outils d'analyse appropriés pour extraire de la connaissance sur le comportement des clients ;
3. des règles commerciales développées à travers l'analyse des données pour s'assurer que le *front office* bénéficie des connaissances acquises ou extraites ;
4. des mesures de performance pour une amélioration continue ;
5. une intégration avec les systèmes opérationnels et support (*back office*), pour s'assurer que les objectifs du front office soient tenus.

Il n'y a donc pas de consensus clair sur la définition du CRM si ce n'est un objectif large de construction et de maintien de relations profitables avec les clients tout en fournissant une valeur et une satisfaction accrues à la clientèle afin d'améliorer les relations d'affaires avec les clients [Soltani et al., 2016]. Cependant, dans un contexte de système d'information, nous pouvons identifier à partir des différentes définitions des éléments techniques ou des mécanismes essentiels pour atteindre les objectifs suivants :

— la construction d'un entrepôt pour rassembler les données et les connaissances sur les utilisateurs ou les clients ;
— la collecte de données à partir de différentes sources, dont les réseaux sociaux ;
— l'analyse des comportements des clients.

2.1.2/ ENJEUX

Traditionnellement, les services marketing des entreprises étaient chargés d'attirer de nouveaux clients, que ce soit ceux qui n'ont pas acheté de produit de l'entreprise avant, ou ceux qui sont déjà clients des entreprises concurrentes. Les moyens utilisés étaient principalement des campagnes publicitaires de masse et des promotions sur les prix des produits. Avec la multiplication des canaux de communication et de distribution des offres des entreprises, ainsi que la révolution numérique et l'avènement d'Internet avec l'instantanéité des échanges qu'ils ont apporté, de nombreux changements se sont produits. Non seulement les moyens à la disposition des services marketing ont fortement évolué ; mais les attentes des consommateurs [1] ont changé, incluant une dynamique et un engagement plus importants [Harrigan et al., 2014]. De nouveaux problèmes se posent alors pour les services marketing, qui doivent s'adapter à l'évolution des canaux de communication et aux nouvelles habitudes des consommateurs :

— Quelle communication émettre ?
— Destinée à qui ?
— Quand ?

La grande quantité de données disponibles offre aussi de nouvelles possibilités pour optimiser les stratégies marketing. En effet, leur exploitation et leur analyse offrent la possibilité d'en faire ressortir de nouvelles connaissances sur les consommateurs, permettant

1. On utilise la terme consommateur dans le sens général, le terme client pour préciser qu'il s'agit d'un consommateur qui a déjà acheté un produit de la marque et le terme prospect pour désigner un consommateur inclus dans une campagne marketing.

par exemple de répondre à la problématique de trouver le meilleur moment et le meilleur canal de communication pour engager une communication avec les consommateurs, ou avec une partie plus ciblée des consommateurs.

Cependant, l'augmentation de la quantité de données produites par les interactions entre les entreprises et les consommateurs pose des difficultés au niveau de leur analyse et de leur interprétation. Cette augmentation concerne à la fois le volume des données, mais aussi leur diversité.

Les réseaux sociaux viennent renforcer cette situation. Ils permettent de suivre et d'analyser le comportement des consommateurs et des clients sur le Web, afin par exemple de prédire leurs démarches futures. Ces prédictions peuvent permettre de leur envoyer des communications directes, par email ou via les systèmes de messagerie privée des réseaux sociaux. Néanmoins, dans la masse des données générées par les réseaux sociaux, déterminer et extraire les bonnes données ; puis choisir, appliquer les algorithmes et interpréter les résultats afin d'effectuer des actions pertinentes est un véritable enjeu.

2.2/ LA TRANSFORMATION DU WEB

Depuis quelques années, le Web s'est transformé en une plateforme d'échange générique, où tout utilisateur devient fournisseur de contenu au moyen d'outils comme les blogs avec commentaires, les wikis avec les fonctionnalités de collaboration et de contribution, ou encore les réseaux sociaux avec le partage de ressources, de contenu et les mécanismes d'annotation.

Il existe de nombreux médias sociaux, de natures différentes, répondant à des besoins variés et ayant des modes d'utilisation et d'interaction entre utilisateurs propres ainsi qu'un langage parfois spécifique [2].

Les réseaux sociaux grand public, tels que *Facebook* [3] ou *Twitter* [4], sont maintenant utilisés quotidiennement par un très grand nombre d'utilisateurs [5]. Certains réseaux sociaux sont spécialisés pour un type de contenu, comme *Flickr* [6], ou apportent des fonctionnalités pour un usage spécifique comme *Pinterest* [7]. Twitter et Facebook restent très généralistes, et l'usage des fonctionnalités proposées n'est pas le même en fonction des buts recherchés. En effet, *Twitter* peut être utilisé comme une messagerie instantanée, comme un canal thématique de diffusion d'informations, ou encore comme un moyen de débattre autour de sujets de société, de produits, de marques, etc.

La nécessité de mieux comprendre le comportement des consommateurs et des clients ; ainsi que l'intérêt prononcé de nombreux gestionnaires de la relation client de se concentrer sur les clients qui peuvent générer des bénéfices à long terme, particulièrement lorsqu'il s'agit des *meilleurs* clients d'une entreprise, ont changé le rôle des services marketing. Ceux-ci sont passés d'un rôle reposant principalement sur l'acquisition de nouveaux

2. *Twitter* avec le retweet par exemple
3. https://www.facebook.com/
4. https://twitter.com/
5. Au 30 juin 2017, Facebook compte 2,01 milliards d'utilisateurs actifs au moins une fois par mois ; et, en moyenne sur juin 2017, 1,32 milliard d'utilisateurs actifs au moins une fois par jour. https://newsroom.fb.com/company-info/
6. Média social spécialisé dans le partage de photographies. https://www.flickr.com/
7. Site Web mélangeant les concepts de réseau social et de partage de photographies. https://fr.pinterest.com/

clients, à la fidélisation des clients déjà existants. Ce changement majeur nécessite une manière de fonctionner différente, ainsi qu'un ensemble d'outils nouveaux et adaptés à ce nouveau rôle.

2.2.1/ IMPACTS SUR LA GESTION DE LA RELATION CLIENT : LE *Social CRM*

Le Web en général, et plus particulièrement le Web 2.0 et les réseaux sociaux, permettent aux entreprises de choisir comment elles interagissent avec leurs clients et prospects, et réciproquement.

Du point de vue des entreprises, les réseaux sociaux permettent de répondre directement aux demandes des clients et de leur fournir une réelle interactivité personnalisée qu'il est difficile d'obtenir avec les courriers électroniques, ou avec les campagnes de publicité de masse. La présence sur les réseaux sociaux est également une question d'image. Les entreprises sont conscientes qu'elles doivent être présentes sur les réseaux sociaux. Elles investissent donc de plus en plus dans la création de profils (ou pages) sur les différents médias sociaux et embauchent des spécialistes pour les gérer. Cette présence leur permet de partager du contenu, parfois spécifique pour un média social en particulier ; de toucher facilement et rapidement un nombre de consommateurs très important ; et de constituer des bases de données sur les interactions entre l'entreprise et les clients, et entre les consommateurs entre eux sur les profils des entreprises.

L'explosion du nombre d'entreprises de e-commerce depuis le début des années 2000 a bouleversé le commerce en général. Le *social commerce*, c'est-à-dire, l'achat de biens et services utilisant le levier d'un réseau social [Rignault et al., 2012], est une évolution du e-commerce s'appuyant sur l'évolution du Web en Web Social. Ces nouveaux usages du Web 2.0 et la richesse des données sociales apportent des promesses importantes quant au développement de la connaissance sur les clients et les consommateurs [Melville et al., 2009].

Du point de vue des clients, les réseaux sociaux généralistes sont utilisés pour commenter des produits de marques ; obtenir un avis ; échanger des conseils sur des produits ; demander de l'aide sur leur fonctionnement, que ce soit à la marque ou aux autres consommateurs ; interpeller la marque sur des problèmes rencontrés, etc. Par exemple, Twitter est de plus en plus utilisé pour le service après-vente, poser des questions, demander des informations, notifier des problèmes, etc.

Grâce aux réseaux sociaux, les entreprises ont aujourd'hui une plus grande capacité à établir, nourrir, et maintenir les relations clients à long terme. Mais les clients cibles du CRM ne sont plus isolés. Ils communiquent entre eux et forgent leur opinion sur un produit, une marque ou un service à partir des leurs interactions. De ce fait, la lecture des commentaires sur les sites de e-commerce ou les sites spécialisés n'est plus le seul moyen de se faire une opinion avant de consommer. De plus, la viralité de certains événements sur les réseaux sociaux peut également être un danger. Le CRM doit donc intégrer complètement cette dimension des réseaux sociaux.

2.2.2/ ÉVOLUTION DES OUTILS DE GESTION DE LA RELATION CLIENT

Du point de vue de l'entreprise, l'offre d'outils de CRM dit *classique* est vaste. Ces outils visent à proposer des fonctionnalités permettant d'améliorer la gestion des services com-

merciaux, marketing ou après-vente et incluent généralement des méthodes d'analyse, notamment statistiques. Cependant, les transformations induites par les technologies du Web 2.0 font évoluer les outils CRM vers une meilleure prise en compte de la dimension sociale des échanges entre les clients et la société ou entre les clients eux-mêmes vis-à-vis de la société.

Ainsi, le terme *Social CRM* ou *s-CRM* succédant au terme *e-CRM* [Harrigan et al., 2014], est associé à l'utilisation des médias sociaux dans le cadre d'outils de gestion de la relation client [Mohan et al., 2008]. On distingue généralement trois catégories d'outils CRM : généralistes, intégrables et génériques.

— Les outils *généralistes* sont des logiciels dits « sur étagère » et sont plutôt destinés aux petites et moyennes entreprises. Les logiciels *SugarCRM* et *SalesForce* sont des exemples de cette catégorie ayant des fonctionnalités proches. *SugarCRM* propose des fonctionnalités classiques des CRM (gestion de la relation commerciale, marketing, service client, outils d'analyse), mais aussi quelques fonctionnalités collaboratives pour la vente et l'intégration de contacts depuis les médias sociaux.
— Les outils *intégrables* sont des modules logiciels destinés à s'interconnecter avec le système d'information de l'entreprise. Ils peuvent avoir des fonctionnalités plus ciblées comme l'analyse ou la fouille de données. C'est le cas par exemple de logiciels comme *Smarter Analytics* d'IBM, qui permet l'analyse de données à des fins décisionnelles au sein d'une entreprise.
— Les outils *génériques* sont des logiciels développés pour être paramétrables et adaptables aux besoins des sociétés. L'offre peut être modulaire et toucher tous les domaines de la gestion de la relation client. Ces derniers peuvent plus facilement être étendus pour évoluer vers le *Social CRM*.

Comme nous l'avons mis en évidence dans les sections précédentes, la compréhension des mécanismes d'interaction entre une entreprise et ses clients, la diffusion de l'information entre les clients ou futurs clients, ainsi que la connaissance sur les profils des consommateurs sont des éléments essentiels pour la croissance et la compétitivité. Par conséquent, un s-CRM doit fournir des modèles de données adaptés au stockage des profils des utilisateurs et de leurs interactions. Cependant, le volume des données produites par les réseaux sociaux est important et le s-CRM doit proposer des moyens pour se connecter aux réseaux sociaux, sélectionner les données nécessaires et absorber le flux.

Les logiciels CRM doivent évoluer non seulement pour pouvoir analyser le comportement des consommateurs aussi bien en tant que clients, prospects mais aussi comme acteurs qui participent à la *e-réputation* d'une entreprise. Le suivi de cette e-réputation doit être intégrée dans le s-CRM. Elle concerne aussi bien une marque que ses produits, ses services et ses dirigeants. Ainsi, il est important de savoir qui parle d'une entreprise, la portée des opinions émises, et de pouvoir réagir le cas échéant [Cordina et al., 2013]. En conclusion, le périmètre n'est plus uniquement celui des clients ; et avant même l'analyse de données, il convient de spécifier les données qui seront collectées et selon quels critères.

Les impacts sur l'architecture logicielle des CRM concernent non seulement l'interconnexion avec les applications traditionnelles des SI d'entreprise, avec des outils collaboratifs tels les réseaux sociaux d'entreprises [8] mais aussi l'interconnexion avec les médias

8. Il existe plusieurs outils spécifiques qualifiés de réseaux sociaux d'entreprise, par exemple *Yam-*

sociaux grand public [Ajmera et al., 2013]. La notion de communauté est un élément central des s-CRM, les principales fonctionnalités liées à leur analyse peuvent être classées en trois catégories :

- **la détection de communautés** dans les populations de clients ou de prospects. D'un point de vue algorithmique, la détection de communautés produira une définition en extension permettant d'obtenir une vue macroscopique d'un système complexe, utilisée ensuite dans la compréhension et l'analyse du système. L'approche duale est l'identification des personnes extérieures à une communauté, afin d'analyser leurs profils ou afin de déterminer la communauté qui leur est la plus proche ;
- **l'analyse de communautés** ou leur caractérisation par une définition en intention, c'est-à-dire au moyen de propriétés caractéristiques éventuellement hiérarchisées. Les méthodes et algorithmes utilisés doivent permettre de faire émerger une sémantique à partir des données des éléments de la communauté. L'analyse peut faire appel à des concepts de socio-psychologie pour la définition de profils utilisateur, par exemple l'endoreprésentation [9] ou l'exoreprésentation [10] [Perrin, 2011] ;
- **l'analyse des flux d'information** intra-communauté et inter-communautés pour identifier les personnes influentes, afin de diffuser au mieux des informations de l'entreprise, mais aussi d'effectuer une prédiction.

2.3/ PRÉSENTATION DE L'ENTREPRISE EB-LAB

La société eb-Lab est une entreprise dijonnaise créée le 19 mars 2010 par Emmanuel Mignot, au capital social de 80 100 euros. Elle se dédie à la recherche, à l'étude et à la conception de solutions logicielles ; et a pour vocation de mettre au point des solutions innovantes en rapport avec le Web et la gestion de la relation client.

2.3.1/ ACTIVITÉS DE L'ENTREPRISE

eb-Lab entretient un partenariat étroit avec la société Teletech International (TTKI), toutes deux dirigées, au moment de ma thèse CIFRE, par Emmanuel Mignot. Celui-ci a souhaité créer une nouvelle société, eb-Lab, afin de diversifier ses activités sans avoir à lier TTKI à ses nouveaux projets de recherche et développement. La société TTKI étant spécialisée dans la gestion de centres d'appels, la conception et le développement de solutions orientées Web 2.0 ne correspond pas directement à son cœur de métier. Emmanuel Mignot a donc décidé de confier le développement de ces solutions innovantes à la société eb-Lab, TTKI s'occupant de les commercialiser en tant que partenaire principal.

eb-Lab compte quatre développeurs qui travaillent à plein temps, les principaux produits existants développés par eb-Lab sont :

mer (http://www.yammer.com) qui permet de travailler en réseau avec ses collègues ; ou *Bluekiwi* (http://www.bluekiwi-software.com), une plateforme de collaboration et de dialogue pour les échanges internes et externes de l'entreprise.

9. Représentation qu'un groupe se fait de lui-même, indépendamment de celle que les autres groupes se font de lui

10. représentation qu'un autre groupe se fait de ce groupe

— le lien Hypercall [11], outil marketing à destination des plateformes Web de e-commerce ;
— Social Buddies [12], outil de gestion de la relation clients, orienté s-CRM, utilisable depuis une page Facebook sous la forme d'un module permettant de gérer les demandes et questions des clients.

2.3.2/ NOUVELLES FONCTIONNALITÉS CRM

TTKI souhaite proposer de nouvelles fonctionnalités liées aux réseaux sociaux dans : 1) une plateforme Web de type s-CRM d'entreprise qu'elle a développée pour un pôle de compétitivité et 2) dans son offre logicielle existante (appelée NestCRM). Cette offre comprend différents modules, chacun étant dédié à une fonctionnalité spécifique du CRM ou de la gestion d'un centre d'appel. La figure 2.1 présente les différentes fonctionnalités présentes au sein de l'offre NestCRM.

FIGURE 2.1 – Offre NestCRM

Pour répondre à ces attentes, un groupe de travail, commun à eb-Lab et TTKI et formé juste avant le début de ma thèse, a identifié l'objectif général : comprendre le comportement des clients et des groupes de clients (communautés) sur les réseaux sociaux en

11. http://www.hypercall.org/
12. http ://www.social-buddies.fr/

relation avec une marque, un produit, un service, une entreprise. Les modèles prédictifs plus orientés opérationnel, c'est-à-dire liées aux actions, ont été dans un premier temps laissés de côté pour se concentrer sur l'analyse, l'interprétation, la compréhension et l'explication des phénomènes observés. Ainsi, en réponse à ces objectifs, le groupe de travail a identifié quelques fonctionnalités essentielles :

- extraire en continu des données du Web et surtout des principaux réseaux sociaux (Twitter et Facebook) ;
- définir une architecture d'un entrepôt de données muni d'une modélisation et d'un stockage flexibles pour ce qui est des performances et de l'évolutivité ;
- pouvoir lier les données aux connaissances du domaine ;
- disposer d'un jeu d'algorithmes adaptés pour analyser les données collectées selon trois aspects identifiés comme prioritaires : structures communautaires, utilisateurs influents [13], événements importants ;
- permettre d'adapter les analyses pour des clients spécifiques c'est-à-dire disposer d'une plateforme générique qui peut être instanciée pour un client ou pour réaliser une étude faite chez eb-Lab pour le compte d'un client.

Les objectifs de la thèse recouvrent donc deux aspects, l'un appliqué au travers de la notion de plateforme, d'entrepôt et de modèle de données flexible et permettant un passage à l'échelle des mécanisme de stockage ; l'autre plus académique concernant l'analyse de données pour étudier les communautés, les événements et les utilisateurs influents.

2.4/ PROBLÉMATIQUE ET APPROCHE

Les réseaux sociaux, nommés aussi réseaux sociaux numériques ou média sociaux, constituent un domaine de recherche très actif qui, pour sa partie analyse, est inclus dans le champ de réseaux complexes [14]. Ce domaine de recherche rassemble des chercheurs de nombreuses disciplines, physiciens, informaticiens, mathématiciens, chercheurs en sciences sociales, géographes et intègre trois grands axes de recherche :

1. l'étude phénoménologique dont l'objet est de dégager des caractéristiques générales des réseaux complexes ;
2. la modélisation qui vise à comprendre les mécanismes de formation, d'évolution, de structuration du réseau ;
3. les applications qui cherchent à comprendre le rôle des différentes caractéristiques pour étudier des phénomènes particuliers, par exemple la propagation de messages, d'épidémies, la résilience, l'importance de certains éléments du réseau, etc.

Dans le domaine de l'étude des réseaux sociaux, [Quan, 2011] présente un état de l'art technique sur les réseaux sociaux numériques, leurs fonctionnalités, les plateformes et les nouvelles problématiques de recherche comme la structure distribuée des informations, l'interopérabilité des plateformes, la recherche d'identité, la propriété des données

13. Même si les résultats des analyses ne ciblent pas l'élaboration de modèles prédictifs, le potentiel d'actions envisageables par les gestionnaire du CRM a été jugé important du point de vue des communautés et des utilisateurs influents pour lesquels il est possible d'envoyer des messages ciblés dont la couverture sera importante.

14. Un réseau complexe est un réseau constitué d'entités liées par des liens qui représentent leurs interactions et dont le comportement global n'est pas déductible des comportements individuels.

utilisateur et la sécurité. Les réseaux sociaux partagent avec de nombreux autres domaines applicatifs des problématiques de gestion de données structurées sous la forme de graphe. En effet, ils traitent d'entités en relation qui peuvent être modélisées par des graphes dans lesquels les entités (utilisateurs ou ressources) sont des sommets et les relations des arcs ou arêtes. Les relations peuvent décrire des liens d'affinité entre les utilisateurs, des similarités thématiques entre des ressources, etc. Ces graphes, appelés réseaux complexes ou graphes de terrain, ont la particularité d'être produits par des interactions et interventions humaines et s'opposent aux graphes théoriques produits à partir de modèles. Ces graphes se rencontrent également dans les sciences humaines et sociales, les réseaux informatiques ou la biologie. De plus, ils ont les particularités d'être de grande taille et d'évoluer au cours du temps, sans montrer de propriété structurelle évidente.

Les fonctionnalités identifiées dans le cadre de l'entreprise et les données nécessaires aux analyses nous placent clairement dans le contexte d'analyse de données massives. Nous abordons les réseaux complexes dans le sens des données à grandes dimensions (volumes et variétés) créées par des interactions humaines (soit entre humains directement via l'application, soit entre l'humain et l'application). Ces réseaux sont également complexes à analyser et à interpréter. En effet, les données sont souvent recueillies hors contexte, même si les opérateurs ou fonctionnalités servant à les produire sont simples et non ambigus, leur multiplicité rend leur interprétation difficile. Par exemple, sur Twitter, l'utilisation du nom d'un compte en début de *tweet* signifie généralement que l'on souhaite s'adresser ou répondre à la personne concernée ; alors que l'utilisation d'une liste de noms de comptes à la fin d'un *tweet* permet souvent d'exposer ce dernier à d'autres personnes ou à des médias.

Nos travaux se concentrent sur la détection et la caractérisation des communautés. Ces communautés peuvent aussi bien se trouver dans les différents réseaux sociaux publics que dans le réseau social professionnel d'une entreprise. De plus, les communautés d'utilisateurs existent de manière implicite et sont le résultat des comportements des utilisateurs par rapport aux applications, mais aussi le résultat des interactions entre utilisateurs. Les données recueillies à partir des réseaux sociaux ou bien à partir du système d'information des entreprises peuvent être utilisées pour constituer les profils des utilisateurs. Cependant, le profil peut contenir une très grande variété de caractéristiques. Ainsi, une sélection des composantes du profil est nécessaire afin d'obtenir un profil thématique permettant de répondre à une analyse. Les algorithmes de détection de communautés sont nombreux [Plantié et al., 2013, Fortunato, 2010, Porter et al., 2009]. Au delà de leur utilisation par des experts ayant connaissance des limites d'applicabilité et de la qualité des résultats, il est nécessaire de fournir au gestionnaire de la relation client un moyen d'interpréter et de comprendre les résultats [Yang et al., 2012]. Nous proposons d'aborder cette problématique à deux niveaux ; d'une part en modélisant des profils utilisateurs sémantiquement riches ; et d'autre part en injectant une partie de la connaissance du domaine dans les données des algorithmes afin de faciliter l'interprétation des résultats.

Du point de vue de l'analyse des données des réseaux sociaux, la détection d'événements est un élément complémentaire à l'analyse de communautés, permettant de sélectionner des intervalles de temps spécifiques à analyser. Les algorithmes de détection d'événements sont le plus souvent issus de travaux sur la détection d'anomalies dans les journaux systèmes ou applicatifs et nécessitent pour l'utilisateur une connaissance des paramètres. Nous cherchons à développer un mécanisme permettant à la fois de détecter des événements de courte durée et des événements plus longs ;

tout en fournissant un moyen à l'utilisateur de comprendre leur origine. La piste que nous explorons est similaire à celle que nous proposons pour la détection de communautés, c'est-à-dire un enrichissement sémantique des résultats de l'algorithme sous la forme d'une contextualisation par des mots-clés. Dans le cas de données de Twitter, il pourra s'agir par exemple de *hashtags*.

Nous abordons également les problématiques liées à la construction d'une plateforme pour la collecte, le stockage et l'analyse des données issues des réseaux sociaux. Les mécanismes de collecte et de stockage des données des réseaux sociaux doivent répondre aux contraintes liées à la durée de la collecte, pouvant s'étendre sur plusieurs mois, et au volume de données à collecter. Ainsi, il est nécessaire que ces mécanismes valident des propriétés de passage à l'échelle et de reprise sur panne.

Une multitude de méthodes et d'algorithmes existe pour analyser les données issues des réseaux sociaux. Ces méthodes et algorithmes ont des objectifs très différents (détection de communautés, détection d'événements, prédictions de liens, recommandation, détection de personnes influentes, analyse de sentiments, etc.). Toutefois, un gestionnaire de la relation client doit pouvoir utiliser plusieurs algorithmes pour enrichir sa connaissance sur les usages des réseaux sociaux par les consommateurs ; et ensuite élaborer des stratégies pour agir. Les algorithmes travaillant sur des représentations différentes des données (séries temporelles, matrices, graphes etc.), le modèle de stockage doit permettre d'alimenter rapidement les algorithmes en sélectionnant les données et en les mettant dans le modèle attendu par l'algorithme, sans avoir recours à de nombreux processus ETL (*Extract Transform Load*) qui peuvent s'avérer coûteux, notamment en temps. Les modèles d'entrepôt de données doivent donc être flexibles et/ou multiples. Cet aspect est renforcé par le fait que la connaissance se construit et évolue à mesure que les analyses sont réalisées, les résultats interprétés et les hypothèses confirmées ou rejetées.

Depuis 2010, différents travaux ont cherché à étendre ou adapter les techniques OLAP (*OnLine Analytical Processing*) pour les données des réseaux sociaux. Dans l'article [Costa et al., 2012], les auteurs mettent en évidence la nécessité de contextualiser les données ; et indiquent que peu d'approches s'intéressent à l'analyse des liens, et que seulement quelques unes intègrent l'analyse de sentiment. [Gallinucci et al., 2013] et [Rehman et al., 2013, Rehman, 2015] proposent un modèle intégrant un enrichissement sémantique des données capturées. L'enrichissement nécessite d'avoir une bonne connaissance du domaine et est dépendant des analyses. Dans le cas de la gestion de la relation client, il doit donc être effectué après des analyses exploratoires, ce qui rend ce type d'approche complexe à mettre en œuvre. Les approches les plus nombreuses s'orientent vers un modèle d'entrepôt pour un usage particulier, c'est-à-dire en sélectionnant certaines dimensions pour les analyses. Par exemple, dans [Bringay et al., 2011], les auteurs se concentrent sur la localisation, le temps et les mots clés. [Cuzzocrea et al., 2016] développent une modélisation dédiée à l'analyse formelle de concepts (*Formal Concept Analysis*) et une extension en logique floue. [Kraiem et al., 2015] proposent un modèle conceptuel générique pour Twitter et la traduisent en R-OLAP en tenant compte des sources, du temps, des utilisateurs et en organisant les liens selon les activités des utilisateurs et les activités concernant les tweets. Les limites sont les performances. En effet, la modélisation induit des requêtes nombreuses et coûteuses pour les parcours de graphes ou la construction de matrice d'adjacence. [Lee et al., 2014] proposent une modélisation centrée contenu, qui présente les même limites quant aux performances pour l'analyse des liens, c'est-à-dire l'applica-

tion d'algorithmes sur un réseau complexe. La thèse de Mansmann [Mansmann, 2008] présente des travaux plus généraux, orientés vers une extension des technologies OLAP pour les données complexes. Leur caractère générique et leur orientation langage et opérateurs ne permettent pas de suivre l'évolution des algorithmes proposés pour l'analyse des réseaux sociaux. Dans [Zhao et al., 2011], les auteurs proposent le modèle *Graph Cube* pour modéliser les réseaux complexes incluant des nœuds avec des attributs multiples, et définissent un ensemble d'opérateurs spécifiques aux graphes. Le modèle est implémenté en C++ et expérimenté sur des graphes de taille moyenne (jeux de données DBLP [15] et IMDB [16]. Les attributs sur les liens ne sont pas pris en compte à la conception du modèle.

Notre approche consiste donc à utiliser différents modèles de données pour construire un entrepôt qui utilisera différents systèmes de stockage, ayant chacun leurs propres paradigmes de modélisation, afin de minimiser le coût des transformations à appliquer aux données pour alimenter les algorithmes. L'entrepôt devra comporter une couche d'abstraction pour 1) collecter les données de différentes plateformes (Twitter, Facebook, etc.) et 2) aiguiller les données à stocker dans les différents systèmes (SGBDR, SGBD Graphe, stockage clé-valeur, etc.).

15. http://dblp.uni-trier.de/xml/
16. http://www.imdb.com/interfaces/

3

ÉTAT DE L'ART

Sommaire

Dans le chapitre précédent, décrivant le contexte et les évolutions de la gestion de la relation client vers des plateformes de collecte et d'analyse de données en ligne intégrant des données des réseaux sociaux, nous avons mis en évidence trois types de besoins centrés sur l'analyse des données collectées sur les utilisateurs ou les clients. Classés par ordre d'importance pour notre contexte d'application, ils sont :

— la détection et la caractérisation de communautés globales ou locales ;
— la détection des utilisateurs influents et la caractérisation thématique de cette influence ;
— la détection et la caractérisation d'événements.

La détection d'événements et la détection des utilisateurs influents ne sont pas au cœur de notre problématique. La détection d'événements doit servir de support aux analyses exploratoires. En effet, elle permet une sélection des intervalles temporels à étudier et surtout de caractériser les événements en fonction des hashtags utilisés. Pour la détection des utilisateurs influents, nous ne recherchons pas une mesure de l'influence, mais nous souhaitons utiliser les propriétés topologiques des nœuds qui peuvent refléter l'influence. En effet, la notion d'influence est complexe et plusieurs définitions ont été proposées sans réel consensus. Cependant, cette notion dépend du domaine d'étude et de la question qui motive les analyses. En effet, certains types de liens sont plus révélateurs que d'autres sur l'influence d'un utilisateur. Dans le chapitre 5, nous dresserons un rapide état de l'art de ces deux types d'analyses, nous décrirons quelques algorithmes très utilisés puis nous détaillerons notre approche.

Avant de présenter les principales approches de détection de communautés, leurs apports et les verrous restant à aborder, nous allons préciser les caractéristiques des données sur lesquelles nous travaillons, les modèles associés et les théories formelles qui les supportent. Les sections suivantes de ce chapitre décriront de manière synthétique les travaux sur la détection de communautés, qu'elle soit globale ou locale.

3.1/ NOTIONS DE RÉSEAUX COMPLEXES

Que ce soit dans le monde réel ou virtuel, les interactions entre des individus et des dispositifs logiciels produisent des données ou des traces qui peuvent être collectées, stockées, analysées et traitées afin d'en extraire de la connaissance. Les exemples de tels systèmes sont nombreux. Historiquement, les connexions entre pages Web et les journaux des serveurs HTTP contenant les pages visitées sont les premières données issues des activités d'utilisateurs réels sur Internet qui ont donné lieu à des études théoriques [Barabási et al., 1999]. Les achats sur les sites de commerce électronique ou dans les grandes surfaces génèrent également des grandes quantités de données sur les clients et les produits achetés. Les réseaux sociaux en ligne, depuis un peu plus de 10 ans, produisent des données avec de nombreux liens traduisant des relations entres individus et ou ressources. Twitter ou Facebook, par exemple, proposent des fonctionnalités de publication de messages, de re-transmission de messages avec ou sans commentaires, d'abonnements à des thématiques, de discussions, etc. Par ailleurs, les activités quotidiennes des individus, comme l'utilisation des transports en commun, les déplacements, les appels téléphoniques, ou les traces GPS, génèrent des données incluant une localisation spatiale et temporelle.

D'autres dispositifs produisent aussi des données volumineuses comme les machines

de séquençage du génome, les spectromètres, ou les capteurs embarqués dans les satellites. Cependant, les données produites par les interactions des utilisateurs et celles produites par des dispositifs automatiques ou embarqués sont très différentes en fonction des domaines considérés. À titre d'illustration, considérons trois exemples. Le projet Digital Earth [1], qui a pour but de construire une représentation virtuelle et exhaustive de la planète Terre, produit plutôt des données matricielles à grandes dimensions, issues de mesures de nombreux paramètres descriptifs. De même, pour un réseau de capteurs, chaque capteur possède plusieurs attributs (position, type de données captées, etc.), et produit des séries temporelles. Pour un réseau social, chaque utilisateur a plusieurs attributs (nom, âge, date d'inscription, etc.), tout comme chaque ressource disponible (posts créés, partagés, etc.), mais les liens entre les utilisateurs et entre les utilisateurs et les ressources ont plus d'importance que les différentes propriétés ou attributs des utilisateurs et ressources.

En restant très général, une modélisation consiste à voir ces ensembles de données comme un graphe, c'est-à-dire un ensemble de sommets ou nœuds (par exemple les utilisateurs) reliés par des liens en fonction de leurs interactions (par exemple les abonnés d'un utilisateur). Les sommets peuvent être de différents types, par exemple dans le cas d'un réseau social comme Flickr [2], qui permet à des utilisateurs de partager et d'annoter des photos, les sommets sont les utilisateurs, les photos et les annotations.

Les graphes, constitués de données réelles issues de l'évolution et de l'interaction non planifiée des "éléments du réseau", sont appelés graphes de terrain, graphes d'interactions, ou réseaux complexes [Barrat, 2013] .

Empiriquement, on constate que les réseaux complexes possèdent plusieurs propriétés [Danisch, 2015] :

- Étudiés globalement, ils sont peu denses. En effet, le nombre de liens est faible par rapport au nombre total de liens pouvant exister. Cependant, à un niveau de détail plus fin, les réseaux complexes révèlent des structures denses, avec la présence de nombreux liens formant des triangles.
- La distance moyenne entre les différents nœuds est faible. En effet, le nombre moyen de liens à suivre pour aller d'un nœud choisi aléatoirement à un autre est peu élevé.
- La distribution du nombre de liens est hétérogène. On retrouve des nœuds très connectés (*hubs*), mais aussi beaucoup de nœuds peu connectés.

L'analyse des réseaux complexes peut permettre de répondre à différents besoins comme augmenter la connaissance sur les utilisateurs, ou sur l'utilisation d'applications, de dispositifs ou de phénomènes ; améliorer l'expérience utilisateur ou encore être utilisée à des fins marketing (recommandation de produits, de contenu, d'utilisateurs).

3.1.1/ MODÈLES THÉORIQUES DES RÉSEAUX COMPLEXES

Différents modèles théoriques ont été proposés afin d'étudier plus en détail les réseaux complexes et d'appuyer leur analyse sur des bases théoriques pour tenter de répondre à des questions comme :

- Quelles sont les propriétés structurelles du réseau (globales ou locales) ?
- Quels sont les acteurs ou les liens les plus centraux ?

1. http://www.digitalearth-isde.org/
2. https://www.flickr.com/

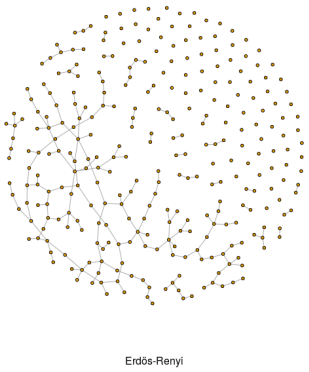

FIGURE 3.1 – Graphe aléatoire généré par le modèle Erdös Rényi

— Quelle est la vulnérabilité du réseau ?
— Comment se forme le réseau ?

Le premier est le modèle mathématique de graphe aléatoire proposé par Erdös et Rényi [Erdos et al., 1961] à la fin des années 1950. À la fin des années 1990 sont apparus deux autres modèles qui traduisent mieux certaines propriétés des réseaux complexes : les réseaux petit monde et les réseaux sans échelle.

Erdös et Rényi ont proposé leur modèle mathématique pour la construction de graphes aléatoires en s'appuyant sur un graphe $G(n, p)$ dont on fixe le nombre de sommets (n) ainsi qu'une probabilité de présence de relations entre les sommets (p). Cela permet de générer des liens de manière aléatoire entre les sommets avec les contraintes suivantes : pas de boucle ni de liens multiples, et des liens non orientés (figure 3.1 [3]). Deux propriétés ont été déterminées à partir de ce modèle. Tout d'abord, si le graphe est grand, la distribution des degrés [4] suit une loi de Poisson. Ainsi, malgré des liens aléatoires, beaucoup de sommets auront à peu près le même degré. Ensuite, une composante connexe de grande taille $(n^{2/3})$ se forme quand le degré moyen est égal à 1. Cependant, ce modèle ne permet pas de bien représenter la réalité des réseaux complexes au sens des graphes de terrain et d'interactions.

Le modèle de réseaux de type petit monde (*small-world network*) a été élaboré à la suite des travaux du sociologue Milgram [Travers et al., 1967] à partir d'une expérience qui s'est déroulée dans les années 1960 consistant à faire parvenir un courrier à un destinataire, en le donnant uniquement à des personnes connues de son entourage. L'étude a montré que la longueur moyenne de la chaîne des relations entre l'émetteur et le destinataire est de 5. La validité de cette expérience a ensuite été critiquée du fait de biais induits par le choix des échantillons de personnes participantes [Kleinfeld, 2002]. Dans [Watts et al., 1998] les auteurs proposent de partir d'un graphe dont tous les sommets ont le même degré (graphe k-régulier) et de supprimer un lien puis de le remplacer par un lien aléatoire (figure 3.2). Ce modèle permet alors de mettre en évidence deux propriétés : 1) la longueur moyenne des chemins chute très rapidement et 2) quelque soit la taille du réseau, les 5 premiers liens générés aléatoirement réduisent la longueur moyenne des

3. Les figures 1, 2 et 3 de ce chapitre ont été générées avec la bibliothèque igraph http://igraph.org du logiciel R https://cran.r-project.org/.
4. Le degré d'un sommet est le nombre de liens reliant ce sommet à d'autres sommets.

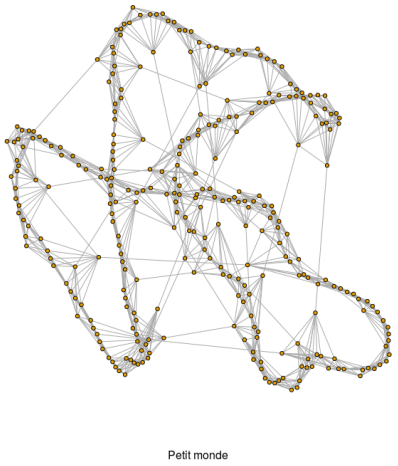

FIGURE 3.2 – Réseau petit monde

chemins par 2.

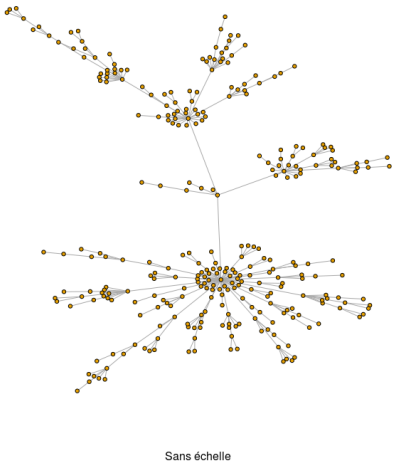

FIGURE 3.3 – Réseau sans échelle

Le modèle de réseau sans échelle (*scale-free network*) représente des réseaux dans lesquels on retrouve quelques sommets très fortement connectés, et un très grand nombre de sommets très faiblement connectés. Ils sont caractérisés par une distribution des degrés suivant une loi de puissance. Depuis les années 2000, ce modèle suscite de nombreuses recherches. Cependant, l'observation du phénomène d'absence d'échelle date des années 1960 et d'une étude de D. de Solla Price [DeSolla Price, 1965] qui portait sur l'étude des publications scientifiques et qui a montré qu'une petite partie des articles scientifiques produits étaient massivement cités et que la très grande majorité ne l'était jamais ou presque. L'explication du phénomène nommé attachement préférentiel traduit le fait que lorsqu'une entité rejoint un réseau, elle tend à se connecter aux entités les plus connectées. Les applications Web de réseaux sociaux et les pages Web constituent des

exemples de réseaux sans échelle (figure 3.3).

Dans la section suivante, nous reprenons les modèles principaux des réseaux complexes et nous donnons une définition plus formelle de leurs caractéristiques.

3.1.2/ CARACTÉRISTIQUES ET MESURES DES RÉSEAUX COMPLEXES

Un réseau est dit petit-monde lorsque la distance topologique moyenne dans le réseau, qui mesure le nombre moyen de sauts pour aller d'un nœud à un autre, varie très lentement avec le nombre total de nœuds (de manière logarithmique), même si chaque nœud est relié à un faible nombre d'autres nœuds et même si la cohésion locale reste forte [Barrat, 2013]. Ce type de réseau présente deux caractéristiques :

— la distance moyenne entre toute paire de nœuds est faible ;
— le niveau de clustering est élevé, c'est-à-dire que les nœuds sont généralement très connectés à leurs voisins immédiats.

Il est possible de calculer un coefficient de clustering, appelé aussi coefficient d'agglomération, d'agrégation ou encore de transitivité, de manière globale ou locale.

Le coefficient de clustering global C_g est défini comme :

$$C_g = \frac{3 \times \textit{nombre de triangles}}{\textit{nombre de triplets connectés}}$$

où un triangle est un sous-graphe complet [5] à trois sommets, et un triplet connecté est un sous-graphe connexe [6] à trois sommets.

Le coefficient de clustering local C_i est défini pour un sommet i, comme le rapport entre le nombre d'arêtes entre ses voisins, divisé par le nombre total d'arêtes qu'il pourrait y avoir. Ainsi, le coefficient de clustering local mesure à quel niveau le voisinage d'un point se rapproche d'une clique, c'est-à-dire d'un graphe dont deux sommets quelconques sont adjacents. Par définition, il est nécessaire que le graphe de départ soit connexe. Soit $N(i)$ le voisinage du sommet i, c'est-à-dire l'ensemble des sommets connectés directement à i, et $E(N(i))$ les liens entre les sommets de l'ensemble de sommets $N(i)$. Pour un sommet avec k_i voisins, le nombre de liens qu'il pourrait y avoir s'il s'agissait d'une clique est $k_i(k_i - 1)/2$. La valeur du coefficient de clustering local est donc :

$$C_i = 2 \times \frac{|E(N(i))|}{k_i(k_i - 1)}$$

Cette notion est étroitement liée à celle du degré de séparation popularisée par l'expérience de Stanley Milgram [Travers et al., 1967].

Un réseau est dit sans échelle (ou invariant d'échelle) lorsque la distribution des degrés, c'est-à-dire la proportion de nœuds ayant k voisins, noté $P(k)$ suit une loi de puissance, définie par :

$$P(k) \sim k^{-\gamma}$$

5. Un graphe complet est un graphe dont tous les sommets sont adjacents (reliés).
6. Un graphe est dit connexe s'il existe un chemin entre toute paire de sommets du graphe.

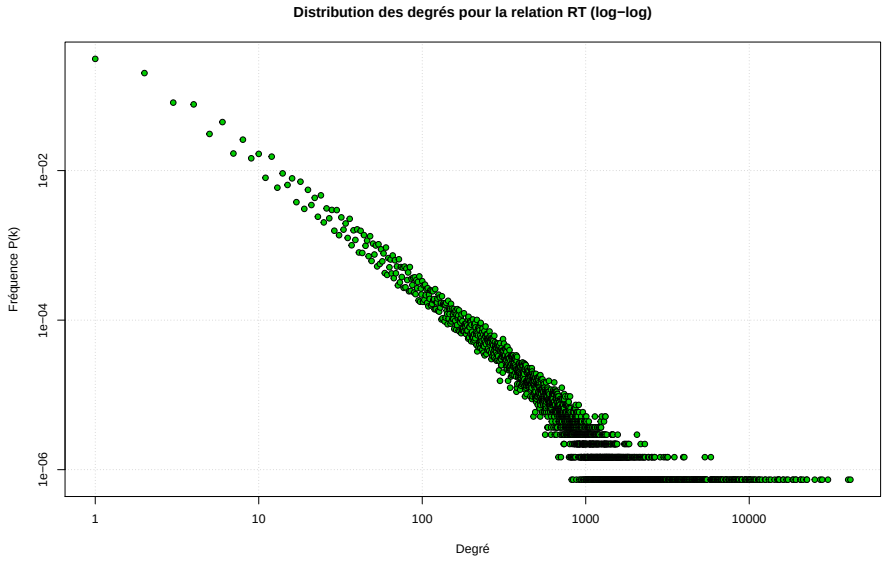

FIGURE 3.4 – Représentation de la fréquence des liens retweets en fonction du nombre de nœuds (données issues d'une étude eb-Lab sur le co-voiturage)

L'exposant γ est appelé coefficient d'invariance d'échelle, il est strictement positif et généralement compris entre 2 et 3. La figure 3.4, issue d'un graphe dont les nœuds sont des tweets et les liens des retweets[7], représente la fréquence des degrés sur un graphique logarithmique, avec le degré des nœuds en abscisse et leur fréquence en ordonnée.

L'attachement préférentiel est utilisé comme méthode générative pour les réseaux sans échelle [Albert et al., 2002] de la manière suivante :

1. l'initialisation se fait avec un nombre de nœuds limités noté m_0
2. à chaque étape on ajoute un nouveau nœud qui possède m liens avec m nœuds déjà existants, le choix des nœuds auxquels il doit se lier repose sur la probabilité dépendant du nombre de voisins du nœud i :

$$\Pi(k_i) = \frac{k_i}{\sum_j k_j}$$

3. après t itérations, on obtient un réseau comportant $N = t + m_0$ nœuds et mt liens.

Ainsi, le coefficient γ de la loi de puissance est lié au seul paramètre du modèle m. De part la méthode de construction, on retrouve dans ce type de réseaux quelques nœuds très fortement connectés, et un très grand nombre de nœuds très faiblement connectés.

Le coefficient de clustering, et l'attachement préférentiel sont des notions des réseaux petit monde et sans échelle qui révèlent des structures communautaires cachées [Girvan et al., 2002, Newman, 2006]. Nous détaillerons davantage cet aspect dans la section 3.3.

7. Un **tweet** est un message posté par un utilisateur sur Twitter ; un **retweet** a lieu lorsqu'un utilisateur reposte un tweet d'un autre utilisateur.

3.2/ SUPPORT DE LA THÉORIE DES GRAPHES

Afin d'étudier plus en profondeur les réseaux complexes et de définir des mesures et des propriétés quantifiables, il est nécessaire d'utiliser un modèle théorique plus formel que ceux qui viennent d'être définis empiriquement. La théorie des graphes constitue naturellement les fondations de l'étude des réseaux complexes. Elle peut être couplée avec d'autres théories, comme celle des probabilités.

3.2.1/ DÉFINITIONS STRUCTURELLES

Un graphe G est défini par un couple $G = (V, E)$ où V est un ensemble non vide de sommets et E un ensemble non vide d'arêtes reliant deux éléments de V. Par conséquent : $E \subseteq V \times V$. Un graphe est non-orienté si $\forall (v_1, v_2) \in E$ alors $(v_2, v_1) \in E$. Les arêtes peuvent être orientées, on parle alors d'arcs et de graphe orienté. Un poids peut être associé ou non sur les arêtes ou les arcs, on parle alors de graphe pondéré. Ce formalisme décrit les graphes simples, ou uniplexes.

Les sommets sont des entités qui sont en relation, on parle également de nœuds et on note $n = |V|$ le nombre de sommets. Une arête est incidente à un sommet si le sommet est l'une de ses extrémités. Deux sommets $v_1, v_2 \in V$ sont adjacents si ils sont aux extrémités d'une même arête, c'est-à-dire si $(v_1, v_2) \in E$. Une arête est incidente à chacun des deux sommets qu'elle relie, et deux arêtes incidentes au même sommet sont aussi dites incidentes entre elles.

En plus du modèle formel de la théorie des graphes, l'approche algébrique de la théorie des graphes s'appuie sur l'algèbre linéaire. Les matrices et leurs opérations associées permettent de représenter et de manipuler les graphes.

3.2.2/ APPROCHE ALGÉBRIQUE DES GRAPHES

Plusieurs représentations matricielles peuvent être associées à un graphe afin d'en décrire différentes relations ou propriétés.

La matrice d'adjacence A d'un graphe non orienté G est une matrice carrée booléenne de taille $|V| \times |V|$ définie par :

$$a_{ij} = \begin{cases} 1 & \textit{si } (v_i, v_j) \in E \\ 0 & \textit{sinon} \end{cases}$$

La matrice d'adjacence A est symétrique, c'est une conséquence de la définition car le graphe n'est pas orienté. La définition stricte de la matrice d'adjacence peut être généralisée aux graphes orientés et pondérés, dans ce cas A peut contenir des entiers ou des réels. Ainsi, un élément non-diagonal a_{ij} représente le nombre d'arêtes liant le sommet v_i au sommet v_j. Pour les éléments diagonaux, a_{ii}, ils représentent le nombre de boucles au sommet v_i. De plus pour le cas d'un graphe orienté, A n'est plus symétrique.

À partir de la matrice d'adjacence et de sa forme généralisée, on peut définir le degré d'un sommet. Le degré d'un sommet v_i, noté $deg(v_i)$ est le nombre de liens (arêtes ou arcs) reliant ce sommet : $deg(v_i) = \sum_{v_j \in V} a_{ij}$. Dans le cas de graphes orientés, on définit le degré sortant, ou *out-degree*, $(d^+(v))$, et le degré entrant, ou *in-degree*, $(d^-(v))$. $d^+(v)$

correspond au nombre d'arcs sortant du sommet v, et $d^-(v)$ au nombre d'arcs entrant dans le sommet v. Dans ce cas, $deg(v) = d^+(v) + d^-(v)$.

La matrice d'incidence B, pour un graphe orienté G comportant n sommets numérotés de 1 à n et p arêtes numérotées de 1 à p, est une matrice $n \times p$ définie par :

$$b_{ij} = \begin{cases} +1 & \textit{si l'arc } j \textit{ sort de } v_i \\ -1 & \textit{si l'arc } j \textit{ entre dans } v_i \\ 0 & \textit{sinon} \end{cases}$$

Pour un graphe non orienté, B est définie par :

$$b_{ij} = \begin{cases} 1 & \textit{si } v_i \textit{ est une extrémité de l'arête } j \\ 2 & \textit{si l'arête } j \textit{ est une boucle sur le sommet } v_i \\ 0 & \textit{sinon} \end{cases}$$

La matrice des degrés D d'un graphe G est une matrice carrée diagonale de taille $n \times n$ définie par :

$$d_{ij} = \begin{cases} deg(v_i) & \textit{si } i = j \\ 0 & \textit{sinon} \end{cases}$$

La matrice laplacienne L, pour un graphe G non orienté et non réflexif [8], est la différence entre la matrice des degrés D et la matrice d'adjacence A ($L = D - A$), aussi définie par :

$$l_{ij} = \begin{cases} deg(s_i) & \textit{si } i = j \\ -1 & \textit{si } i \neq j \textit{ et si } i \textit{ et } j \textit{ sont reliés par une arête} \\ 0 & \textit{sinon} \end{cases}$$

3.2.3/ OPÉRATEURS

Un avantage lié à la représentation des graphes par des matrices est l'utilisation des opérateurs algébriques sur les matrices. Ces opérateurs peuvent permettre d'extraire des données des matrices, d'effectuer des opérations de parcours sur le graphe, voire même d'exprimer de manière synthétique des algorithmes complexes au moyen de la structure de semi-anneau apportée par les opérateurs $+$ et $\times$ sur les matrices.

À titre d'exemples d'opérations d'extraction de données, en considérant la matrice $diag(v)$ comme une matrice diagonale booléenne, on peut définir les opérations suivantes :

- sélection de colonnes $A \times diag(v)$
- sélection de lignes $diag(v) \times A$
- sélection de lignes et de colonnes, c'est-à-dire d'élément à l'intersection d'une ligne et d'une colonne $diag(u) \times A \times diag(v)$

8. Un graphe non réflexif ne contient pas de boucle sur chaque sommet.

Dans le cas des graphes bipartis [9], la multiplication matricielle peut permettre d'autres interprétations. Considérons un graphe G pour lequel l'ensemble des sommets est constitué de deux ensembles disjoints $V = V_1 \cup V_2$ ayant chacun m et p éléments ($n = m + p$). Plutôt que d'utiliser la représentation par matrice d'adjacence A de taille $n \times n$, on peut utiliser une représentation plus compacte par une matrice M de taille $m \times p$ qui ne décrit que les liens entre les ensembles V_1 et V_2. Le produit $MM^\intercal$ [10] représente les co-occurrences de lignes et le produit $M^\intercal M$ les co-occurrences des colonnes. Par exemple si V_1 est un groupe de consommateurs et V_2 est un ensemble de produits, $MM^\intercal$ représente les consommateurs qui achètent les mêmes produits et $MM^\intercal$ les produits qui sont achetés ensemble.

Le produit de la matrice d'adjacence par elle-même permet d'identifier des parcours ou des chemins entre sommets. Soit G un graphe orienté ou non avec éventuellement des boucles et A sa matrice d'adjacence. A^k représente le nombre de parcours de longueur k et $a_{ij}^{(k)}$ le nombre de parcours de longueur k entre les sommets v_i et v_j. La longueur du plus court chemin entre v_i et v_j est le plus petit k tel que $a_{ij}^{(k)} \neq 0$.

La décomposition spectrale de graphe [11], issue de la théorie algébrique des graphes, étudie les rapports entre l'ensemble des valeurs propres associées aux différentes matrices (d'adjacence ou laplacienne) représentant un graphe et les propriétés générales qui peuvent en découler.

Soit M une matrice $m \times n$, dont les coefficients appartiennent au corps K ($K = \mathbb{R}$ ou $K = \mathbb{C}$). Une valeur propre λ est un élément de K tel qu'il existe un vecteur x de K^n, non nul tel que :

$$Mx = \lambda x$$

Le vecteur x est appelé vecteur propre de M associé à la valeur propre λ.

Les valeurs propres d'un graphe permettent d'en établir certaines propriétés. À titre d'exemples, nous pouvons citer les suivantes [12] :

- Soit $Diam(G)$ le diamètre [13] de G, alors A possède au moins $Diam(G) + 1$ valeurs propres distinctes.
- Si un graphe G est biparti et λ une valeur propre de A de multiplicité m, alors $-\lambda$ est aussi une valeur propre de A de multiplicité m. Le spectre du graphe est symétrique par rapport à 0.
- Soit G un graphe pondéré dont les poids des arêtes sont positifs et L la matrice laplacienne non normalisée associée à G, alors la multiplicité de la valeur propre 0 de L est le nombre de composantes connexes de G.
- La connectivité algébrique d'un graphe G est la seconde plus petite valeur propre de la matrice laplacienne de G (valeur propre de Fiedler). Cette valeur est

9. Un graphe est dit biparti si l'ensemble de ses sommets est une partition de deux sous-ensembles telle que chaque arête ait une extrémité dans l'un et l'autre des sous-ensembles.

10. La matrice transposée, notée $M^\intercal$, ou ${}^tM$, ou M^t, est le résultat de la transposition d'une matrice, obtenue en échangeant les lignes et les colonnes de M. $(M^\intercal)^\intercal = M$

11. Nommée aussi théorie spectrale des graphes.

12. [Von Luxburg, 2007, Zhang, 2013] présentent une étude plus complète.

13. La distance entre deux sommets est la longueur du plus court chemin entre ces deux sommets. Le diamètre d'un graphe est défini comme la plus grande distance entre deux sommets quelconques.

strictement supérieure à 0 si le graphe est connexe (corollaire de la propriété précédente). La multiplicité de cette valeur traduit la connectivité du graphe.

Les relations entre les valeurs propres et les propriétés du graphe sont utilisées dans les approches de marche aléatoire, qui, en s'appuyant sur la théorie des chaînes de Markov, permettent de prouver la convergence des algorithmes. De même, le vecteur de Fiedler associé à la valeur propre du même nom peut être utilisé pour partitionner un graphe.

3.2.4/ GRAPHES ET MODÈLE DE DONNÉES : DISCUSSION ET LIMITES

Considérons l'exemple de Facebook : ce réseau social permet à des utilisateurs de créer un profil, d'être *ami* avec d'autres utilisateurs, de créer des posts (messages), de partager des contenus provenant d'autres sites, etc. Une modélisation simple de ce réseau social consiste à considérer :

- l'ensemble V comme l'ensemble des utilisateurs (U), des posts (P) et des contenus partagés (C), c'est-à-dire $V = U \cup P \cup C$ et les intersections des ensembles pris deux à deux sont vides ;
- l'ensemble E comme l'ensemble des types d'interaction entre les éléments de V, à savoir les interactions utilisateur-utilisateur (un lien d'*amitié*), utilisateur-post (un post créé par un utilisateur), et utilisateur-contenu (un contenu partagé par un utilisateur), par conséquent $E = (U \times U) \cup (U \times P) \cup (U \times C)$ avec les intersections des trois composants deux à deux disjointes ;
- les liens comme orientés : un utilisateur créé un post, mais un post ne peut pas créer d'utilisateur, le lien est donc orienté et peut être associé à une fonction $post : U \rightarrow P$. Pour le cas de l'interaction utilisateur-utilisateur, l'orientation de l'arête permet de connaître quel utilisateur a réalisé la demande d'*amitié*. Les liens peuvent représenter la nature de la relation entre deux utilisateurs, mais aussi l'action d'un utilisateur.

Cet exemple, bien que simplifié, montre que le formalisme de base des graphes ne suffit pas à traduire la richesse des données des réseaux sociaux. Les multigraphes, les graphes multi-parties, les hypergraphes permettent une meilleure représentation la complexité des relations.

Afin de de traiter l'hétérogénéité des liens, il est possible d'utiliser les multigraphes. Un multigraphe est un graphe permettant d'avoir plusieurs arêtes (arêtes multiples ou arêtes parallèles) avec les mêmes extrémités [Ducruet, 2012], dans ce cas E est un multi-ensemble. Autrement dit, c'est un graphe aux relations multiples. Un multigraphe orienté est un multigraphe dont les relations sont orientées et de différentes natures. On considère généralement deux cas d'arêtes multiples avec deux sémantiques différentes :

- arêtes sans identité propre : l'identité d'une arête est définie par les deux sommets qu'elle connecte ;
- arêtes avec identité propre : les arêtes sont des entités primitives, tout comme les sommets. Lorsque plusieurs arêtes relient deux sommets, chaque arête est différente des autres.

Les multigraphes avec labels, c'est-à-dire des multigraphes étiquetés, sont ceux qui nous intéressent. Leurs liens peuvent être de différentes natures. Le faible nombre de propriétés et de théorèmes concernant les multigraphes dans la théorie des graphes conduit à contourner le problème en utilisant plusieurs graphes simples et homogènes couplés

[Ducruet, 2012].

Pour traiter l'hétérogénéité des nœuds dans le cas de deux groupes de nœuds distincts, on utilise les graphes bipartis, qu'il est possible de généraliser par la notion de graphe k-parti. Dans la théorie des graphes, ils bénéficient des travaux sur les graphes colorés avec k-couleurs, avec la contrainte que deux extrémités d'une même arête aient des couleurs différentes.

Pour traiter les relations n-aires, par exemple l'utilisation de plusieurs citations et/ou hashtags dans un même tweet, ou encore la réponse par un utilisateur à un tweet d'un autre utilisateur, il est possible d'utiliser la théorie des hypergraphes. Un hypergraphe généralise la notion de graphe non orienté dans lesquels les arêtes (appelées ici hyperarêtes) ne relient plus un ou deux sommets, mais un nombre quelconque de sommets. Les hypergraphes peuvent être orientés. Un hypergraphe H est un couple (V, ε) où V est un ensemble non vide et ε une famille de sous-ensembles non vides de V. Les éléments de V sont les sommets de H, et les éléments de ε sont les arcs ou arêtes (aussi appelées hyperarcs ou hyperarêtes) de H [Plantié et al., 2013].

Des structures comprenant des couches en plus des nœuds et des arêtes permettent de représenter des réseaux avec plusieurs niveaux ou avec des arêtes multiples de différents types. C'est ce qu'on appelle des réseaux multi-couches [Kivelä et al., 2014]. Une manière simple de les construire est d'autoriser chaque nœud à appartenir à n'importe quel sous-ensemble des couches, et de considérer les arêtes incluant des connexions par paire de toutes les combinaisons possibles de nœuds et de couches. Cette définition étant générale, elle induit plusieurs types de réseaux multi-couches.

Tout d'abord, les réseaux multi-relationnels, ou réseaux multiplexes (*multirelational networks, multiplex networks*), où les arêtes sont caractérisées par leur type. On peut les définir comme une séquence [14] de graphes $\{G_\alpha\}_{\alpha=1}^{b} = \{(V_\alpha, E_\alpha)\}_{\alpha=1}^{b}$, où $E_\alpha \subseteq V_\alpha \times V_\alpha$ est l'ensemble des arêtes et α l'index de la séquence de graphes. Généralement, les nœuds sont les mêmes entre les différentes couches, ou alors les couches partagent au moins quelques nœuds.

Les réseaux multiniveaux (*multilevel networks*), proposent une modélisation où les nœuds peuvent avoir un nombre fini de types (ou de niveaux, *levels*), et les liens peuvent exister entre les nœuds de mêmes types, ou de types "adjacents". Ces réseaux se basent sur l'analyse multiniveaux des réseaux. Par exemple, un réseau social de chercheurs (premier niveau) et un réseau d'échange de ressources entre laboratoires (deuxième niveau) auxquels les chercheurs appartiennent constituent un réseau multiniveaux à deux niveaux. Chaque niveau est vu comme une couche d'un réseau multicouches.

Face à ces différents modèles, en fonction des besoins nécessaires pour les analyses, une sélection des données pour un modèle de graphe doit être réalisée (simple, multi-relationels ou multi-couches). Ainsi, les données brutes sont transformées pour former le modèle de graphe retenu. En fonction des algorithmes à appliquer pour analyser le graphe, il est possible d'appliquer d'autres transformations afin de passer d'un modèle de graphe à un autre, par exemple la projection d'un graphe k-parti (multi-mode) en graphe simple (mono-mode), la transformation par des cliques des hyperarêtes d'un hypergraphe, etc. Cependant, les transformations ne sont pas sans impact sur l'interprétabilité

14. ou suite finie, c'est-à-dire une famille finie d'éléments indexés par $\mathbb{N}$.

des résultats des algorithmes et doivent être opérées avec une bonne connaissance du fonctionnement des algorithmes, de leurs conditions d'application et de leurs limites.

3.3/ DÉTECTION DE COMMUNAUTÉS

La classification automatique et la détection de communautés dans les graphes sont deux domaines ayant évolué de manières relativement indépendantes. Les résultats des recherches dans ces deux domaines aboutissent à des algorithmes essentiels à l'analyse des réseaux complexes et plus particulièrement des réseaux sociaux. La classification automatique rassemble des techniques dont l'objectif est une catégorisation algorithmique d'objets, c'est-à-dire l'attribution d'une classe (catégorie) à chaque objet en utilisant des caractéristiques mesurées. Elle concerne des domaines multiples comme la reconnaissance de forme dans des images, ou la détection de *spam* dans les courriers électroniques. La détection de communautés dans les graphes est beaucoup plus spécialisée. Dans cette section, nous présenterons les principes importants de ces deux domaines, et nous mettrons en évidence quelques points communs afin de préciser les mécanismes de détection de communautés dans les graphes.

3.3.1/ DÉFINITIONS DE LA NOTION DE COMMUNAUTÉ

Les définitions de la notion de communauté sont nombreuses, mais on retrouve deux orientations principales qui ont servi à forger les définitions :

- par les **sciences sociales** (plus de 100 définitions), généralement sans modèle formel permettant de quantifier les propriétés, mais riches et dépendantes du domaine. Par exemple, [Gusfield, 1978] distingue la notion de communauté du point de vue géographique (voisinage), d'une notion de communauté faisant référence à la *qualité* des relations humaines. [McMillan et al., 1986] proposent une définition du "sentiment de communauté" : le sentiment (*feeling*) que les membres ont d'appartenance à un groupe, que les membres sont importants les uns aux autres ainsi qu'au groupe, et la *foi partagée* traduisant le fait que les besoins des membres seront satisfaits à travers leur engagement à être ensemble ;
- par les **sciences physiques** : dans ce cas, il s'agit principalement de définitions utilisant une ou plusieurs mesures reliées à une fonction à optimiser.

Dans tous les cas, on retrouve des préoccupations communes : identifier des groupes d'individus ayant des comportements similaires ; identifier des groupes d'entités ayant les mêmes caractéristiques ; déterminer des zones denses dans un graphe ; etc. Dans la suite de cette section, nous nous intéresserons en premier lieu aux approches liées à des mesures. En effet, elles peuvent donner lieu à une transcription sous la forme d'un algorithme, et être combinées ou spécialisées afin de répondre aux attentes des sciences sociales, du marketing, etc.

3.3.2/ CLASSIFICATION AUTOMATIQUE ET CLUSTERING

Dans une acception générale, la classification automatique consiste à regrouper des entités qui sont semblables et à les séparer des autres, c'est-à-dire de celles qui leur sont différentes. Les entités peuvent être par exemple des individus décrits par des

caractéristiques, des produits, des documents, des images. Les critères permettant de quantifier la similarité sont nombreux, s'appuient sur des mesures quantitatives ou qualitatives et utilisent des notions de distance, de proximité ou de similarité.

La classification automatique consiste à établir un partitionnement de l'ensemble des entités $E = \{e_1, \ldots, e_n\}$ en k classes disjointes $\mathcal{P} = \{C_1, \ldots, C_k\}$ de E de telle manière à ce que les éléments d'une même classe soient proches. La partition $\mathcal{P}$ doit répondre aux contraintes suivantes :

- toutes les classes contiennent au moins un élément, $C_i \neq \emptyset, j \in \{1, \ldots, k\}$;
- tous les éléments de E sont affectés à une classe, $\cup_{i \in \{1,\ldots,k\}} C_i = E$;
- les classes forment une partition, c'est-à-dire ne se recouvrent pas, $C_i \cap C_j = \emptyset, \forall i, j \in \{1, \ldots, k\}$.

3.3.2.1/ APPROCHES POUR LA CLASSIFICATION

Suivant la connaissance préexistante, deux familles d'approches se distinguent : les classifications supervisées et les classifications non supervisées.

La classification supervisée fait l'hypothèse de la connaissance d'un certain nombre de classes et d'un échantillon E' de E qui est utilisé pour un apprentissage. Ainsi, à partir des éléments de l'échantillon, dont on connaît les caractéristiques et la classe d'appartenance, il est possible de construire un mécanisme permettant de déterminer la classe d'un élément quelconque de E à partir de ses caractéristiques.

Dans les approches de classification non supervisée, l'hypothèse de l'échantillon est absente et le nombre de classes est souvent inconnu. Cependant, certains algorithmes de classification ont besoin de paramètres comme le nombre de classes. Les algorithmes de classification se distinguent selon les résultats produits. Il peut s'agir d'affecter chaque élément à une classe ; de fournir une partition déterminée par des groupes d'éléments (les classes) ; de fournir une hiérarchie de partitions exploitables à différents niveaux de granularité. Les deux premiers types d'algorithmes sont par nature itératifs et paramétrés par le nombre de classes à produire sans avoir recours à d'autres connaissances a priori. Le troisième type apporte une hiérarchie de partitions, c'est-à-dire un ensemble ordonné $\mathcal{H} = \{\mathcal{P}_0, \ldots, \mathcal{P}_r\}$, où $\mathcal{P}_0$ est une partition de classes qui ne contiennent qu'un élément, et $\mathcal{P}_r$ est une partition contenant une classe rassemblant tous les éléments. La question est alors de savoir quel niveau de la hiérarchie est le plus pertinent.

Plus généralement, les algorithmes peuvent être ascendants et travailler par agglomération, ou descendants et procéder par division.

3.3.2.2/ ALGORITHMES POUR LA CLASSIFICATION

Le partitionnement non hiérarchique consiste à partir d'un état initial arbitraire des classes, pour ensuite aboutir par raffinement à une partition. On distingue trois grandes approches :

- Les centres mobiles (ou centroïdes) déterminent des classes. Ils sont calculés, puis chaque entité est affectée à la classe du centre le plus proche, et les centres de chacune des nouvelles classes sont recalculés [Forgy, 1965].
- L'algorithme des *k-means*, ou k-moyennes, est très proche de la méthode des

centres mobiles. Une seule entité est considérée à chaque itération, et les centres des nouvelles classes sont recalculés immédiatement après chaque nouvel ajout d'une entité [MacQueen, 1967].

— L'algorithme des nuées dynamiques est une variante de l'algorithme des *k-means*, dans laquelle les classes ne sont plus représentées par leur centre mais par un ensemble d'individus, le noyau. La distance entre une entité et le centre de la classe est remplacée par une distance moyenne par rapport aux éléments du noyau [Diday, 1971].

Ces algorithmes ont pour principal inconvénient de devoir connaître le nombre de classes souhaitées. Par ailleurs, ils sont sensibles à l'initialisation. Pour contourner ces problèmes, le principe proposé dans [Pelleg et al., 2000] consiste à ajouter progressivement de nouveaux centres (ou classes), et à mesurer si leur ajout améliore la classification. L'article de synthèse de [Jain, 2010] reprend les principales extensions des *k-means*, et montre comment les méthodes répondent à une dizaine de questions essentielles telles que la robustesse, ou la normalisation des données.

La classification hiérarchique ascendante est une méthode qui débute avec une partition constituée de classes contenant un seul élément, et qui regroupe de proche en proche les classes les plus proches en utilisant : 1) une distance définie entre les entités et 2) une mesure d'agrégation afin de pouvoir comparer des groupes d'entités entre eux.

Le choix d'une distance dépend du domaine étudié : distance euclidienne pour des vecteurs de caractéristiques numériques, ou similarité cosinus dans le cas de la fouille de textes. De même, plusieurs critères d'agrégation peuvent être utilisés, ils reposent sur la distance précédemment choisie. Le lien minimum, moyen, la méthode de Ward [Ward Jr, 1963], sont des critères couramment utilisés.

3.3.2.3/ QUALITÉ DU PARTITIONNEMENT

La qualité du partitionnement peut être évaluée soit avec des critères dépendants ou non du type de distance choisie (critères internes) ; soit avec une vérité de terrain (critères externes). Même si les méthodes de classification automatique se placent généralement dans un cadre non supervisé, l'évaluation du résultat d'un algorithme peut se faire par rapport à une partition déterminée par des experts.

Critères internes.

La notion d'inertie inter, intra-classes ou totale est un critère très utilisé, défini par analogie avec la notion d'inertie d'un corps en physique qui est la force à appliquer à ce corps en mouvement rectiligne uniforme pour le modifier. Cette notion est liée à la masse du corps. Dans le cas d'un partitionnement, on suppose que les centres [15] des classes de la partition $\mathcal{P}$ précédemment définie sont les $c_i, i = 1, \ldots, k$ et que c est le centre de E. L'inertie inter-classes caractérise la séparation entre les classes, et est définie par :

$$I_{inter} = \sum_{i=1}^{k} m_i dist(c_i - c)^2$$

15. Au sens du centre de gravité

où m_i est la masse associée à la classe (par exemple le nombre d'éléments de la classe, ou bien une fonction du nombre d'éléments). Plus l'inertie inter-classes est importante, plus les classes sont distinctes.

L'inertie intra-classe est la moyenne pondérée des inerties de chaque classe (qui mesure la variation des valeurs dans une classe), définie ainsi :

$$I_{intra} = \sum_{i=1}^{k} m_i I_{C_j}$$

où I_{C_j} est l'inertie de la classe C_j définie par :

$$I_{C_j} = \sum_{e_i \in C_j} m_{e_i} dist(e_i - c_j)^2$$

L'inertie totale est définie de la même manière par la formule suivante :

$$I_{totale} = \sum_{e_i \in E} m_{e_i} dist(e_i - c)^2$$

L'indice de Dunn [Dunn, 1973] va au delà de l'inertie, et cherche à caractériser des classes compactes et bien séparées en utilisant le rapport entre la plus petite distance inter-classes δ sur la plus grande distance intra-classe Δ. Plus la valeur de l'indice est élevée, meilleur est le résultat de la classification. En revanche, la définition de l'indice est générique, les distances ne sont pas prédéfinies. Ainsi, la distance intra-classe, notée Δ_{C_i} et nommée aussi diamètre du cluster, et peut être calculée de différentes manières (maximum des distances entre deux entités, moyenne des distances entre toutes les paires d'entités, moyenne des distances au centre, etc.).

$$I_{Dunn} = \frac{\min\limits_{1 \leq i < j \leq k} \delta(C_i, C_k)}{\max\limits_{1 \leq m \leq k} \Delta_{C_m}}$$

L'indice de Davis et Bouldin est similaire à l'indice de Dunn, mais travaille sur des paires de classes [Davies et al., 1979].

Comme les deux indices précédents, l'indice Silhouette[16] sert à caractériser la séparation entre classes, ainsi que leur compacité [Rousseeuw, 1987]. L'indice est une mesure de la similarité d'un objet par rapport à sa classe (cohésion) comparée aux autres classes (séparation). L'indice Silhouette donne des valeurs dans l'intervalle $[-1, 1]$. Une valeur proche de 1 indique qu'une entité est proche des autres entités de sa classe, et très peu des autres classes. Il est défini pour tout élément $e_i \in E$ par :

$$S(e_i) = \frac{b(e_i) - a(e_i)}{max(a(e_i), b(e_i))}$$

où $a(e_i)$ est la dissimilarité moyenne entre e_i et les autres éléments de sa classe. Elle peut se calculer comme la moyenne de la distance de e_i par rapport à tous les autres membres de la classe. $b(e_i)$ est la plus petite dissimilarité moyenne de e_i à tous les autres classes.

16. Silhouette est également une méthode graphique d'interprétation de la qualité d'une partition.

L'indice peut également se définir pour une classe comme la moyenne des indices de chaque élément de la classe, et par extension il peut être calculé sur une partition.

Dans [de Amorim et al., 2015], les auteurs expérimentent l'indice Silhouette avec les distances Euclidiene, Manhattan, et Minkowski. Les indices de Dunn, Calinski-Harabasz et Hartigan sont également testés. Les auteurs développent une méthode qui améliore l'estimation du nombre réel de clusters dans le jeux de données dans le cas des *k-means*.

Critères externes, liés à une vérité de terrain.

Parmi les approches pour l'évaluation de la qualité d'un partitionnement par rapport à une vérité de terrain, la comparaison avec les classes de référence, la comparaison d'éléments pris deux a deux dans une forme combinatoire et l'entropie sont des éléments essentiels pour établir des critères ou indices de qualité.

La comparaison des classes détectées avec celles existantes, au niveau des entités permet de définir des mesures très utilisées comme le taux de bonne classification ou l'exactitude (*accuracy*), la pureté, la précision et le rappel. Ces approches s'appuient sur un appariement qui consiste à faire correspondre les classes réelles à des classes produites par un algorithme. Prenons l'exemple de la pureté définie pour deux partitions $\mathcal{P}_{algo} = \{C'_1, C'_2, \ldots, C'_l\}$ et $\mathcal{P}_{ref} = \{C_1, C_2, \ldots, C_k\}$ désignant respectivement la partition en classes produites par l'algorithme et la partition en classes de référence (la vérité de terrain) pour n entités. Soit $n_i^j = |C'_i \cap C_j|$ le nombre d'objets communs entre les classes C'_i et C_j. La pureté est définie par :

$$Pureté(\mathcal{P}_{algo}, \mathcal{P}_{ref}) = \frac{1}{n} \sum_{i}^{l} \max_{j}(n_i^j)$$

D'un point de vue calculatoire, la pureté consiste à rechercher la classe de référence majoritaire pour chaque classe détectée par l'algorithme [Manning et al., 2008]. Les mesures utilisant un appariement soulèvent un problème important car des éléments bien classés dans des classes mal appariées ne seront pas pris en compte.

L'indice de Rand [Rand, 1971] permet de comparer des partitions en vérifiant si les couples d'entités de la même classe de référence sont présents dans la même classe détectée. Un couple d'entités est un vrai positif (VP) si les deux entités de la même classe de référence sont dans la même classe détectée, il s'agit d'un vrai négatif (VN) quand deux entités de classes de référence différentes sont placées dans deux classes détectées différentes. Un faux positif (FP) correspond à deux entités de classes de référence différentes placées dans la même classe détectée. Un faux négatif (FN) correspond à deux entités de la même classe de référence placées dans deux classes détectées différentes. L'indice de Rand peut ainsi être défini par :

$$Rand(\mathcal{P}_{algo}, \mathcal{P}_{ref}) = \frac{VP + VN}{VP + VN + FN + FP}$$

Cet indice donne un résultat dans l'intervalle $[0, 1]$ où la valeur 1 indique que les deux partitions sont identiques. Cependant, avec l'indice de Rand, les faux positifs et les faux négatifs ont le même poids. Les indices de Tanimoto [Rogers et al., 1960] et de Jaccard [Jaccard, 1912] sont construits de la même manière, la F-mesure [Van Rijsbergen, 1979] pondère les valeurs FP et FN au moyen de la précision P et du rappel R définis de la

manière suivante :

$$P = \frac{VP}{VP + FP} \quad R = \frac{VP}{VP + FN}$$
$$F_\beta = \frac{(\beta^2 + 1)P \times R}{\beta^2 P + R}$$

L'entropie de Shannon [Shannon, 1948], proche de la notion d'entropie de la physique statistique et de la thermodynamique, mesure la quantité d'information contenue dans un ensemble de messages et l'incertitude. Elle est définie, pour une variable aléatoire discrète X prenant n valeurs dans un ensemble $\{x_1, \ldots, x_n\}$, de la manière suivante :

$$H_\alpha(X) = \sum_{i=1}^{n} P(x_i) log_\alpha \left(\frac{1}{P(x_i)} \right)$$

L'entropie conjointe $H_\alpha(X, Y)$, définie pour deux variables aléatoires X et Y, est définie par :

$$H_\alpha(X, Y) = \sum_{i=1}^{n} \sum_{j=1}^{m} p(x_i, y_j) log_\alpha \left(\frac{1}{p(x_i, y_j)} \right)$$

On peut alors définir l'information mutuelle de deux variables aléatoires pour mesurer la dépendance probabiliste de ces variables.

$$IM(X, Y) = H(X) + H(Y) - H(X, Y)$$

L'information mutuelle est nulle si les variables sont indépendantes et elle augmente en fonction de la dépendance. Appliquée aux classes issues d'un partitionnement, $P(C_i) = |C_i|/|E|$ est la probabilité qu'une entité appartienne à la classe C_i. On peut calculer $H(\mathcal{P})$ et par conséquent $H(\mathcal{P}_{algo}, \mathcal{P}_{ref})$ puis $IM(\mathcal{P}_{algo}, \mathcal{P}_{ref})$. Dans [Vinh et al., 2009] et [Vinh et al., 2010] les auteurs discutent des principaux indices construits à partir de l'entropie et étudient leurs limites.

3.3.3/ ALGORITHMES POUR LA DÉTECTION DE COMMUNAUTÉS DANS LES GRAPHES

D'une manière générale, dans un graphe, une communauté est un sous-graphe dense. Une structure communautaire est un ensemble de communautés faiblement connectées entre elles. L'objectif de la détection de communautés dans les graphes est souvent de déterminer une partition du graphe, en tenant compte des relations entre les sommets, et en optimisant un critère qui traduit par exemple l'intensité des liaisons à l'intérieur d'une communauté. D'autres critères, tels que les coupes (normalisée, ration, min-max), la mesure de la modularité, ou les divisions hiérarchiques, peuvent être utilisés. Les travaux s'appuyant sur l'hypothèse d'une structure communautaire réduite à une partition du graphe sont majoritaires [Fortunato, 2010, Fortunato et al., 2016]. Cependant, d'autres approches permettent d'identifier des communautés multi-échelles, des communautés recouvrantes et des communautés locales à un nœud ou à un ensemble de nœuds, ou des micro-communautés.

Les articles présentant un état de l'art sur la détection de communautés dans les graphes sont nombreux, mais on peut distinguer deux types d'approches. Les unes organisent

les travaux selon les techniques algorithmiques ou mathématiques, et sont fort nombreuses. Les autres, minoritaires, organisent les travaux, dans une orientation *guide d'utilisation*, selon des critères généraux et non nécessairement orthogonaux. Dans cette dernière catégorie, on retrouve une évolution récente du très populaire état de l'art [Fortunato, 2010] vers une approche centrée utilisation [Fortunato et al., 2016].

La synthèse que nous proposons reprend ces deux approches : méthodes de base et critères orientés utilisateur ; et pour chaque outil algorithmique, nous montrerons en quoi il peut aider à satisfaire les besoins utilisateur. Pour cela nous nous sommes appuyés sur les travaux [Parthasarathy et al., 2011, Papadopoulos et al., 2012, Plantié et al., 2013, Kanawati, 2013] et plus particulièrement sur celui de Kanawati qui a proposé une classification des approches existantes en quatre catégories :

- les approches centrées réseau ou globales, pour lesquelles la structure entière du réseau est examinée afin de produire une décomposition du graphe en communautés ;
- les approches centrées groupes, pour lesquelles des nœuds sont regroupés en communautés en fonction de propriétés topologiques partagées ;
- les approches centrées propagations, qui simulent le déplacement d'un marcheur aléatoire (*random-walk* ou *random-surfer*) afin de déterminer, de manière probabiliste, l'appartenance de nœuds à une communauté ;
- les approches centrées graine, où la structure de la communauté est construite autour d'un ensemble de nœuds choisis pour leur caractère particulier.

Les différentes approches peuvent servir de guide en fonction du type d'analyse souhaitée. Les approches centrées réseau permettent des analyses exploratoires de communautés larges, formant le plus souvent une partition ; tandis que les approches centrées groupes permettent d'obtenir des groupes d'individus au comportement homogène. Les approches par propagation, incluant les marches aléatoires, sont polyvalentes, et rejoignent les approches centrées graine dans leur capacité à détecter des petites communautés, éventuellement recouvrantes, appelées communautés locales ou ego-centrées. Ce dernier type de communauté est essentiel pour identifier des groupes d'utilisateurs de réseaux sociaux échangeant autour de sujets particuliers, que ce soit des sujets de société, des produits ou des marques. De plus, de par la nature des méthodes mises en œuvre qui relaxent la contrainte de la partition, les approches centrées graine permettent d'identifier les éléments communs ou les frontières de communautés.

Du point de vue des outils mathématiques et des bases formelles utilisées pour le développement des algorithmes, on peut distinguer cinq grandes familles d'outils :

- les outils utilisant la topologie de graphe comme critère de constitution de communautés ;
- les optimisations d'une fonction de qualité, de manière gloutonne ou non, selon un processus agglomératif ou divisif ;
- l'algèbre linéaire et les décompositions spectrales ;
- la diffusion et les algorithmes de marches aléatoires ;
- les algorithmes reposant sur un modèle spécifique.

Ces approches peuvent s'hybrider pour produire des algorithmes complexes, ou s'enchaîner dans le cadre d'une méthode d'analyse.

3.3.4/ Outils et principaux algorithmes

3.3.4.1/ Topologie du graphe

La définition de la notion de communauté peut être mise en relation avec des propriétés topologiques caractéristiques, par exemple la notion de clique [17], ou de clique maximale. Comme la recherche d'une clique maximale est un problème NP-complet, son utilisation pour des données massives est difficile. La notion de *k-core* assouplit les contraintes de la clique : il s'agit d'un sous-graphe maximum dans lequel le degré de chaque nœud est supérieur ou égal à k. La percolation de clique suit cette logique. Le principe est de calculer des cliques de taille k, puis de construire un nouveau graphe dans lequel chaque clique est représentée par un nœud, et les liens connectent les cliques qui partagent $k-1$ sommets. Les communautés sont alors les composantes connexes du graphe de cliques.

Plutôt que d'utiliser des propriétés topologiques fortes, il est préférable de s'appuyer sur des mesures prenant en compte la connexion du graphe. Alors que les notions de clique et de *k-core* peuvent être utilisées comme amorce pour des approches centrées graine, les mesures tenant compte de la connectivité du graphe permettent de transformer le problème de détection de communauté en un problème de classification utilisant une matrice de similarité entre le nœuds du graphe. La similarité peut être basée sur le voisinage des nœuds (similarité cosinus, voisins communs, similarité de Jaccard, attachement préférentiel), ou sur les chemins (proximité, centralité de Katz, temps de commutation moyen, intermédiarité de chemin) .

3.3.4.2/ Fonctions de qualité et optimisation

Le problème de détection de communautés peut être défini comme un problème d'optimisation d'une fonction de qualité d'une partition. Parmi les principales méthodes de détection de communautés proposées dans la littérature, et s'appuyant sur une fonction de qualité pour évaluer la qualité d'une partition donnée, celles utilisant une coupe de graphe ont été largement étudiées, y compris pour la segmentation d'images. La coupe normalisée, la coupe min-max ou la coupe ratio en sont des exemples représentatifs [Shi et al., 2000, Newman et al., 2004, Kernighan et al., 1970].

Les coupes de graphe ont pour objectif d'effectuer une partition des sommets en deux sous-ensembles. La coupe normalisée d'un ensemble de sommets $S \subset V$ se calcule en additionnant :

- la somme des poids des arcs qui connectent S au reste du graphe, normalisée par la somme totale des poids des arcs de S ;
- et de la même somme normalisée par les poids du reste du graphe $\overline{S}$ [18].

La coupe normalisée est donnée par la formule suivante, définie pour la matrice d'adjacence A d'un graphe G :

17. Une clique est un sous-graphe complet.

18. $\overline{S}$ est le graphe complémentaire, ou graphe inversé, du graphe S. Il peut aussi être noté S^c. $\overline{S}$ comporte les mêmes nœuds que S, et est tel que deux nœuds distincts de $\overline{S}$ sont adjacents si et seulement s'ils ne sont pas adjacents dans S.

$$Ncut(S) = \frac{\sum_{i \in S, j \in \overline{S}} a_{ij}}{\sum_{i \in S} deg(i)} + \frac{\sum_{i \in S, j \in \overline{S}} a_{ij}}{\sum_{j \in \overline{S}} deg(j)}$$

La coupe normalisée d'une partition d'un graphe en k clusters est la somme des coupes normalisées de chaque cluster [Dhillon et al., 2007].

La modularité, proposée par [Newman et al., 2004], est une autre fonction de qualité et est utilisée dans de nombreux algorithmes. En se basant sur la notion intuitive qu'une communauté est un ensemble de sommets dont la densité des connexions internes est plus importante que la densité des connexions externes, elle consiste à mesurer la différence entre la proportion de liens internes aux communautés et la même mesure dans un modèle sans structure communautaire, c'est-à-dire un graphe aléatoire ayant le même nombre de nœuds, de liens et la même distribution des degrés. Pour un graphe $G = (V, E)$ et une partition en communautés disjointes $\mathcal{P} = \{C_i, i = 1, .., n\}$, la fraction de liens situés à l'intérieur des communautés de $\mathcal{P}$ est notée $\sum_{i=1}^{n} e_{C_i}$. Dans le graphe aléatoire, pour $\mathcal{P}$, la probabilité qu'un lien ait une extrémité dans la communauté C_i est $a_{C_i} = (\sum_{j \in C_i} deg(j))/2|E|$, avec $deg(j)$ le degré du sommet j. La probabilité que les deux extrémités du lien soient dans la communauté C_i est $a_{C_i}^2$ et par conséquent la modularité est :

$$Q(\mathcal{P}) = \sum_{i=1}^{n} (e_{C_i} - a_{C_i}^2)$$

Cependant, maximiser la modularité d'un graphe est un problème NP-difficile. Deux approches opposées servent de guide au développement des algorithmes :

- les approches agglomératives ou *bottom-up* démarrent avec une partition en classes contenant un seul élément et fusionnent les communautés à chaque itération. Les communautés candidate à la fusion sont celles qui vont maximiser la modularité. Ces approches sont reprises dans [Newman et al., 2004, Pons et al., 2005] et dans la méthode de Louvain [Blondel et al., 2008], qui est une heuristique, composée de deux phases qui sont répétées jusqu'à obtenir un maximum local de modularité.
- les approches divisives ou *top-down* démarrent avec une classe contenant tous les nœuds, et la divisent en deux communautés à chaque itération de manière à maximiser la modularité. Ces approches sont présentées dans [Newman, 2006] et [Costa et al., 2016].

La notion de modularité introduite par Newman a été étendue pour traiter des graphes avec des liens valués par [Blondel et al., 2008].

Même si la modularité est très populaire, les algorithmes présentent souvent un problème de limite de résolution. Pour des graphes non pondérés, la maximisation de la modularité ne permet pas d'identifier des communautés ayant un nombre de liens inférieur à $\sqrt{\frac{|E|}{2}}$ [Fortunato et al., 2007]. De plus, les algorithmes sont très sensibles à des perturbations [Aynaud et al., 2010] et donc à l'initialisation si elle comporte une partie aléatoire. De ce fait, plusieurs exécutions du même algorithme peuvent produire des structures communautaires différentes. Cet effet est étudié formellement dans [Good et al., 2010], qui démontrent qu'il existe des partitions différentes ayant une valeur de la modularité maximale.

L'optimisation est généralement effectuée en utilisant une méthode gloutonne ou une méthode stochastique. En effet, de part leur rapidité d'exécution, elles sont assez bien adaptées au traitement de grands graphes. L'état de l'art met en évidence : 1) qu'il n'y a pas de fonction de qualité faisant l'unanimité, mais plutôt un choix à faire en fonction des données et 2) que les méthodes d'optimisation peuvent conduire à des extrema locaux peu pertinents. Ces limites ne remettent pas en cause les algorithmes utilisant la modularité, mais démontrent la nécessité d'utiliser différents critères de mesure de qualité, ou d'utiliser plusieurs algorithmes, par exemple dans une approche par consensus.

3.3.4.3/ OUTILS DE L'ALGÈBRE LINÉAIRE

Les algorithmes spectraux [Von Luxburg, 2007] sont des méthodes classiques pour le regroupement de nœuds et la découverte de communautés dans les réseaux. Les méthodes spectrales se réfèrent généralement à des algorithmes affectant les nœuds aux communautés en fonction des vecteurs propres des matrices caractéristiques du graphe, comme la matrice d'adjacence. L'idée principale du regroupement spectral est d'obtenir une représentation à plus faible dimension, induite par les vecteurs propres associés aux plus grandes valeurs propres, de manière à révéler la structure communautaire du graphe. Les k principaux vecteurs propres définissent une représentation des nœuds dans un espace de dimension k. On peut ensuite utiliser des techniques classiques de regroupement de données, telles que *k-means*, pour déterminer l'affectation finale des nœuds aux clusters.

Les approches de type *block-modeling* sont mixtes. Elles utilisent une représentation matricielle pour regrouper les éléments d'un graphe ayant les mêmes caractéristiques topologiques. Le principe algorithmique est simple : une première étape permute les lignes et les colonnes de la matrice d'adjacence, afin de positionner les uns à coté des autres les sommets ayant le même type de relations structurelles. La matrice obtenue est une *blocked matrix* ; la seconde étape consiste à agréger les sommets appartenant au même bloc pour créer une *density matrix*. L'équivalence structurelle de deux sommets peut être mesurée de différentes manières, par exemple avec une distance ou une corrélation [Snyder et al., 1979], ou encore des équivalences selon le type des acteurs représentés par les sommets [Borgatti et al., 1992].

Par extension, les factorisations matricielles, et les méthodes de décomposition associées, permettent d'effectuer une réduction d'intentionnalité, c'est-à-dire d'exprimer dans un espace plus petit les données d'origine. Les factorisations LU (pour *lower upper*, QR, la décomposition en valeurs singulières (SVD), ou CUR sont des méthodes ayant fait l'objet de nombreuses recherches, tant d'un point de vue théorique qu'appliqué [Abdi, 2007, Hogben, 2013, Leskovec et al., 2014]. Une fois la réduction opérée, un des algorithmes de détection de communautés peut être appliqué, mais l'interprétation des résultats peut être complexe du fait de l'agrégation de différentes caractéristiques sur les dimensions de l'espace réduit.

3.3.4.4/ MARCHES ALÉATOIRES

Les marches aléatoires [Motwani et al., 2010] exploitent la propriété de la densité des liens intra-communautaires pour faire l'hypothèse qu'un marcheur ou une diffusion

aléatoire dans le graphe aura plus de chance de rester dans la communauté du nœud initial que de sortir ou se propager aux autres communautés.

L'algorithme WalkTrap [Pons et al., 2005] suppose qu'on effectue une marche aléatoire d'une courte longueur en partant d'un sommet i de G. La probabilité d'accéder à des voisins de i en une étape est $1/|N(i)|$. Il est donc possible de calculer la probabilité de se trouver au sommet j en partant de i et en ayant parcouru k liens. Il est alors possible de définir une distance entre les paires de sommets de G. Deux sommets i et j sont proches si leurs vecteurs de probabilité d'atteindre les autres sommets sont similaires. Une fois les vecteurs de probabilité initiaux calculés pour tous les sommets de G, l'algorithme utilise un clustering hiérarchique pour partitionner le graphe au travers des vecteurs de probabilité. À partir d'une partition en n communautés, chacune contenant un seul sommet, l'algorithme recherche les deux communautés les plus proches, les fusionne, puis recalcule les vecteurs de probabilité, et recommence l'itération jusqu'à obtenir une seule communauté contenant tous les sommets du graphe. Pour chaque configuration de partition, l'algorithme évalue la modularité et conserve la configuration qui maximise la modularité.

L'algorithme PageRank enraciné [Page et al., 1999] propose d'utiliser un marcheur aléatoire démarrant depuis un nœud d'intérêt et se déplaçant aléatoirement à chaque étape vers un voisin ou en se téléportant au nœud de départ avec une probabilité α. À l'étape $t+1$, la probabilité de trouver le marcheur à un nœud donné est fournie par le vecteur PageRank ou vecteur des scores :

$$X_{t+1} = \alpha T X_t + (1-\alpha) X_0$$

où X_t est le vecteur des scores après t itérations, x_{t_i} la probabilité d'avoir un marcheur au nœud i après t itérations, X_0 le vecteur d'initialisation contenant 1 pour le nœud d'intérêt et zéro pour les autres composantes, et T la matrice de transition. T est définie par :

$$T_{ij} = \frac{w_{ij}}{deg(j)}$$

où w_{ij} est le poids du lien entre les nœuds i et j, et $deg(j)$ le degré du nœud j. En plus de spécifier la probabilité d'un déplacement ou d'un saut, le paramètre α conditionne la profondeur d'exploration.

Du point de vue de la détection de communautés, la valeur associée à chaque sommet, donnée par x_{t_i}, peut être interprétée comme une proximité par rapport au nœud d'intérêt, ou encore comme une probabilité d'appartenance à la communauté du nœud d'intérêt.

Le vecteur X_{t+1} est la solution de l'équation PageRank, le vecteur des proximités peut être obtenu soit de manière itérative, soit en calculant le premier vecteur propre du système. La convergence du processus est prouvée à l'aide de la théorie des chaînes de Markov.

La structure communautaire extraite par le PageRank enraciné peut être mise en évidence en étudiant les variations de la proximité en fonction du classement des nœuds [Danisch, 2015]. Ainsi, des plateaux révèlent les communautés et des structures de plateaux multiples peuvent s'interpréter comme des communautés multiniveaux.

Il est possible d'utiliser une initialisation du vecteur X_0 contenant plusieurs nœuds d'intérêt, avec ou sans lien sémantique réel, afin d'étudier par exemple la formation de

communautés autour de hashtags, ou encore des frontières de communautés entre des groupes d'utilisateurs distincts.

Ces techniques permettent de définir des algorithmes centrés graine, partant d'un ensemble de nœuds obtenus soit par des experts du domaine soit par un moyen automatique (cliques par exemple), puis en appliquant un processus d'expansion autour des graines afin d'identifier une ou plusieurs communautés.

3.3.4.5/ APPROCHES GUIDÉES PAR UN MODÈLE

La propagation de label [Raghavan et al., 2007] est un principe itératif. À chaque itération, un nœud envoie son label à ses voisins et reçoit les leurs, chaque nœud calcule le label majoritaire qui devient alors son label pour les itérations suivantes. À l'issue du processus, les nœuds auront un label stable qui déterminera les groupes. Le processus peut être exécuté de manière synchrone ou asynchrone, cette dernière favorisant le parallélisme. Cependant la convergence du processus n'est pas toujours assurée dans le cas asynchrone. De plus, cette méthode présente deux inconvénients importants. Le premier est que certaines propagations peuvent produire de nombreuses communautés, le second est l'instabilité de la méthode qui ne produit que rarement le même résultat. Ce dernier inconvénient est en partie dû à la méthode d'initialisation qui établit le label de certains nœuds alors que les autres sont laissés sans label. Pour un meilleur comportement vis à vis du premier inconvénient, l'indice de Jaccard peut être utilisé pour fusionner des communautés [Raghavan et al., 2007]. Dans [Malek et al., 2016], les auteurs proposent des solutions pour contourner ces inconvénients et un framework Hadoop/Spark pour implanter leur algorithme.

L'algorithme de colonie d'abeilles artificielles (*artificial bee colony*) est un modèle bio-inspiré utilisé comme heuristique pour des problèmes d'optimisation [Karaboga, 2005, Karaboga et al., 2007]. Il trouve naturellement sa place dans les approches de clustering [Zhang et al., 2010]. Le modèle comporte trois composantes essentielles : 1) les sources alimentaires, 2) les butineuses actives et 3) les butineuses non actives ; ainsi que deux modes principaux du comportement : 1) le recrutement de butineuses pour exploiter une source de nectar et 2) l'abandon d'une source. La valeur de la source alimentaire dépend de nombreux facteurs, tels que la proximité de la ruche, la richesse ou la concentration de nectar (énergie) et la facilité d'extraction. Chaque butineuse active est associée à une source et peut transmettre ou partager les informations sur cette source avec d'autres butineuses selon une probabilité prédéfinie. Parmi les butineuses non actives, on distingue deux catégories. Les éclaireurs (*scouts*) recherchent l'environnement entourant la ruche pour de nouvelles sources de nourriture, et les spectatrices (*onlookers*) attendent dans la ruche et sont informées de la présence d'une source de nourriture par les butineuses actives. Une source de nourriture représente une solution possible au problème à optimiser. La quantité de nectar d'une source correspond à la qualité de la solution représentée par cette source. La colonie d'abeilles cherche des sources de nourriture en maximisant le ratio E/T où E est l'énergie obtenue et T est le temps consacré à la recherche de nourriture. Dans un problème de maximisation, l'objectif est de trouver le maximum de la fonction objectif $F(\theta), \theta \in \mathbb{R}^p$ avec θ_i représentant la position de la $i^{\text{ème}}$ source et $F(\theta_i)$ représentant la quantité de nectar θ_i et est proportionnelle à l'énergie $E(\theta_i)$.

L'algorithme LICOD [Kanawati, 2011] propose un modèle de formation de communautés autour de *leaders*. Les leaders sont identifiés à chaque étape de l'algorithme, puis un

processus d'agrégation utilisant une distance permet d'effectuer les regroupements.

Les algorithmes s'appuyant sur des modèles spécifiques exploitent la sémantique introduite dans le modèle à des fins différentes, par exemple simuler une diffusion d'information structurant les communautés, définir une heuristique (méta) pour une optimisation d'une fonction qualité, etc. Ils permettent donc, en intégrant une sémantique, d'être plus proche des besoins des utilisateurs. Toutefois, comme en témoignent les inconvénients identifiés pour les algorithmes de propagation de label, la convergence ou l'unicité des résultats peuvent être difficiles à atteindre.

3.3.5/ DISCUSSION

Majoritairement, les algorithmes considèrent les communautés comme des partitions. Ceci implique qu'un nœud appartient à une seule communauté, ce qui est contraire au sens commun et à la plupart des communautés que l'on rencontre dans le monde réel. En effet, un découpage global d'un réseau ne permet pas de faire ressortir les relations qui ont lieu à un niveau local, ni de faire ressortir les relations entre les communautés recouvrantes [Ahn et al., 2010]. Néanmoins, comme nous l'avons détaillé dans la section 3.3.4.4, il existe des travaux avancés portant sur les communautés locales, et les communautés ego-centrées. Ces travaux s'intéressent aux communautés formées à partir d'un nœud donné ou d'un groupe de nœuds. Les questions connexes abordées sont : 1) trouver toutes les communautés du nœud d'intérêt ; 2) trouver la ou les relations entre le nœud en question et une ou toutes ses communautés locales (existent-elles d'un point de vue global, ou uniquement local ?) et 3) le nœud est-il indispensable à la ou aux communautés trouvées ? ou bien quelle est la position ou l'importance du nœud dans les communautés [Danisch, 2015] ? Dans le cas des communautés recouvrantes, la notion de frontière de communauté est peu étudiée. En d'autres termes, les travaux de recherche sur les communautés locales ne s'intéressent donc pas à toutes les communautés du graphe, mais seulement à certaines, identifiées par des nœuds d'intérêt. Se pose alors la question d'amorcer le processus. Nous pensons que seule une approche combinant plusieurs algorithmes dans une démarche itérative et incrémentale permet une analyse fine des données des réseaux sociaux.

Dans le même ordre d'idée, les communautés peuvent elles-mêmes être découpées en sous-communautés, et ainsi de suite. Ces différents découpages récursifs permettent d'obtenir la structure hiérarchique complète, mettant en évidence la structure multi-échelle des réseaux complexes [Aynaud et al., 2010]. On parle de déplier la structure, ou *unfolding, peeling*.

Les algorithmes présentés utilisent plusieurs représentations des données matricielles : matricielle orientées graphe (matrice d'adjacence ou Laplacienne) ; espace multi-dimensionnel. À partir des données brutes collectées, il est nécessaire de modéliser les données en fonction de la question à laquelle on cherche à répondre, et ensuite d'opérer des transformations de modèle ou des extractions afin d'alimenter le ou les algorithmes d'analyse. Ces opérations sont coûteuses en temps de calcul et en temps de développement. De plus, les algorithmes utilisant une optimisation gloutone et/ou heuristique ne sont pas nécessairement stables, plusieurs exécutions pouvant donner des résultats différents. Des auteurs ont proposé la notion de cœur de communauté pour détecter les parties stables [Seifi, 2012].

Parmi les approches décrites, nous pouvons remarquer que les algorithmes ne proposent

pas de caractérisation des communautés, mais uniquement des groupes homogènes. Par conséquent, le résultat est plus difficile à interpréter et à valider. Même si des critères de qualité sont disponibles, ils mesurent la qualité intrinsèque de l'algorithme par rapport à des critères dits objectifs ou par rapport à une vérité de terrain, et non pas l'adéquation entre les résultats et la question. Des informations complémentaires sont par conséquent nécessaires pour caractériser ces communautés.

Du point de vue de la richesse des données, la plupart des approches ne tiennent compte que d'un seul type de lien dans le graphe, c'est-à-dire un graphe homogène et non pas un graphe multi-relationnel. Cependant, depuis quelques années, les algorithmes principaux ont été adaptés aux réseaux multi-couches et multi-relationnels [Kivelä et al., 2014]. Ces travaux renforcent encore plus les aspects modélisation des données, du point de vue de la sémantique portée par le modèle et de son adéquation, avec les questions de recherche ainsi que l'interprétabilité des résultats des algorithmes.

Même si, du point de vue de la modélisation et des algorithmes, il reste encore beaucoup de questions pour pouvoir traiter les relations multiples existantes dans les réseaux réels, de nombreux outils ont été développés, [Fortunato et al., 2016] en présentent une synthèse. Face à la complexité des hypothèses de départ et des techniques mises en œuvre, les outils logiciels de type *boîte noire*, par exemple Visibrain [19], ElasticSearch [20], etc., doivent être utilisés avec précaution pour s'assurer de l'interprétabilité des résultats. En effet, la plupart de ces outils offrent des solutions clés en main, sans nécessairement donner plus de précision sur les algorithmes, modélisations et hypothèses utilisés. Ces logiciels vont produire un résultat à partir d'un jeu de données ; pour autant l'interprétabilité de ce résultat peut se réveler hasardeuse.

3.4/ CONCLUSION

L'étude des principales approches et algorithmes pour la détection de communauté font apparaître deux orientations différentes. Tout d'abord, on observe une première orientation, majoritaire, des travaux vers la détection de communautés globales. Ces dernières sont souvent considérées comme des partitions d'un graphe. Cependant, depuis quelques années, une deuxième orientation se concentre sur des communautés locales, formées à partir d'un ensemble de nœuds (par exemple des comptes d'utilisateurs ou des hashtags) qui permettent d'identifier des communautés thématiques. On constate aussi que les travaux sur la prise en compte de la sémantique du domaine sont rares, que ce soit au niveau des algorithmes (de manière à caractériser ou à assurer l'interprétabilité des résultats), ou que ce soit au niveau des modèles. Concernant ce dernier point, des travaux récents sur les notions de réseaux multi-couches ou multi-relationnels ont permis d'adapter les principaux algorithmes. Toutefois, l'aspect modélisation des données, au sens des systèmes d'information, est très peu discutée.

D'une manière plus générale, l'état de l'art a montré que l'outillage algorithmique et théorique pour l'analyse des données issues des réseaux sociaux numériques existe, mais que les aspects système d'information ne sont pas encore réellement pris en compte du point de vue des modèles de données, de la sémantique des données et des résultats des algorithmes. L'intégration de ces outils avec des modèles et des connais-

19. http ://www.visibrain.com/fr/
20. https ://www.elastic.co/fr/

sances dans une plateforme permettra aux utilisateurs de tirer pleinement parti de la capacité des algorithmes, garantie par un support théorique fort.

4

MODÉLISATION, DÉTECTION ET CARACTÉRISATION SÉMANTIQUE DES COMMUNAUTÉS

Sommaire

L'étude des besoins de la gestion de la relation client étendue aux réseaux sociaux nous a permis d'identifier trois grandes catégories d'outils d'analyse qui font actuellement défaut aux solutions existantes : la détection et la caractérisation de communautés, d'événements et d'utilisateurs influents.

Ces trois catégories peuvent se décliner de manière plus précise au niveau des outils nécessaires à l'analyse de données :

- détection de communautés s'appuyant sur un profil (sémantiquement riche) des utilisateurs pour des plateformes collaboratives ;
- détection et caractérisation sémantique de communautés pour les plateformes collaboratives et les réseaux sociaux en s'appuyant sur les liens entre utilisateurs et mots-clés (annotations, hashtags) ;
- détection d'événements dans les réseaux sociaux et leur caractérisation sémantique ;
- détection de personnes influentes ou faisant autorité en utilisant la richesse des liens entre les utilisateurs des réseaux sociaux.

La sémantique, abordée dans notre cas comme la contextualisation des données ou des résultats des algorithmes par rapport à la connaissance du domaine, ainsi que la modélisation des données issues des réseaux sociaux par des graphes ou des profils utilisateurs, constituent les deux fils conducteurs de nos recherches.

Pour chacun des quatre types d'outils identifiés nous développerons nos contributions dans les chapitres suivants et nous les replacerons dans le contexte de l'état de l'art présenté au chapitre précédent. Dans ce chapitre, nous présentons en détail nos travaux portant sur les communautés. Après une synthèse des modèles de profil utilisateur et de leur utilisations, nous décrivons notre proposition de modèle de profil thématique pour des utilisateurs d'une plateforme CRM (section 2). Nous montrons comment un profil thématique entre dans un mécanisme de détection de communauté. Ensuite, nous prenons en compte des données non plus issues uniquement d'une plateforme CRM, mais aussi celles issues des réseaux sociaux ; et nous enrichissons des algorithmes de détection de communautés globales et locales pour permettre leur caractérisation, c'est-à-dire, avec notre approche, leur description en fonction des hashtags.

4.1/ Modélisation des profils utilisateurs : annotations et liens sociaux

Les plateformes de CRM ou les plateformes collaboratives utilisées dans un domaine particulier permettent une description fine des utilisateurs de la plateforme sous la forme d'un profil détaillé. Ce dernier est déclaré par l'utilisateur et/ou établi à partir de ses activités sur la plateforme. Ainsi, le profil d'un utilisateur de la plateforme est constitué d'un ensemble d'informations le concernant comme son nom, son âge, sa ville, complété par des informations sur ses centres d'intérêts et les évaluations qu'il donne à des ressources [Golbeck, 2009]. Les centres d'intérêts peuvent être renseignés directement par l'utilisateur, de manière explicite ; ou bien calculés, de manière implicite, en analysant son comportement.

Dans les approches avec profil explicite, [Hung et al., 2008] définissent un profil comme un ensemble de tags (annotations utilisant des mots-clés) et de poids. [Cattuto et al., 2008] proposent un algorithme de recommandation basé sur les tags.

Dans ce travail, l'utilisation des tags est analysée sur le site de musique `last.fm`[1], où les pistes musicales sont filtrées en fonction des classements personnels (votes) de l'utilisateur. Cette méthode se heurte au problème de l'initialisation du profil des nouveaux utilisateurs (*cold start*) qui reçoivent d'abord que des recommandations peu pertinentes. [Firan et al., 2007] utilisent les tags pour construire des profils pour `last.fm` en utilisant la musique déjà présente sur l'ordinateur de l'utilisateur pour contourner le problème du *cold start*. Cette amélioration peut s'apparenter à l'utilisation d'une connaissance locale pour améliorer le comportement de l'algorithme. Les auteurs ont montré que l'utilisation des tags dans les profils aboutissait à de meilleures recommandations par rapport à une utilisation de profils basés uniquement sur les activités, c'est-à-dire ici sur les chansons écoutées par les utilisateurs, mais remarquent un besoin de désambiguïsation des tags.

Les approches intégrant une composante implicite dans le profil utilisateur proposent de l'enrichir avec les usages et les comportements de l'utilisateur. En effet, le profil d'un utilisateur est également défini par son environnement ou contexte. [Dey et al., 2001] décrivent le contexte comme l'ensemble des informations qui peuvent être utilisées pour caractériser la situation d'une entité, par exemple le réseau d'amis et les ressources annotées par l'utilisateur.

Les données issues des médias sociaux peuvent alimenter le contenu d'un profil utilisateur aussi bien dans sa composante explicite qu'implicite. [Abel et al., 2011a] proposent un framework de modélisation du profil utilisateur en se basant sur ses activités sur différents médias sociaux tels que Flickr, Twitter et Delicious[2]. Ils remarquent que cette méthode permet d'améliorer la qualité des recommandations dans un système n'ayant que peu d'information sur ses utilisateurs. Dans [Abel et al., 2011b], les mêmes auteurs utilisent des tweets pour modéliser les intérêts d'un utilisateur. Ils analysent le contenu posté par l'utilisateur sur Twitter : les tweets et hashtags[3], ainsi que les liens inclus dans les tweets pour caractériser leur contexte.

Afin de pouvoir exploiter les traces des interactions des utilisateurs avec les applications Web, le format standardisé, Activity Streams[4], associe des métadonnées aux actions réalisées par un utilisateur afin de les différencier les unes des autres et leur donner plus de sens. Il est basé sur le schéma **acteur, verbe, objet cible**. Par exemple, un utilisateur publie une ressource, un utilisateur associe un tag à une ressource, ou un utilisateur est en contact avec un autre utilisateur.

Plusieurs travaux ont étudié les posts et messages issus de médias sociaux, dans le cadre du Social CRM. [Ajmera et al., 2013] définissent un système prenant en compte différents paramètres comme l'intention d'un post ou la nature de son auteur pour identifier les posts pertinents pour une entreprise. [Wu et al., 2009] proposent un nouveau framework pour le CRM transformant les méthodes traditionnelles des CRM, qui se basent sur les individus, en méthodes se basant sur des groupes d'utilisateurs émergeant de l'analyse des réseaux sociaux.

La modélisation des utilisateurs avec un profil unique pour chacun est une étape importante dans la détection des communautés. Les méthodes de détection de communautés utilisent principalement des graphes simples homogènes, comme des liens entre utilisateurs ou entre documents, pour construire les communautés. La sémantique véhiculée

1. https://www.last.fm/
2. https://delicious.com/
3. Mot clé, symbolisé par un # sur Twitter et Facebook, utilisé pour catégoriser un message
4. http://activitystrea.ms/

par le modèle de graphe strict, constitué de liens et de nœuds homogènes, est simple mais elle ne reflète pas la complexité de la réalité. Le comportement d'un utilisateur, ses actions et centres d'intérêts sont rarement pris en compte dans la détection des communautés.

La construction des profils avec les tags est généralement basée sur des mots-clés choisis par les utilisateurs et donc non contrôlés par le système, ce qui pose des problèmes d'interprétation, induits par leur homogénéité et leur variabilité sémantique. Dans le cadre d'une plateforme CRM les biais engendrés par les tags peuvent être contournés en utilisant une ontologie de domaine ou une ontologie applicative [Guarino, 1997].

L'approche que nous développons dans ce chapitre propose une solution combinant modèles et algorithmes pour répondre à la problématique de la détection de communautés dans le cadre de la gestion de la relation client. Du point de vue modèle, dans le cas d'une plateforme CRM, nous définissons un modèle générique de profils thématiques ; et dans le cas des données des réseaux sociaux, nous proposons une prise en compte des différents types de liens et de nœuds.

4.2/ MODÈLE GÉNÉRIQUE DE PROFIL THÉMATIQUE

Nous présentons dans cette section notre solution de modèle générique de profil thématique utilisateur dans le cadre de la relation client en étendant la notion de réseau social d'entreprise aux clients, en prenant en compte les intérêts et les usages des utilisateurs ainsi que leur propre réseau de contacts. Avant de définir le modèle, nous définissons une architecture type de plateforme Social CRM, DisCoCRM. Les utilisateurs de DisCoCRM sont les *community manager*, les clients interagissant avec le système d'information de l'entreprise. Ce dernier, ainsi que les réseaux sociaux grand public, constituent les sources des données qui vont servir à alimenter les profils.

4.2.1/ L'ARCHITECTURE DISCOCRM

L'architecture générale d'un Social CRM prend en compte les interconnexions entre une entreprise, ses ressources et les utilisateurs. Celle que nous décrivons est issue des travaux de l'entreprise Teletech International, partenaire de la thèse. Cette entreprise, de par son expérience dans le développement de plateformes CRM classiques, a abouti a la spécification d'une extension de ces dernières pour le Social CRM. Du point de vue des exigences, il est nécessaire de modéliser les ressources et les interactions des utilisateurs, entre eux et avec les ressources. Le profil utilisateur reprendra les différents types de liens dans une vision centrée utilisateur. Ce profil sera ensuite exploité par un mécanisme de détection de communautés.

Les figures 4.1 et 4.2 présentent l'architecture générale de DisCoCRM, composée de trois parties fonctionnelles distinctes :

1. *Le site Web* dédié à l'entreprise (figure 4.1) contient un ensemble de ressources, catégorisées via une base de tags. Chaque ressource est associée à un ou plusieurs tags qui représentent des catégories. L'historique des actions d'un utilisateur sur le site Web permet d'étudier son comportement. Il peut consulter, partager,

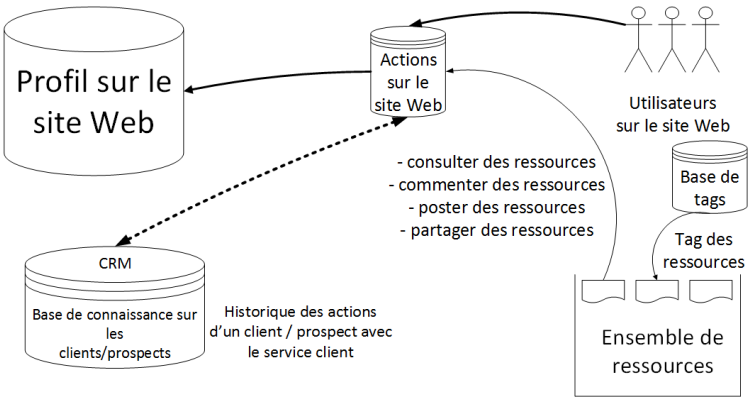

FIGURE 4.1 – Construction du profil utilisateur sur le site Web d'une entreprise

commenter des ressources existantes, mais aussi en poster de nouvelles. Les actions des utilisateurs sur le site Web permettent de construire le profil utilisateur sur le site Web.

2. *Les média sociaux grand public*, tels que Facebook et Twitter (figure 4.2), sont utilisés pour affiner le profil utilisateur avec des informations qui ne sont pas disponibles à l'intérieur du SI de l'entreprise. Sur ces réseaux, on s'intéresse aux interactions utilisateur suivantes : un utilisateur peut poster, partager ou mettre en favori des ressources, il peut également créer des ressources annotées (par exemple avec un hashtag sur Twitter) ou poster des liens vers d'autres sites Web, et posséder une liste de contacts au sein des réseaux sociaux. On suppose qu'il y a une intersection entre l'ensemble des tags utilisés dans les média sociaux et l'ensemble des tags utilisés dans le site Web de l'entreprise. Ainsi, les actions des utilisateurs sur les réseaux sociaux effectuées sur des ressources annotées avec des tags du SI de l'entreprise sont utilisées pour construire le profil utilisateur sur les médias sociaux.

3. *Le CRM de l'entreprise*, contenant une base de connaissances sur les clients et prospects de l'entreprise (en bas à gauche sur la figure 4.1), est aussi utilisé pour compléter les profils des utilisateurs connus et identifiés dans le SI de l'entreprise, et pour faire le lien entre les différentes identités d'un même utilisateur, soit en utilisant des services tels que *Facebook connect* ou *Google+ sign-in* ou des fédérations d'identités tels que *OpenID*, ou encore des algorithmes de mise en correspondance [Li et al., 2011]. Dans l'expérimentation décrite dans la section 4.4, le prototype de DisCoCRM contient une déclaration explicite des correspondances d'identité entre un compte de l'application Web et un compte Twitter. Nous supposons que cette base de connaissances contient des informations générales comme une adresse, un numéro de téléphone, une liste de produits achetés ou de problèmes déjà rencontrés par l'utilisateur.

Après avoir défini l'ensemble des ressources à notre disposition, nous allons détailler le modèle de profil thématique d'un utilisateur ainsi que les différentes étapes de la construction de ce profil.

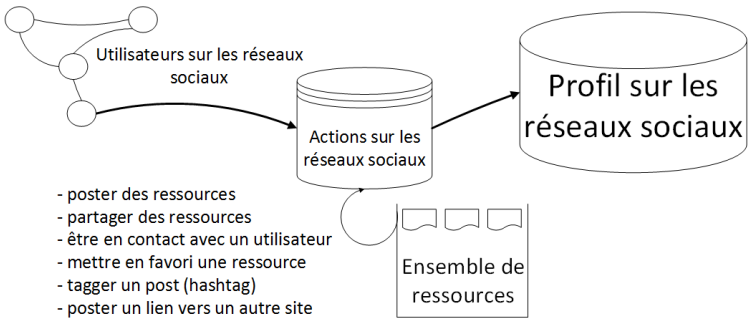

FIGURE 4.2 – Construction du profil utilisateur sur les réseaux sociaux

4.2.2/ MODÈLE DE PROFIL THÉMATIQUE

La construction du profil thématique utilisateur est basée sur les intérêts d'un utilisateur vis-à-vis des ressources du système, décrits de manière explicite et/ou implicite. Nous utilisons pour cela (i) les évaluations des ressources par un utilisateur, sous forme de notes, (ii) l'intérêt d'un utilisateur pour une ressource par son dépôt et sa consultation et (iii) le réseau de contacts de l'utilisateur.

4.2.2.1/ DÉFINITIONS DES ÉLÉMENTS DE BASE

On considère un ensemble d'utilisateurs $U = \{u_1, \ldots, u_n\}$ et un ensemble de ressources R qui peuvent être à l'intérieur du système dédié à l'entreprise R_{int} ou à l'extérieur du système R_{ext}. Nous supposons que les utilisateurs évaluent par une note $n \in \mathbb{N}$ les ressources du système $R_{int} \subseteq R$. Les notes sont modélisées par une matrice $M : U \times R_{int}$ pour un utilisateur $u_i \in U$ et une ressource $r_j \in R_{int}$, $M(u_i, r_j) = n_{ij}$. Les ressources R_{int} sont annotées par des tags, qui sont des termes issus du thésaurus du système. On note $T = \{t_1, \ldots, t_m\}$ l'ensemble des tags, chaque ressource étant annotée avec un sous-ensemble de T. Les ressources R_{ext} sont déjà annotées lorsqu'elles sont intégrées dans le système, et on ne conserve que les tags qui sont inclus dans T. Les tags associés aux ressources sont modélisés par une matrice $MT : R \times T$ définie comme suit, pour une ressource $r_j \in R$ et un tag $t_k \in T$:

$$MT(r_j, t_k) = \begin{cases} 1 & \textit{si } t_k \textit{ est associé à } r_j, \\ 0 & \text{sinon.} \end{cases} \tag{4.1}$$

On défini également la fonction liant les utilisateurs aux ressources par l'intermédiaire des tags $URT : U \times T \to \mathcal{P}(R)$.

L'approche DisCoCRM incluant une prise en compte des interactions entre les utilisateurs, nous utilisons les réseaux sociaux grand public comme source de données et par exemple, la liste des *amis* d'un utilisateur sur Facebook ou les liens *follower* / *following* de Twitter. Pour un utilisateur u_i, on distingue donc : les *followers*, qui sont les utilisateurs qui ont déclaré explicitement vouloir suivre u_i, c'est-à-dire être en contact avec u_i sans qu'il n'ait besoin de donner son accord ; des *following*, qui sont les utilisateurs que u_i déclare explicitement vouloir suivre. Dans le cadre des expériences que nous détaillerons dans les sections suivantes, nous ne nous intéressons qu'au second type d'interaction,

le premier étant indépendant des actions de u_i, alors que le deuxième correspond aux personnes à qui u_i s'intéresse.

On suppose que le site Web de l'entreprise permet aux utilisateurs de définir une liste de contacts au sein même du site, c'est-à-dire un sous-ensemble de U. Les différents liens entre les utilisateurs forment alors le réseau social interne du site Web. Les liens de contacts, qu'ils soient issus des réseaux sociaux externes à l'entreprise ou de son site Web, sont modélisés par une matrice symétrique $A : U \times U$ définie comme suit, pour deux utilisateurs $u_i, u_j \in U$:

$$A(u_i, u_j) = \begin{cases} 1 & \textit{si } u_i \textit{ est en contact avec } u_j, \\ 0 & \text{sinon.} \end{cases} \tag{4.2}$$

À partir de ces différents éléments de base nous allons définir et construire le profil thématique d'un utilisateur.

4.2.2.2/ CONSTRUCTION DU PROFIL THÉMATIQUE

L'objectif de l'approche proposée est d'exploiter la sémantique des actions des utilisateurs pour regrouper ces derniers en communautés thématiques, en se basant sur les ressources qu'ils apprécient, et en tenant compte de leurs actions sur la plateforme et des liens sociaux qu'ils développent. Pour cela nous construisons un profil unique pour chaque utilisateur en trois étapes intégrant chacune une des composantes.

Étape 1 : définition du profil explicite

Nous calculons le degré d'affinité da_{ij} entre un utilisateur u_i et un tag t_j :

$$da_{ij} = \frac{|URT(u_i, t_j)|}{|R|} \times \frac{\sum n_{ij}}{n_{max} \times |n_{ij}|} \tag{4.3}$$

avec

- $URT(u_i, t_j)$ est l'ensemble des ressources notées par l'utilisateur u_i où le tag t_j apparaît et $|R(u_i, t_j)|$ la cardinalité de cet ensemble ;
- la seconde fraction de la formule est la moyenne des notes données par l'utilisateur u_i aux ressources de $URT(u_i, t_j)$ divisée par n_{max}, la note maximale donnée aux ressources par les utilisateurs, afin d'obtenir une valeur comprise entre 0 et 1.

Comme nous voulons prendre en compte le comportement de l'utilisateur dans le système, que ce soit sur le site Web de l'entreprise ou sur les réseaux sociaux, nous ajoutons à l'expression de da_{ij} deux autres expressions centrées utilisateur (u_i) pour modéliser :

- son intérêt pour une ressource : le dépôt ou la consultation d'une ressource, le tweet, re-tweet, la mise en favori sur Twitter ou le post ou partage de ressources sur Facebook ;
- son réseau social : en prenant en compte les tags de ses contacts.

Étape 2 : prise en compte du comportement

Afin d'intégrer ces différents éléments, nous définissons un degré d'affinité d'_{ij}, utilisant l'historique de consultation de u_i sur le site Web, ainsi que ses tweets, re-tweets et favoris

sur Twitter et les ressources postées et partagées sur Facebook. La formule de calcul de d'_{ij} est définie comme suit, avec pour exemple de réseau social Twitter :

$$d'_{ij} = a \times \frac{|R_{consult}(u_i, t_j)|}{|R_{consult}|} + b \times \frac{|R_{tweet}(u_i, t_j)|}{|R_{tweet}|} + c \times \frac{|R_{re-tweet}(u_i, t_j)|}{|R_{re-tweet}|} + d \times \frac{|R_{bookmark}(u_i, t_j)|}{|R_{bookmark}|} \quad (4.4)$$

avec

- $R_{consult}$ l'ensemble des ressources R_{int} consultées par u_i
- R_{tweet}, $R_{re-tweet}$ et $R_{bookmark}$ sont respectivement les ensembles des tweets, des re-tweets et des favoris de u_i sur Twitter
- a, b, c et d des pondérations avec $a + b + c + d = 1$

Les pondérations peuvent être utilisées pour donner un poids plus ou moins important à une partie du degré d'affinité, en fonction du comportement des utilisateurs au sein du système. Par exemple, il est possible de mettre en avant les tweets et les ressources consultées si les utilisateurs ne re-tweetent pas et n'ont pas beaucoup de favoris sur Twitter.

Étape 3 : prise en compte des contacts

Afin de prendre en compte les informations provenant des utilisateurs en contact avec u_i, nous utilisons le premier degré da pour chaque utilisateur u_k étant en contact avec u_i. Ainsi, pour un utilisateur u_i et un tag t_j, nous définissons dc_{ij} comme suit :

$$dc_{ij} = \frac{\sum_{k \in A(u_i)} da_{kj}}{m} \quad (4.5)$$

avec

- $A(u_i) \subseteq U$ l'ensemble des contacts de u_i et $m = |A(u_i)|$
- da_{kj} le degré d'affinité de l'utilisateur u_k avec le tag t_j, avec $u_k \in A(u_i)$

Agrégation des trois composantes.

La combinaison des trois paramètres pris en compte dans notre approche (notes, consultations et actions sur les réseaux sociaux, contacts) nous permet de définir le degré d'affinité d_{ij} entre un utilisateur u_i et un tag t_j de la manière suivante :

$$d_{ij} = \alpha \times da_{ij} + \beta \times dc_{ij} + \gamma \times d'_{ij} \quad (4.6)$$

- avec α et β et γ des pondérations telles que $\alpha + \beta + \gamma = 1$

Le profil de l'utilisateur u_i, noté X_i, nommé dans la suite profil affiné, est le vecteur de ses degrés d'appartenance à chaque tag : $X_i = (d_{i1}, d_{i2}, ..., d_{ij})$. Il convient de rappeler que l'ensemble des tags est fixe pour la plateforme et dépendant du domaine (covoiturage, santé, alimentation, etc.). Par conséquent, le nombre de composantes du profil peut atteindre au maximum quelques centaines d'éléments.

Le modèle du profil a été instancié et testé pour une plateforme CRM collaborative développée par l'entreprise partenaire (Teletech International) pour le compte d'un pôle de compétitivité dans le domaine de la nutrition et de la santé. Nous développerons cet aspect dans la section dédiée aux expérimentations.

4.3/ DÉTECTION DE COMMUNAUTÉS

Sur la base des profils thématiques construits à partir des informations recueillies depuis la plateforme CRM et depuis les réseaux sociaux, des communautés d'utilisateurs peuvent être extraites. Dans l'approche que nous développons nous avons retenu deux catégories d'algorithmes pour l'analyse des communautés : la première considère un profil utilisateur comme un point dans un espace multi-dimensionnel et la seconde considère le profil comme un sommet d'un graphe pondéré éventuellement orienté. Dans ce dernier cas le profil d'un utilisateur u_i peut être vu comme l'ensemble des liens pondérés (les degrés d'affinité) entre les nœuds du graphe (utilisateurs et tags).

L'objectif est de fournir au *community manager* un ensemble d'outils pour affiner sa connaissance sur les communautés et pouvoir décider des actions à mener, par exemple une campagne marketing ciblée. Dans ce contexte, nous avons identifié trois cas d'utilisations type.

Dans le premier cas, nous supposons que le *community manager* n'a pas de connaissance sur la population qu'il étudie, cependant il peut définir l'ensemble des mots clés (les tags) qui sont pertinents dans son domaine et estimer le nombre de communautés à rechercher ou donner un intervalle pour ce nombre.

Dans le second cas, le *community manager*, de par son expérience métier, a une intuition des communautés existantes, il utilise l'algorithme pour confronter sa connaissance empirique aux résultats de l'algorithme appliqué sur les données recueillies. Cependant, il a besoin d'éléments sémantiques pour analyser ou interpréter le résultat de l'algorithme. Nous proposons une caractérisation des communautés extraites sous la forme d'un ensemble de tags représentatifs.

Enfin, dans le troisième cas d'utilisation, le *community manager* à une connaissance précise des termes utilisés dans son domaine, qu'il peut organiser dans une hiérarchie, qui peut être utilisée pour piloter l'algorithme.

Pour les trois cas d'utilisations, nous avons sélectionné les algorithmes suivants : l'algorithme K-means et la méthode de Louvain [Blondel et al., 2008].

Comme nous l'avons présenté dans le chapitre précédent, la classification par l'algorithme K-means est une des techniques de classification non supervisées les plus utilisées. Nous l'avons retenue car elle converge rapidement après quelques itérations. Cela permet d'effectuer plusieurs simulations, avec un nombre de classes différent, et de laisser le choix au *community manager* d'interpréter les résultats. L'espace représentant les données sur lesquelles l'algorithme K-means travaille est à m-dimensions, m étant le nombre de tags présents dans le thésaurus métier identifié par le *community manager*. De par la construction des degrés d'affinité, chaque dimension ou axe a un intervalle de valeurs comprises entre 0 et 1. Le barycentre de la communauté peut être utilisé comme utilisateur "type" de cette communauté. Les barycentres des communautés peuvent-être utilisés lorsque de nouveaux utilisateurs seront ajoutés au système afin de les classer dans la communauté la plus proche de leur profil.

La méthode de Louvain, en utilisant une représentation sous forme de graphe, permet d'introduire les éléments sémantiques nécessaires au troisième cas d'utilisation identifié. Les données en entrée étant un graphe pondéré utilisateurs et tags, la méthode permet de faire ressortir les tags associés à une communauté et ainsi proposer une caractérisation qui facilitera l'interprétation des résultats. En outre, la méthode a l'avantage

d'être très rapide, mais elle ne donne pas forcément les partitions optimales du graphe.

L'introduction des pondérations (formules 4.4 et 4.6) permet la prise en compte des différents aspects du comportement de l'utilisateur et d'en mettre un ou plusieurs en avant par rapport à d'autres. Elles nous permettent aussi d'obtenir un outil générique capable de s'adapter au contexte d'utilisation et aux souhaits du *community manager*. Cependant, trouver les "bonnes" pondérations n'est pas simple. Il est possible de les fixer de manière arbitraire, ou d'utiliser des *templates* prédéfinis ayant été adaptés au domaine métier. Toutefois, le *community manager* peut avoir une connaissance *a priori* des membres de son réseau et de ses communautés. Cette connaissance peut être prise en compte en utilisant des méthodes d'apprentissage améliorant les pondérations en fonction du contexte, par exemple avec une méthode de *feature sampling*.

4.4/ EXPÉRIMENTATIONS

Nous avons testé notre approche sur un jeu d'essai incluant des utilisateurs représentatifs de la plateforme CRM développée par l'entreprise Teletech International pour le compte d'un pôle de compétitivité. Les utilisateurs de l'application, ou clients, sont des industriels, des laboratoires de recherche et des experts du domaine de l'agro-alimentaire.

Le jeu d'essai est issu d'une étude du comportement d'utilisateurs sur une partie de la plateforme : une application de type base de connaissances. Celle-ci permet de stocker un ensemble de connaissances spécifiques à une thématique donnée, dans notre cas il s'agit des thématiques du goût, de la nutrition et de la santé. L'application peut être considérée comme une plateforme d'échanges, car les utilisateurs ont la possibilité de chercher, poster, noter, annoter (de manière privée) et commenter (de manière publique) les ressources et se créer un réseau de contacts au sein de la plateforme. Les connaissances sont organisées par un thésaurus. Les différentes données utilisées pour l'expérimentation sont présentées en annexe.

Notre jeu d'essai contient environ 20 utilisateurs, 30 tags et 50 ressources. Nous nous intéressons uniquement aux actions de poster, consulter et noter une ressource. Ces ressources sont des articles de recherche, des synthèses d'études, des comptes-rendus de réunion, au format PDF. Chaque ressource est annotée avec un ensemble de tags issus du thésaurus dont un extrait est représenté dans la figure 4.3. Au niveau du réseau social d'un utilisateur, nous ne prenons en compte que la partie interne de ce réseau. Le comportement des utilisateurs est assez variable, certains n'échangent que sur des domaines très pointus, alors que d'autres restent très généraux et ne s'intéressent qu'aux branches hautes du thésaurus. La plupart des utilisateurs s'intéressent à plusieurs thèmes, mais consultent des ressources dont le thème n'est pas obligatoirement lié à leurs centres d'intérêts. Les ressources sont évaluées par les utilisateurs au moyen d'un système de notation. Les notes possibles sont comprises entre 1 et 5. Si un utilisateur n'a pas noté une ressource, la note est de 0.

L'expérimentation comporte deux étapes : 1) la construction du profil thématique de chaque utilisateur en calculant les degrés d'affinité et 2) la détection des communautés d'utilisateurs.

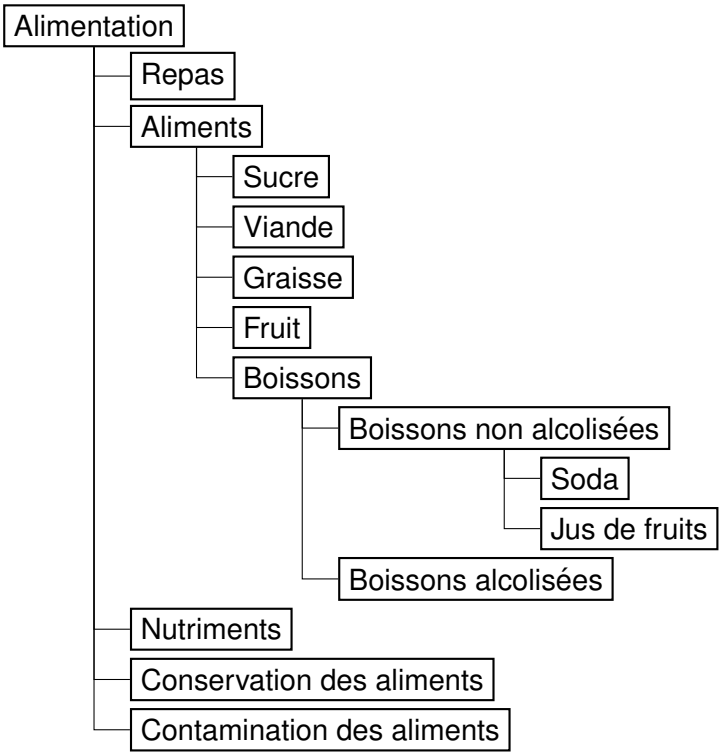

FIGURE 4.3 – Extrait de thésaurus dans le domaine alimentaire

4.4.1/ CONSTRUCTION DU PROFIL UTILISATEUR

Nous avons construit une matrice M des notes attribuées par chaque utilisateur pour chaque ressource. Le tableau 4.1 montre un extrait de cette matrice pour 4 utilisateurs et 5 ressources. Chaque ressource étant annotée par plusieurs tags, nous avons la liste des ressources avec les tags qui lui sont associés. Le tableau 4.2 montre un extrait de ces annotations. Nous avons ensuite construit la matrice M_d des degrés d'affinité des utilisateurs avec le différents tags, en nous basant sur les notes qu'ils ont attribuées aux ressources (formule 4.3). Le tableau 4.3 montre un extrait de ces résultats pour 4 utilisateurs et 4 tags. Ces éléments constituent le profil d'un utilisateur c'est-à-dire le vecteur de ses degrés d'affinité avec chaque tag.

Nous avons ensuite affiné le calcul du profil des utilisateurs en prenant en compte leur comportement implicite, c'est-à-dire les consultations des ressources (première partie de la formule 4.4) et leur réseau de contacts interne sur la plateforme (formule 4.5). Le tableau 4.4 illustre le degré d'affinité qui servira de profil en prenant en compte ces deux paramètres.

Nous avons expérimenté avec deux configurations de pondérations différentes, une première privilégiant les notes données par les utilisateurs, et une deuxième donnant le même poids à son comportement et son réseau social qu'à son profil explicite. Pour la première expérimentation, nous avons donc fixé $\alpha = 0{,}6$, $\beta = 0{,}3$ et $\gamma = 0{,}1$, pour la seconde $\alpha = 0{,}5$, $\beta = 0{,}3$, $\gamma = 0{,}2$. Le tableau 4.4 donne un exemple de profil pour la première expérimentation. Pour rappel, la pondération α correspond aux notes données par l'utilisateur, β à son comportement, et γ à son réseau social.

	R_1	R_2	R_3	R_4	R_5
u_1	4	0	5	4	5
u_3	5	4	4	3	4
u_{10}	1	0	0	0	0
u_{11}	5	3	4	5	3

TABLE 4.1 – Exemples de notes associées à des ressources

Ressources	Tags
R_1	Repas, Viande, Graisse
R_2	Nutriments
R_3	Contamination des aliments, Viande, Fruit
R_4	Conservation des aliments, Viande, Fruit
R_5	Sucre, Fruit, Soda, Jus de fruits

TABLE 4.2 – Exemples d'annotations associées à des ressources

Nous pouvons déjà remarquer par exemple que la valeur d_{ij} de l'utilisateur u_1 pour le tag $Repas$ a augmenté. Ceci peut s'expliquer par l'existence d'un lien indirect entre les utilisateurs u_1 et u_{10} (contact) et par le degré d'affinité de l'utilisateur u_{10} avec le tag $Repas$ associé au fait que u_1 consulte régulièrement des documents annotés par ce tag.

4.4.2/ DÉTECTION DE COMMUNAUTÉS ET BILAN DE L'EXPÉRIMENTATION

Pour tous les tableaux représentant les communautés, la première colonne, notée C#, correspond aux différentes communautés avec leur numéro. Les communautés obtenues à partir des profils affinés, illustrées dans les tableaux 4.5, 4.6, 4.7, 4.8 et 4.9 ont pour différence une pondération plus forte pour l'historique de consultation, pour la colonne *Profil affiné 2* et une pondération plus forte pour les ressources notées, pour la colonne *Profil affiné 1*. Les pondérations influent sur les degrés et donc sur les communautés qui en résultent.

4.4.2.1/ MÉTHODE DES K-MEANS

Pour la détection de communautés d'utilisateurs, nous avons utilisé le logiciel libre de data-mining WEKA [Witten et al., 2005] qui nous a donné les résultats présentés dans le tableau 4.5. Les représentants des communautés sont en italique. Après plusieurs essais de l'algorithme avec un nombre différent de communautés, la valeur de 5 communautés est apparue comme raisonnable pour les trois profils. Dans la suite de la section, nous interprétons les résultats (tableau 4.5).

u_1, u_3 et u_{11} notent des ressources sur des thèmes proches au niveau de la structure du thésaurus, mais différents : Viande, Graisse, Obésité pour u_1 ; Repas, Nutriments, Jus de Fruits pour u_3 ; et u_{11} s'intéresse à la partie générale Alimentation du thésaurus sans se spécialiser dans un ou plusieurs domaines spécifiques.

u_1 a consulté 14 ressources et u_3 13 ressources. Parmi les ressources consultées par u_1, 4 ont été annotées avec le tag Repas, et pour u_3 2 ont été annotées avec le tag Repas,

	Repas	Nutriments	Viande	Graisse
u_1	0,0008	0,0001	0,0251	0,0130
u_3	0,0143	0,0057	0,0076	0,0011
u_{10}	0,0005	0,0002	0,0034	0
u_{11}	0,0074	0,0032	0,0156	0,0025

TABLE 4.3 – Extrait de composantes de profils thématiques explicites

	Repas	Nutriments	Viande	Graisse
u_1	0,0215	0,0019	0,0415	0,0242
u_3	0,0319	0,0112	0,0275	0,0199
u_{10}	0,0003	0,0002	0,0096	0
u_{11}	0,0096	0,0173	0,0341	0,0054

TABLE 4.4 – Extrait de composantes combinant les composantes explicites et implicites dans le profil affiné

ce qui fait augmenter leur degré d'affinité respectif à ce tag. u_1 est en contact avec u_3 et u_{10}, qui ont tous les deux un degré d'affinité non nul au tag Repas, ce qui fait augmenter le degré d'appartenance de u_1 au tag Repas. u_3 est intéressé par le tag Nutriments, alors que u_1 ne l'est pas. Comme u_1 et u_3 sont en contact et que u_1 a consulté quelques ressources associées au tag Nutriments, le degré d'affinité de u_3 à ce tag augmente en affinant les degrés. u_{11} est lui en contact avec u_2, dont les intérêts sont différents : Infection, Bactérie, Parasite, etc.

C#	Profil explicite — α = 1 et β = 0 et γ = 0	Profil affiné 1 — α = 0,6 et β = 0,3 et γ = 0,1	Profil affiné 2 — α = 0,5 et β = 0,3 et γ = 0,2
1	u_1, u_4, u_{12}	$u_1, u_3, u_4, u_{11}, u_{12}, u_{13}, u_{20}$	$u_1, u_3, u_4, u_8, u_{11}, u_{12}, u_{13}, u_{17}, u_{18}, u_{20}$
2	$u_2, u_6, u_7, u_9, u_{10}, u_{13}, u_{15}, u_{18}$	$u_2, u_6, u_7, u_8, u_9, u_{10}, u_{15}, u_{18}$	$u_2, u_6, u_7, u_9, u_{10}, u_{15}$
3	u_3, u_{14}, u_{20}	u_5, u_{16}, u_{17}	u_5, u_{16}
4	$u_5, u_8, u_{11}, u_{16}, u_{17}$	u_{14}	u_{14}
5	u_{19}	u_{19}	u_{19}

TABLE 4.5 – Communautés obtenues pour les profils non affiné et affinés, détectées avec l'algorithme K-Means

En observant le comportement des utilisateurs, on remarque que u_1, u_3 et u_{11} partagent les mêmes intérêts, principalement les thèmes Repas, Graisse, Obésité et Viande. Il semble donc naturel que ces trois utilisateurs soient regroupés dans la même communauté. En utilisant les degrés non affinés, ces utilisateurs sont dans des communautés séparées. Cependant, en construisant les communautés à partir des profils affinés, ces utilisateurs se retrouvent bien dans la même communauté.

L'utilisateur u_{14} se retrouve seul dans une communauté avec les degrés affinés alors que ce n'était pas le cas avec les degrés non affinés. Il ne consulte pas beaucoup de ressources, et pas les mêmes que u_3 et $_{20}$ qui étaient dans sa communauté, ce qui fait diminuer les degrés d'affinité des tags qu'ils ont en commun. De plus, il est en contact avec des utilisateurs qui ont des intérêts complètement différents des siens.

Les utilisateurs u_8, u_{17}, u_{18} ne sont pas dans la même communauté que u_1 dans la colonne *Profil affiné 1*. Cependant, ils consultent beaucoup de ressources consultées aussi par

les utilisateurs de la communauté de u_1, et sur des thématiques proches des intérêts de la communauté de u_1. Ils sont donc passés dans la communauté de u_1.

L'utilisation des contacts d'un utilisateur et de son historique de navigation permet d'affiner le profil et de regrouper des utilisateurs au comportement similaire dans la même communauté, ce qui n'était pas toujours le cas avec un profil prenant en compte uniquement les notes données par un utilisateur à des ressources.

4.4.2.2/ MÉTHODE DE LOUVAIN

Nous avons ensuite utilisé la méthode de Louvain, qui ne nécessite pas de définir le nombre de communautés *a priori* et vise à maximiser la modularité. Les données en entrée de la méthode sont un graphe dont les sommets sont les utilisateurs et les tags, et les liens sont pondérés par le degré d'affinité correspondant aux différentes composantes du vecteur qui décrit le profil d'un utilisateur. Le nombre de nœuds de ce graphe est la somme du nombre d'utilisateurs et des tags.

On peut remarquer que le nombre de communautés varie en fonction des profils utilisés, passant de 5 pour la méthode des K-Means à 6 pour la méthode de Louvain. Cependant, nous devons contrôler et interpréter les résultats par rapport à deux contraintes propres à la méthode que nous avons décrites dans le chapitre de l'état de l'art : le seuil de résolution et la non stabilité des résultats. Pour ce faire, la qualité globale de la structure communautaire est donnée par la valeur de la modularité (valeur Q dans les en-têtes des colonnes des tableaux). Le seuil de résolution (valeur r dans les en-têtes des colonnes des tableaux) dépend du nombre de liens et donne la taille minimum d'une communauté détectable et significative.

Du fait du processus de construction du profil thématique, des liens ayant un poids très faible sont créés. Nous nous sommes fixés comme règle de ne retenir que les liens les plus significatifs, c'est-à-dire contribuant à plus de 95% de la somme totale du poids des liens. Pour répondre à la non stabilité, nos exécutons plusieurs fois l'algorithme et nous retenons la configuration majoritaire.

Le tableau 4.6 présente le résultat de la méthode de Louvain pour les communautés d'utilisateurs et le tableau 4.7 présente la liste des tags qui caractérisent chaque communauté. Les communautés suivies d'une étoile sont en dessous du seuil de la limite de résolution induite par la modularité, par conséquent elles ne peuvent pas être suivies d'une interprétation par le *community manager*. Un travail de caractérisation à un plus faible niveau de granularité est donc nécessaire. Notre approche, utilisant la notion de communauté locale, sera présentée à la section suivante. La figure 4.4 donne un exemple de représentation orientée utilisateur qui permet une visualisation des communautés et de leur caractérisation thématique plus aisée à interpréter que les tableaux de résultats bruts de l'algorithme. L'interprétation de la visualisation doit se faire en respectant les précautions précédentes.

Des similarités existent entre les communautés trouvées par le K-Means et celles trouvées par la méthode de Louvain. En effet, pour le profil explicite, u_1 et u_{12} sont dans la même communauté, u_2, u_6, u_9 et u_{10} ainsi que u_5 et u_{17}. Dans le cas des communautés obtenues avec le profil affiné 1, u_2, u_6, u_8, u_9 et u_{18} sont dans la même communauté, ainsi que u_3, u_4 et u_{11}. Dans les cas des communautés obtenues avec le profil affiné 2, u_2, u_6, u_9 sont dans la même communauté.

C#	Profil explicite — α = 1 et β = 0 et γ = 0 (r=10, Q=0,29)	Profil affiné 1 — α = 0,6 et β = 0,3 et γ = 0,1 (r=12, Q=0,23)	Profil affiné 2 — α = 0,5 et β = 0,3 et γ = 0,2 (r=12, Q=0,24)
1	$u_2, u_6, u_8, u_9, u_{10}$	u_7, u_{15} (*)	u_4, u_{12}, u_{14} (*)
2	u_5, u_{17} (*)	$u_1, u_2, u_6, u_8, u_9, u_{18}, u_{20}$	$u_2, u_5, u_6, u_8, u_9, u_{18}$
3	u_4, u_{18}, u_{20} (*)	$u_3, u_4, u_{11}, u_{12}, u_{13}, u_{14}, u_{17}$	u_3, u_{11}, u_{13} (*)
4	u_1, u_{11}, u_{12} (*)	u_{10}, u_{16}, u_{19} (*)	u_1, u_{20} (*)
5	$u_3, u_{14}, u_{16}, u_{19}$ (*)	u_5 (*)	u_{16}, u_{17}, u_{19} (*)
6	u_7, u_{13}, u_{15} (*)		u_7, u_{10}, u_{15} (*)

TABLE 4.6 – Communautés extraites à partir des profils non affiné et affinés avec la méthode de Louvain

C#	Profil explicite	Profil affiné 1 — α = 0,6 et β = 0,3 et γ = 0,1	Profil affiné 2 — α = 0,5 et β = 0,3 et γ = 0,2
1	Contamination-des-aliments, Bactérie, Parasite, Virus, Grippe, Rubéole	Alimentation, Aliments, Allergie, Facteur-allergène	Sucre, Boissons-non-alcoolisées, Soda, Jus-de-fruits
2	Nutriments, Maladie, Maladie-de-la-nutrition, Infection	Contamination-des-aliments, Conservation-des-aliments, Viande, Bactérie, Parasite, Virus, Grippe, Rubéole	Nutriments, Contamination-des-aliments, Conservation-des-aliments, Boissons-alcoolisées, Maladie-de-la-nutrition, Infection, Bactérie, Virus, Grippe
3	Conservation-des-aliments, Boissons, Boissons-non-alcoolisées, Soda, Boissons-alcoolisées, Immunopathologie	Repas, Sucre, Graisse, Boissons, Boissons-non-alcoolisées, Soda, Jus-de-fruits, Boissons-alcoolisées, Maladie, Obésité	Repas, Graisse, Obésité, Hémopathie
4	Aliments, Sucre, Viande, Graisse	Légumes, Carence-alimentaire	Viande, Parasite
5	Fruit, Légumes, Jus-de-fruits, Carence-alimentaire, Obésité	Nutriments, Maladie-de-la-nutrition, Infection	Fruit, Légumes, Boissons, Maladie, Carence-alimentaire, Immunopathologie
6	Alimentation, Repas, Hémopathie, Allergie, Facteur-allergène		Alimentation, Aliments, Allergie, Facteur-allergène, Rubéole

TABLE 4.7 – Caractérisation au moyen des tags associés aux communautés détectées avec les profils non affiné et affinés

Cependant, u_5 et u_{17} n'ont qu'un seul intérêt en commun (Maladie de la nutrition), mais sont dans la même communauté lorsque l'on utilise les profils explicites, que ce soit avec la méthode de Louvain ou avec le K-Means. Avec la méthode de Louvain, ces deux utilisateurs ne sont plus dans la même communauté lorsque l'on utilise les profils explicites et implicites, alors qu'avec le k-means, dans le cas des communautés obtenues avec les profils affinés, ils restent toujours dans la même communauté.

Il y a des groupes d'utilisateurs qui sont toujours dans la même communauté, quelles que soient les pondérations : par exemple u_2, u_6, u_8 et u_9, intéressés par les tags Contami-

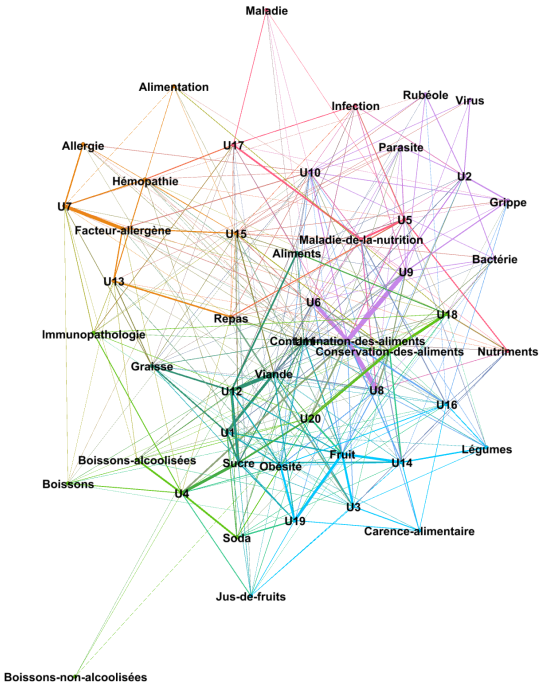

FIGURE 4.4 – Exemple de représentation des six communautés pour le profil explicite

nation, Conservation, Infection. Ces tags sont bien ressortis par la méthode de Louvain dans le tableau 4.7.

Considérons le cas des utilisateurs u_1, u_3 et u_{11}. On remarque que u_1 et u_{11} sont dans la même communauté lorsque seul le profil explicite est pris en compte ; mais il ne s'agit pas de communautés significatives, car en dessous du seuil de résolutions. Cependant, lorsque l'on introduit la prise en compte du comportement, u_3 et u_{11} sont dans la même communauté encore non significative alors que u_1 est dans une communauté différente et significative cette fois.

D'une manière plus générale, les profils thématiques affinés ajoutent des liens entre les nœuds du graphe, qui dans notre jeu d'essai font augmenter la valeur de la limite de résolution. Cependant, dans le cas d'une plateforme en exploitation, le nombre d'utilisateurs est très supérieur au nombre de termes utilisés pour annoter les ressources. En effet, les termes sont proposés par le système et non pas choisis directement par les utilisateurs et leur nombre peut être contrôlé, tout comme leur variabilité sémantique. Le seuil et la valeur de la modularité sont donc tout à fait acceptables pour notre jeu d'essai et ne peuvent qu'être améliorés avec des données d'une plateforme en exploitation.

4.4.2.3/ MÉTHODE DE LOUVAIN PILOTÉE PAR UNE CONNAISSANCE DU DOMAINE

L'objectif est d'utiliser la sémantique apportée par le thésaurus afin de renforcer la signification des liens explicites et implicites entre les utilisateurs et les tags. Nous proposons d'utiliser la hiérarchie de termes du thésaurus (figure 4.3) pour piloter la méthode de Louvain. Pour cela, nous transformons le thésaurus en graphe pondéré au moyen d'une distance sémantique inverse à la profondeur des termes dans le thésaurus. Autrement dit, cette distance donne plus d'importance aux termes spécifiques qu'aux termes généraux. Le thésaurus sera injecté dans le graphe en ajoutant les liens entre les tags et les degrés d'affinité entre utilisateurs et tags seront mis à jour.

Par conséquent, nous avons besoin de définir une similarité sémantique entre un nœud de type utilisateur et un nœud de type tag inclus dans la liste des termes du thésaurus. Nous nous sommes appuyés sur l'état de l'art [Gan et al., 2013] et avons retenu une mesure basée sur la structure hiérarchique du thésaurus.

Pour définir la similarité entre un utilisateur et un tag, nous avons besoin tout d'abord d'une mesure de similarité entre termes du thésaurus. La similarité entre deux termes présents dans le thésaurus, pour un terme t_1 ayant comme parent direct t_2 est définie par :

$$\forall t_1, t_2 \in Th \subset N, t_1 \prec t_2,\ Sim_{termes}(t_1, t_2) = \frac{1}{2^{lg(t_1, top)}}$$

où N est l'ensemble des nœuds du graphe, $Th \subset T$ le sous ensemble des termes appartenant au thésaurus, top désigne la racine du thésaurus, $lg()$ est la longueur d'un chemin entre deux nœuds, et $\prec$ une relation d'ordre partiel de type généralisation/spécialisation associée aux termes du thésaurus. En appliquant la formule de la similarité sur le thésaurus de la figure 4.3, on obtient une version avec des liens pondérés (figure 4.5).

Ainsi, pour notre domaine, un utilisateur qui annote un document avec un tag précis cumule aussi les pondérations des termes plus généraux. La similarité entre un utilisateur et un tag (correspondant à un terme du thésaurus) est donnée par l'expression suivante :

$$Sim(u, t) = \sum_{t_i \in C_{t,top}} Sim_{termes}(t_i, t_{i+1}), avec\ u \in U, t \in Th$$

où $C_{t,top}$ désigne un chemin, c'est-à-dire une suite finie de nœuds entre le terme t et le terme racine.

La valeur de la similarité vient s'ajouter aux autres composantes dans le calcul du degré d'affinité. Le tableau 4.8 présente le résultat de la méthode de Louvain pilotée, au niveau des communautés d'utilisateurs, et le tableau 4.9 présente la liste des tags caractérisant chaque communauté. On remarque que le nombre de communautés a été réduit à 4 et que la valeur de la modularité a augmenté sensiblement. Ainsi, les communautés sont plus denses et la répartition des tags des communautés est proche de la hiérarchie du thésaurus.

Comme dans l'expérience précédente, la valeur de la modularité confirme la présence d'une structure communautaire. Les utilisateurs u_6, u_8 et u_9 notent des ressources avec des tags en commun, par exemple Contamination des aliments. Cependant, la majorité de leurs actions se fait sur des ressources différentes. u_6 consulte principalement des ressources dont les tags sont Conservation des aliments, Fruit, Légumes, Soda et Jus de

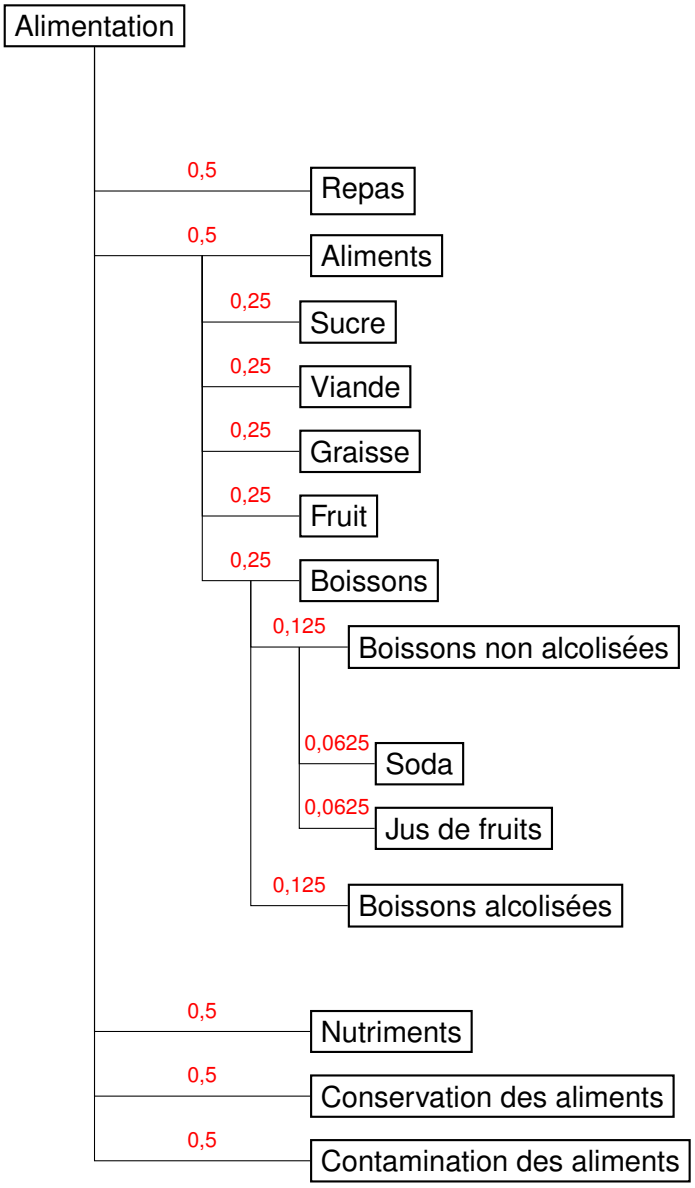

FIGURE 4.5 – Extrait du thésaurus du domaine alimentaire muni d'une distance entre les tags supportée par la relation hiérarchique

C#	Profil explicite — α = 1 et β = 0 et γ = 0 (r=11, Q=0,53)	Profil affiné 1 — α = 0,6 et β = 0,3 et γ = 0,1 (r=13,Q=0,41)	Profil affiné 2 — α = 0,5 et β = 0,3 et γ = 0,2 (r=13,Q=0,36)
1	$u_5, u_6, u_8, u_9, u_{20}$ (*)	u_5, u_8, u_{20}	u_5, u_8, u_{11}, u_{20}
2	$u_1, u_3, u_4, u_{11}, u_{12}, u_{14}, u_{15}, u_{16}, u_{18}, u_{19}$	$u_1, u_3, u_4, u_6, u_{11}, u_{12}, u_{14}, u_{16}, u_{18}, u_{19}$	$u_1, u_3, u_4, u_6, u_{12}, u_{14}, u_{16}, u_{18}, u_{19}$
3	u_7, u_{13}, u_{17} (*)	u_2, u_{17}(*)	u_{13}, u_{17} (*)
4	u_2, u_{10} (*)	$u_7, u_9, u_{10}, u_{13}, u_{15}$	$u_2, u_7, u_9, u_{10}, u_{15}$ (*)

TABLE 4.8 – Communautés obtenues pour les profils explicite, affinés avec Louvain pilotée par la connaissance du domaine

C#	Profil explicite — α = 1 et β = 0 et γ = 0	Profil affiné 1 — α = 0,6 et β = 0,3 et γ = 0,1	Profil affiné 2 — α = 0,5 et β = 0,3 et γ = 0,2
1	Alimentation, Repas, Nutriments, Contamination-des-aliments	Alimentation, Repas, Nutriments, Contamination-des-aliments	Alimentation, Repas, Nutriments, Contamination-des-aliments
2	Conservation-des-aliments, Aliments, Sucre, Viande, Graisse, Fruit, Légumes, Boissons, Boissons-non-alcoolisées, Soda, Jus-de-fruits	Conservation-des-aliments, Aliments, Sucre, Viande, Graisse, Fruit, Légumes, Boissons, Boissons-non-alcoolisées, Soda, Jus-de-fruits	Conservation-des-aliments, Aliments, Sucre, Viande, Graisse, Fruit, Légumes, Boissons, Boissons-non-alcoolisées, Soda, Jus-de-fruits
3	Boissons-alcoolisées, Maladie, Maladie-de-la-nutrition, Carence-alimentaire, Obésité, Hémopathie, Immunopathologie, Allergie, Facteur-allergène, Infection, Bactérie	Maladie, Maladie-de-la-nutrition, Carence-alimentaire	Boissons-alcoolisées, Maladie, Maladie-de-la-nutrition, Carence-alimentaire, Obésité, Hémopathie, Immunopathologie, Allergie
4	Parasite, Virus, Grippe, Rubéole	Boissons-alcoolisées, Obésité, Hémopathie, Immunopathologie, Allergie, Facteur-allergène, Infection, Bactérie, Parasite, Virus, Grippe, Rubéole	Facteur-allergène, Infection, Bactérie, Parasite, Virus, Grippe, Rubéole

TABLE 4.9 – Caractérisation par les tags des communautés obtenues au moyen des profils affinées et affinées avec la méthode de Louvain pilotée

Fruits. u_8 consulte beaucoup de ressources avec le tag Contamination des aliments. u_9 consulte des ressources sur l'infection, les bactéries, la grippe et la rubéole entre autres. u_6 est aussi en contact avec u_4, qui lui note des ressources sur les thèmes Soda et Jus de Fruits. Les utilisateurs u_6, u_8 et u_9 sont dans la même communauté en utilisant les profils explicites. Lorsque l'on introduit les profils implicites, leur différence de comportement fait qu'ils sont les trois dans des communautés séparées. u_6 passe donc dans la communauté 2 qui s'intéresse aux thèmes Soda, Jus de Fruits, etc. u_8 reste dans la communauté s'intéressant à la Contamination des aliments.

On peut aussi remarquer que u_5 et u_{17} ne sont plus dans la même communauté avec le profil explicite. Dans ce cas, u_{17} est dans la même communauté que u_{13}, ils partagent des thèmes communs : hémopathie et immunopathologie.

Dans le cas des utilisateurs u_1, u_3 et u_{11}, ils sont dans la même communauté lorsque seul le profil explicite est pris en compte et dans le cas des communautés associées aux profils affinés 1. Cependant, pour les communautés associées aux profils affinées 2, u_{11} n'est plus dans la même communauté qu'u_1 et u_3. Les 3 utilisateurs notent des ressources qui se situent dans la même partie du thésaurus. Le pilotage de la méthode de Louvain avec la hiérarchie du thésaurus permet de regrouper ces 3 utilisateurs dans la même communauté en utilisant uniquement le profil explicite, ce qui n'était pas le cas avant. Pour les communautés obtenues avec le profil affiné 2, u_1 et u_3 étant en contact ; et u_{11} étant en contact avec u_2 aux intérêts complètement différents, l'accent mis sur le réseau social laisse u_1 et u_3 dans la même communauté alors que u_{11} change de communauté.

4.4.2.4/ Bilan de l'expérimentation

Notre approche permet de regrouper les utilisateurs dans des communautés plus pertinentes que si l'on utilisait un profil explicite, sans modélisation du comportement et du réseau de contacts d'un utilisateur. Les différents paramètres de pondération des profils affinés permettent de découvrir des communautés en fonction des critères que l'on souhaite mettre en avant : basés plus sur les notes des utilisateurs, sur leur liste de contacts ou sur les consultations et dépôts. Cela permet également de privilégier plus ou moins un aspect du comportement de l'utilisateur en fonction du contexte et du type de communautés souhaitées. Il est également possible de faire varier les communautés en utilisant différentes valeurs du paramètre du nombre de classes de l'algorithme K-Means, tout en mesurant la qualité du partitionnement.

Nous avons proposé d'utiliser la méthode de Louvain pour la détection de communautés globales. Cette méthode a l'avantage d'être très rapide dans son exécution, mais ses résultats doivent être accompagnés des valeurs de la modularité et de la limite de résolution. De plus elle ne garantit pas de trouver la solution optimale et ne fournit pas la même structure communautaire lors de plusieurs exécutions ce qui conduit à garder la structure communautaire la plus fréquente. Les communautés détectées grâce à cette méthode varient de celles trouvées avec le K-Means. L'expertise sur la connaissance des utilisateurs de la plateforme nous permet de conclure que les communautés détectées sont qualitativement plus représentatives pour notre contexte d'application. Cependant, le seuil de résolution conduit à des communautés non significatives et la modularité est relativement fiable.

Le pilotage de la méthode de Louvain par la hiérarchie de termes du thésaurus permet de réduire le nombre de communautés. Cela permet de limiter le problème des utilisateurs ne s'intéressant qu'aux feuilles du thésaurus pouvant se retrouver dans une mauvaise communauté. La modularité est nettement améliorée et l'interprétation des communautés est également facilitée par la présence des termes du thésaurus.

4.5/ Détection de communautés locales et prise en compte des données de réseaux sociaux

Les communautés obtenues avec la méthode de Louvain ne sont pas toutes significatives. C'est le cas pour les communautés dont le nombre d'éléments est inférieur au seuil de résolution. Cependant, la valeur de la modularité est un indicateur de la présence d'une structure communautaire et par conséquent les éléments qui constituent les communautés non significatives doivent faire l'objet d'une analyse plus fine. La notion de communauté locale (aussi nommée ego-centrée) doit permettre cette analyse. Une communauté locale est une communauté formée autour d'un nœud d'intérêt ou d'un ensemble de nœuds d'intérêt [Danisch et al., 2013].

4.5.1/ Adaptation de l'algorithme PageRank personnalisé

Nous reprenons l'approche développée dans [Danisch et al., 2013, Danisch, 2015] avec l'algorithme CarOp (*Carry over Opinion*) qui consiste, à partir des nœuds d'intérêt (utilisateurs ou tags), à appliquer un algorithme de propagation pour détecter la ou les commu-

nautés formées autour de ces nœuds. L'algorithme itératif proposé par [Danisch, 2015] travaille sur des graphes non pondérés. Il est facilement adaptable à d'autres types de graphe, mais présente, de par sa méthode de propagation incluant une mise à l'échelle (*rescaling*), des faiblesses théoriques. La preuve de la convergence vers une distribution stationnaire n'est pas réalisable dans la majorité des cas.

Nous proposons de remplacer la méthode de propagation de l'algorithme CarOp par une méthode s'inspirant d'une marche aléatoire de type *random surfer* telle que celle proposée dans l'algorithme PageRank [Page et al., 1999]. Dans ce cas, la mesure produite par l'algorithme peut être interprétée comme une mesure de proximité des nœuds d'un graphe par rapport au(x) nœud(s) d'intérêt, ou comme la probabilité d'appartenance à une communauté formée à partir d'un ou plusieurs nœuds d'intérêt.

Avant d'utiliser l'algorithme PageRank, nous devons nous assurer que ses conditions d'applicabilité peuvent correspondre à notre modélisation ; ou bien si ce n'est pas exactement le cas, si il est possible de transformer les données sans remettre en cause le modèle, pour pouvoir appliquer l'algorithme PageRank. Reprenons la formulation de l'algorithme PageRank telle qu'elle est donnée dans le chapitre de l'état de l'art :

$$X_{t+1} = \alpha T X_t + (1-\alpha)X_0$$

La matrice de transition T représentant les liens entre les pages doit être, dans notre cas, remplacée par la matrice W qui contient les poids des liens entre utilisateurs et tags, c'est-à-dire les liens entre les utilisateurs seulement, entre les utilisateurs et les tags et entre les tags seulement. La facteur d'amortissement α reste inchangé, $\alpha \in]0,1[$. Pour étudier les conditions d'applicabilité de l'algorithme, nous revenons à une forme moins condensée de la matrice de transition $G = \alpha S + (1-\alpha)E$. Généralement pour $E = (e_{ij})_{1\leq i,j\leq n}$ on a $e_{ij} = 1/n$. E sert à éviter que les nœuds sans successeurs ne stoppent la diffusion ou la marche aléatoire, en renvoyant de manière équiprobable vers un nœud quelconque. Dans notre cas, le graphe n'est pas orienté et est connexe. Traduit sur le profil d'un utilisateur, cela signifie qu'un utilisateur est décrit par au moins une valeur. La matrice S ne contient donc pas de ligne nulle. On peut donc modifier l'expression de G de manière à faire apparaître un vecteur de personnalisation :

$$G = \alpha S + (1-\alpha)\mathbb{1}v^t$$

$v_j = 0$ pour tous les nœuds qui ne sont pas dans la liste des nœuds d'intérêt, pour tous les autres leur valeur est choisie de manière à ce que $\|v\| = 1$, généralement répartie de manière uniforme entre les nœuds d'intérêt.

Pour qu'il y ait convergence vers un état stable, c'est-à-dire une valeur de X_{t+1} constante et donc pouvoir fournir une mesure de proximité entre les nœuds, on doit vérifier que la matrice $G = (g_{ij})_{1\leq i,j\leq n}$ est stochastique, c'est-à-dire $\sum_{1\leq j\leq n} g_{ij} = 1$ donc S doit être aussi stochastique. Par conséquent il suffit de normaliser (effectuer une mise à l'échelle) la matrice W et on obtient l'expression :

$$G = \alpha(MW) + (1-\alpha)\mathbb{1}v^t$$

où $M = (m_{ij})_{1\leq i,j\leq n}$ est une matrice carrée diagonale de taille n dont les éléments diagonaux sont l'inverse de la somme des lignes de W. $m_{ij} = 1/\sum_{i=1,n} w_{ij}$.

4.5.2/ EXPÉRIMENTATION

Pour la détection de communautés locales, nous repartons des communautés détectées par la méthode de Louvain mais non significatives car plus petites que le seuil de résolution, et nous calculons la proximité des autres nœuds par rapport à chaque utilisateur considéré individuellement comme nœuds d'intérêt. Pour chaque exécution de l'algorithme nous traçons la courbe représentant la proximité de chaque nœud par rapport au nœud d'intérêt et nous recherchons des ruptures franches sur la courbe que nous interprétons comme une frontière de communauté.

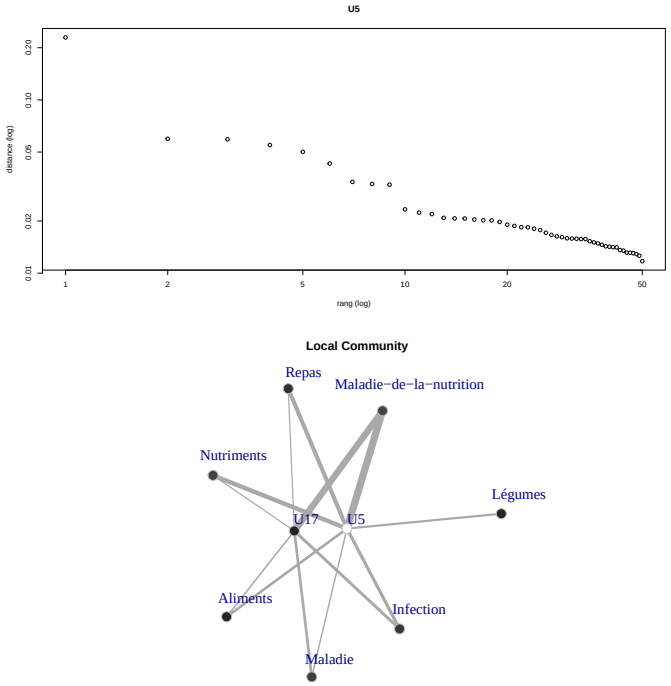

FIGURE 4.6 – Résultats de l'algorithme de détection de communauté locale centrée sur l'utilisateur u_5

En reprenant les expériences dont les résultats on été présentés dans les tableaux 4.6 et 4.8, nous illustrons les résultats de l'algorithme de détection de communautés locales centrées sur les utilisateurs dans le cas d'une communauté locale significative et d'une autre non significative.

Pour le tableau 4.6 avec le profil explicite, nous trouvons un rapprochement effectif entre les utilisateurs u_5 et u_{17}. La figure 4.6 fait apparaître un seuil au rang 9 et le graphe centré autour de u_5 montre les liens avec u_{17}. Pour l'utilisateur u_1, l'algorithme de détection de communautés locales ne montre pas d'effet de seuil sur la proximité (figure 4.7). Pour u_{18}, l'évolution de la proximité en fonction du rang fait apparaître une structure communautaire, mais cette dernière n'est constituée que de tags (figure 4.8).

Pour les deux autres expériences de la méthode de Louvain avec les profils affinés, puis celles avec l'ajout de la connaissance du domaine on ne détecte pas seuil de la proximité reflétant une communauté d'utilisateurs mais seulement un ensemble de termes liés à l'utilisateur initial, comme dans la figure 4.8. Ces communautés utilisateurs/termes

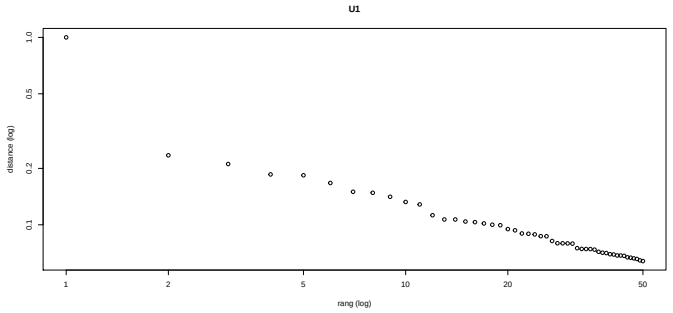

FIGURE 4.7 – Distance des nœuds en fonction du rang pour l'utilisateur u_1

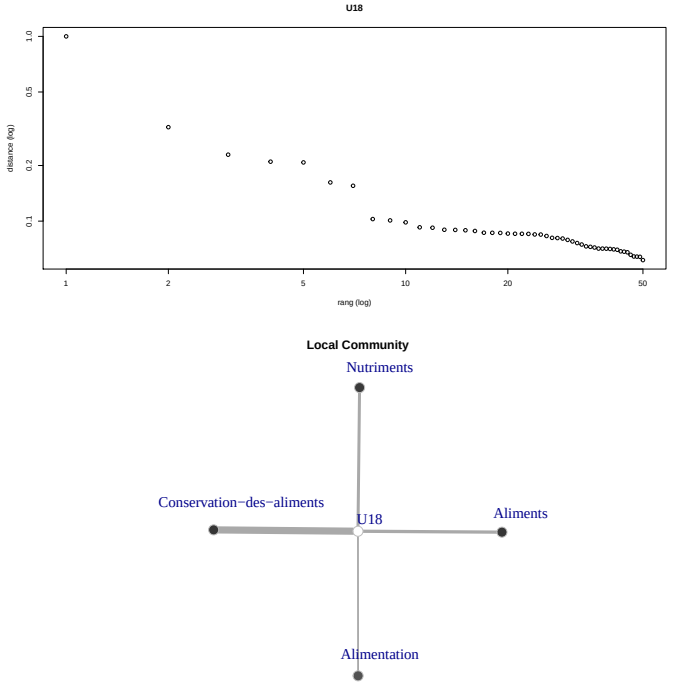

FIGURE 4.8 – Résultats de l'algorithme de détection de communauté locale centrée sur l'utilisateur u_{18}

peuvent contenir jusqu'à 10 termes alors que l'utilisateur en a spécifié explicitement au maximum 5. Cependant l'interprétation de la proximité en fonction du rang peut uniquement être basée sur la valeur de la proximité et ne pas tenir compte des sauts. Dans ce cas, on peut sélectionner les n premiers utilisateurs les plus proches du nœud d'intérêt. Dans l'exemple de la figure 4.9, qui correspond à la troisième expérience du tableau 4.8, les termes communs aux trois utilisateurs sont mis en évidence par l'algorithme lancé avec le nœud d'intérêt u_5.

Les différentes expériences nous montrent que l'algorithme de détection de communautés locales peut être exploité pour caractériser une communauté. Les résultats seront plus significatifs si le nombre d'utilisateurs du jeu d'essai augmente par rapport au nombre de termes utilisés.

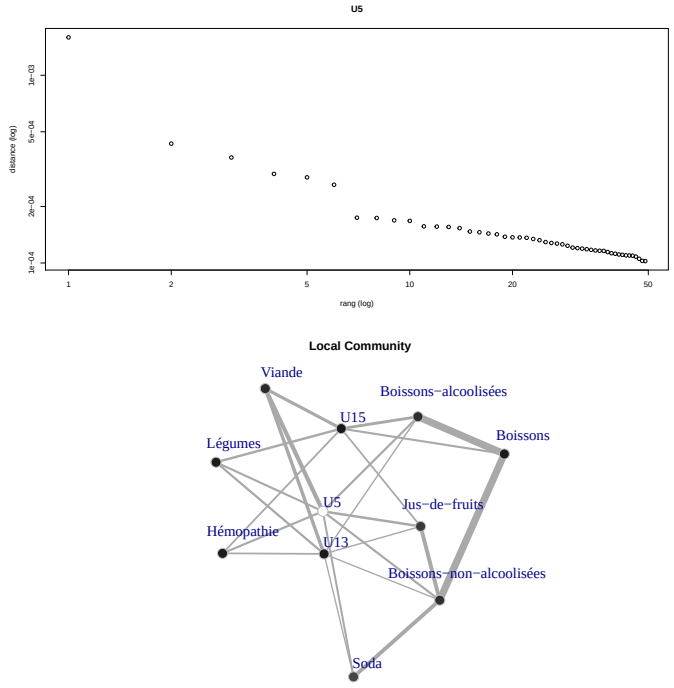

FIGURE 4.9 – Résultats de l'algorithme de détection de communauté locale centrée sur l'utilisateur U5 mettant en évidence les liens avec les utilisateurs u_{15} et u_{13}

L'algorithme peut également être utilisé pour effectuer des recommandations de tags ou d'utilisateurs. De plus, appliqué aux tags d'un utilisateur, l'algorithme peut permettre de les réordonner, non plus selon leur place dans le thésaurus, mais selon leurs liens effectifs avec les documents et les autres utilisateurs dans la plateforme. Ceci peut ensuite être utilisé pour proposer un outil de navigation personnalisée.

4.5.3/ PRISE EN COMPTE DES DONNÉES DES RÉSEAUX SOCIAUX

L'algorithme de détection de communautés locales peut s'appliquer naturellement sur des données issues des réseaux sociaux et modélisées sous la forme de graphe, par exemple des graphes utilisateurs et hashtags. Afin de déterminer les communautés formées autour d'un sujet particulier (un produit, une marque, un événement), nous appliquons notre algorithme de détection de communautés locales.

Nous avons appliqué l'algorithme sur des données Twitter captées pour étudier le domaine du co-voiturage. La collecte des données à été initiée à partir d'une liste de hashtags et des comptes spécifiques à ce domaine (225 critères). Il s'agit principalement des comptes et des hashtags associés aux entreprises de co-voiturage. La collecte s'est déroulée sur une période de 4 mois à partir d'octobre 2014 et environ 30 millions de tweets ont été récoltés.

Nous nous intéressons au hashtag *uberpop*, correspondant à l'application qui a généré de nombreux débats en 2014 pour être finalement interdite en 2015. Sur le période de collecte, plus de 5 800 utilisateurs différents ont utilisé ce hashtag. Á partir de ces

utilisateurs, nous avons construit un graphe utilisateurs-hashtags comportant 22 791 nœuds et 104 800 liens. Dans ce graphe, nous souhaitons étudier les discours négatifs autour du hashtag *uberpop*. Une étude exploratoire des co-occurrences des hashtags avec *uberpop* fait apparaître le hashtag *deleteuber*, pour *delete Uber*, ou *supprimer Uber* en français. Pour étudier la communauté locale des utilisateurs de ce hashtag et leurs liens éventuels avec d'autres hashtags, nous construisons un sous-graphe contenant toutes les co-occurrences de *uberpop* ainsi que les utilisateurs de ces co-occurrences. Puis nous lançons l'algorithme de détection de communautés locales à partir du nœud d'intérêt *deleteuber*. Les résultats sont présentés dans la figure 4.10. La courbe de la distance en fonction des nœuds fait apparaître un seuil à 5 puis un autre seuil à 25. Nous utilisons ce deuxième seuil pour représenter le graphe utilisateurs hashtags. Ce dernier met en évidence les hashtags utilisés par le compte `Taxis de Paris` et 5 autres comptes très proches du hashtag *deleteuber* mais non liés directement.

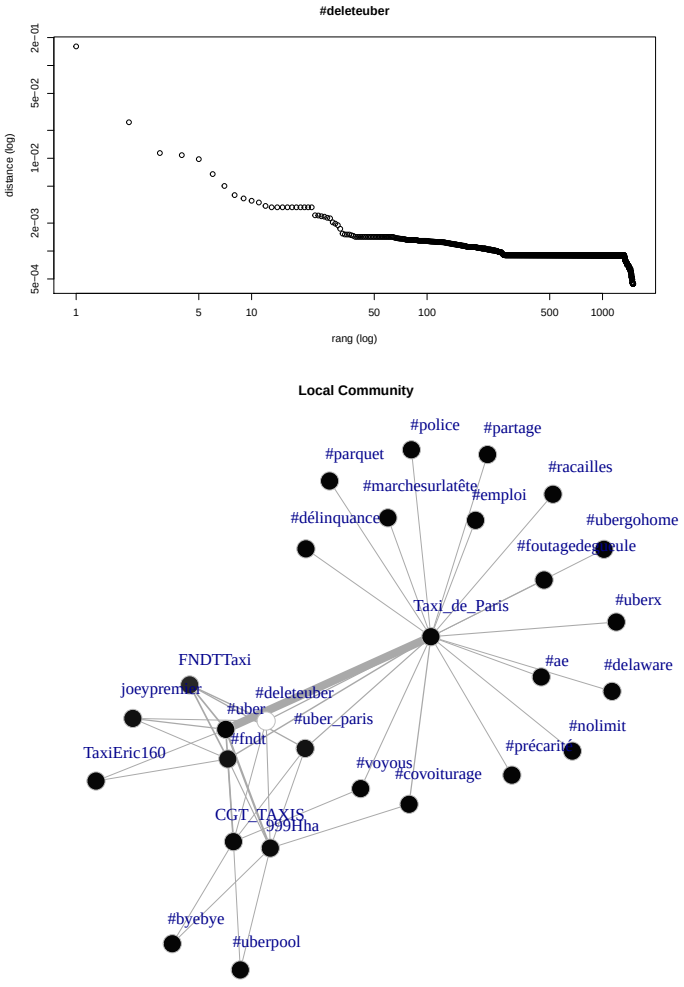

FIGURE 4.10 – Résultat de la détection de communauté locale autour du hashtag *deleteuber*

Cet exemple montre la capacité de l'algorithme à détecter de petites structures dans un graphe et, d'un point de vue métier, à identifier les hashtags utilisés autour du discours

critique.

4.6/ CONCLUSION

Dans ce chapitre, nous avons présenté une approche pour la détection de communautés destinée aux *community managers* dans le contexte de la gestion de la relation client. La détection des communautés exploite des profils utilisateurs basés sur les usages, les comportements et les contacts. Les données du profil sont collectées à partir des applications Web de l'entreprise et à partir des réseaux sociaux. Les composantes du profil sont modulables au moyen de pondérations et aboutissent à la notion de profils affinés, permettant ainsi de détecter des communautés en fonction des critères que l'on souhaite voir renforcer. Nous avons utilisé deux catégories d'algorithmes dont la complexité permet d'envisager le traitement du volume de données nécessaire au CRM : l'un opérant une classification dans un espace multi-dimensionnel (K-Means) ; l'autre, la méthode de Louvain, travaillant sur des graphes pondérés permettant de faire émerger les caractéristiques des communautés globales et de piloter la détection des communautés par la connaissance du domaine. Une série de trois expérimentations sur des jeux de données test a été présentée. Cette série d'expériences à été complétée par la proposition d'un algorithme de détection de communautés locales afin de contourner la limite de résolution de la méthode de Louvain. L'algorithme proposé permet de mettre en évidence des structures formées autour de nœuds d'intérêt. Il a été appliqué sur le graphe utilisateurs-tags et initialisé à partir des résultats obtenus avec la méthode de Louvain, puis testé sur des données plus importantes issues de Twitter.

Afin d'automatiser les analyses que peut demander le *community manager*, nous avons développé la plateforme DisCoCRM qui sera décrite dans le chapitre 6. Elle propose des interfaces Web pour les différent acteurs. Un exemple d'interface pour un gestionnaire de la relation client, en lien avec les profils et les algorithmes proposés dans le chapitre, est présenté dans les figures 4.11, 4.12. Ces figures présentent respectivement les grandes fonctionnalités accessibles depuis la page d'accueil de l'application, ainsi que la définition des pondérations et le choix des algorithmes.

FIGURE 4.11 – Grandes fonctionnalités de la plateforme DisCoCRM

FIGURE 4.12 – Gestion des paramètres pour la constitution du profil

5

PLATEFORME SCALABLE POUR LA COLLECTE, LE STOCKAGE ET L'ANALYSE DE DONNÉES ISSUES DE TWITTER

Sommaire

5.1/ INTRODUCTION

La multitude de réseaux sociaux en ligne disponibles ainsi que les potentialités des analyses des données qu'ils fournissent sont un des atouts essentiels pour s'adapter aux nombreux cas d'utilisations du Social CRM (sCRM). Ainsi, afin de prendre en compte la multiplicité des usages, il est nécessaire d'élaborer une plateforme générique pour la gestion de données sociales. Cette plateforme devra permettre de gérer la collecte, le stockage ainsi que l'analyse et la visualisation des données issues des réseaux sociaux.

Dans un premier temps, nous ne visons pas une orientation de la plateforme pour la prise de décision mais plutôt pour une utilisation dans un contexte de production de connaissances, de caractérisation ou de compréhension de phénomènes.

Afin de répondre aux multiples besoins de l'analyse des données sociales (analyse de communautés, mesure de l'influence, détections d'événements, etc.), et de réduire le temps de mise en forme des données pour des analyses en temps réel, tout en accédant rapidement à de grandes quantités de données, plusieurs paradigmes de modélisation des données (en vue de leur stockage) doivent être envisagés [Sadalage et al., 2012, Lim et al., 2013, Bondiombouy et al., 2015, Elmore et al., 2015], comme les modèles de bases de données :

- **NoSQL orienté colonnes**, pour traiter de gros volumes de données dont le flux de production est important ;
- **NoSQL orienté graphes**, dans le but d'exploiter les liens entre les données et décrire des réseaux complexes ;
- **NoSQL orienté documents**, dans le cadre de l'utilisation d'outils d'analyse exploitant le texte de messages ou de documents ;
- **Relationnelles**, pour des données structurées, c'est-à-dire des domaines pour lesquels une connaissance *a priori* permet de définir un modèle de données précis.

Afin de pouvoir exploiter les données, plusieurs outils d'analyse et de visualisation doivent être intégrés ou couplés avec la plateforme. Ils permettront aussi de supporter des analyses exploratoires rapides des données recueillies. Dans le but de pouvoir enchaîner et combiner les analyses, l'intégration d'un moteur de *worklflow* est également nécessaire.

La collecte de données pouvant s'étaler sur de longues durées, un système de reprise sur panne doit être intégré à la plateforme afin de prévenir une défaillance.

Un *proof of concept* de cette plateforme, appelé SNFreezer, a été développé et a permis de valider plusieurs des fonctionnalités au travers de différents projets. Nous avons validé la scalabilité de la plateforme au niveau de la collecte et du stockage des données. Cette plateforme a servi de point de départ pour une implémentation plus industrielle décrite dans le chapitre 6.

Pour des raisons de facilité d'accès aux données en grandes quantités, et de part le contexte des projets, nous nous sommes concentrés sur le réseau social Twitter. La plupart des profils sur Twitter étant publics, ce réseau social permet un accès assez aisé à une grande quantité de tweets via les différentes API qu'il propose. Les données issues des tweets sont riches pour ce qui est des relations (sociales) et des liens. En effet, Twitter offre différents opérateurs pour établir des liens entre les données grâce aux *hashtags*, aux *mentions* et aux *retweets*.

Dans ce chapitre, nous décrivons l'architecture de la plateforme SNFreezer, et nous détaillons les fonctionnalités principales que sont la collecte et le stockage polyglotte des données, aussi désigné sous le terme de polystore. Nous décrivons ensuite comment ces fonctionnalités ont été validées au travers de trois projets. Nous détaillons nos contributions pour la détection d'événements et l'identification des utilisateurs influents, puis nous abordons par des exemples d'utilisation les outils d'analyse avant d'établir un comparatif avec les plateformes concurrentes existantes et de faire une synthèse des leçons tirées de l'implémentation de cette plateforme.

5.2/ DESCRIPTION DE L'ARCHITECTURE

Nous avons commencé par analyser les différentes solutions existantes s'approchant le plus des fonctionnalités de la plateforme que nous souhaitions réaliser. Très peu de projets sous licence libre offraient ne serait-ce qu'une partie des fonctionnalités que nous avons retenues. Le projet le plus pertinent pour servir de base de travail était YourTwapperKeeper [1] (YTK).

YTK est un projet dont le but est de fournir un outil pour archiver, sur son propre serveur, des données issues de Twitter contenant des mots clés et hashtags définis par l'utilisateur. Il se présente sous la forme d'une application Web, développée en PHP. La base de données utilisée est MySQL. Il utilise les deux APIs [2] principales de Twitter, la *Search API* et la *Stream API*. La dernière mise à jour du code de YTK sur Github date du 24 mai 2013.

YTK étant un logiciel libre, nous l'avons réutilisé comme noyau de base pour notre plateforme SNFreezer. Après une période de tests et de revue de code, plusieurs ajouts, améliorations et correctifs ont été apportés au logiciel afin d'obtenir une plateforme robuste et scalable, disposant de toutes les fonctionnalités souhaitées.

Twitter ayant apporté des changements au niveau de leurs APIs depuis la dernière mise à jour du code de YTK, nous avons corrigé le code afin de se conformer aux dernières exigences de Twitter. Nous avons aussi amélioré le logiciel afin de lui permettre de stocker des données de plusieurs centaines de giga octets, ainsi que la possibilité de collecter les données dans toutes les langues. La plateforme permet aussi d'exploiter facilement les données via des outils d'analyse.

La figure 5.1 décrit l'architecture de la plateforme SNFreezer, dont le cœur est constitué de trois couches principales :

1. une couche pour la collecte de données contenant les connecteurs vers les API Twitter ;
2. une couche pour le stockage polyglotte utilisant une couche de drivers pour chaque système de stockage cible ;
3. une couche d'échange de données vers des outils tiers.

Autour de ce cœur, différents services exploitent principalement la couche assurant le stockage polyglotte : gestion du mode cluster et de la reprise sur panne (*failover*), moteur de gestion de flux (*workflow*) et gestion de la connaissance.

1. https://github.com/540co/yourTwapperKeeper
2. Application Programming Interface, Interface de Programmation Applicative, ensemble de classes, méthodes et fonctions mis à disposition des développeurs

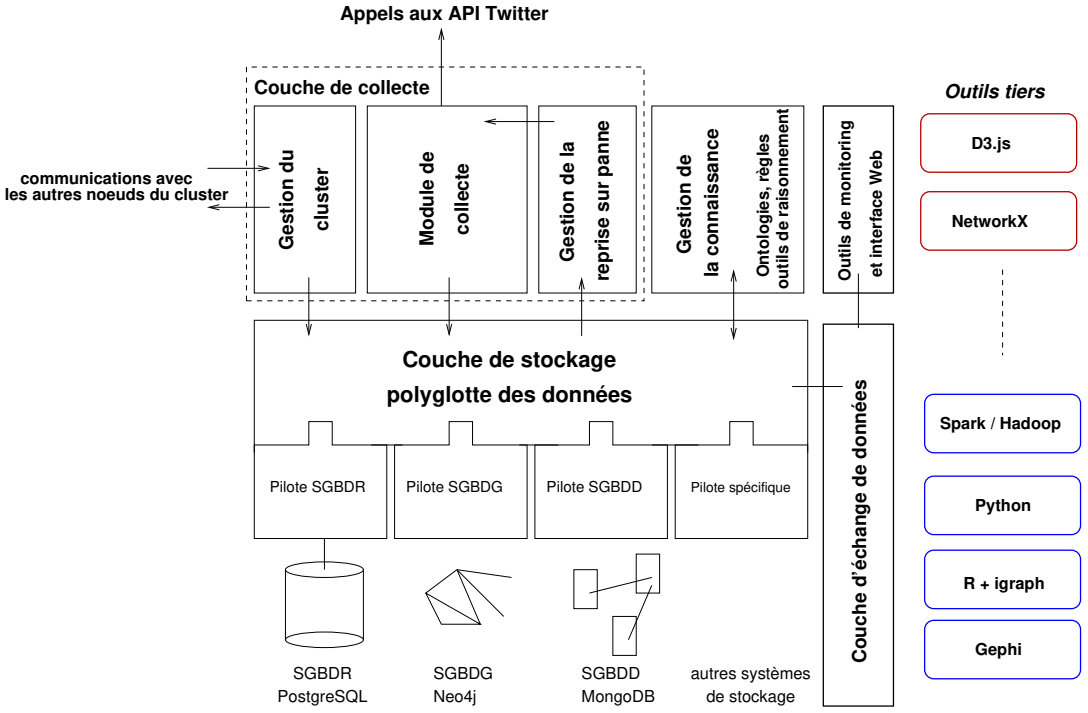

FIGURE 5.1 – Architecture générale de la plateforme SNFreezer

5.3/ COLLECTE DES DONNÉES

La première fonctionnalité développée pour SNFreezer a été celle concernant la collecte de tweets. Celle-ci fait appel à différentes API proposées par Twitter, qui induisent certaines limitations fonctionnelles. Cette fonctionnalité correspond à la partie ”Module de collecte” de la figure 5.1.

Les différents réseaux sociaux proposent des API pour accéder à leurs données, soit de manière libre avec des restrictions, soit avec des abonnements. Chaque réseau social ayant sa propre API spécifique, des connecteurs spécialisés doivent être développés pour collecter des données.

5.3.1/ TYPES D'APIS TWITTER

Depuis sa création Twitter a une politique d'accès aux données permettant au programmeur de développer des applications s'appuyant sur le réseau social. Ainsi, Twitter met à disposition des développeurs deux grandes familles d'APIs :

— les *REST API*[3], fournissant des outils programmatiques pour lire, rechercher et écrire des données sur Twitter ;
— les *Streaming API*[4], fournissant un accès direct au flux (*stream*) global de tweets.

Pour des raisons de simplicité et de compréhension, et compte tenu du contexte de notre utilisation (nous ne faisons que lire des données sur Twitter, nous n'en écrivons pas), nous utiliserons le terme de *Search API* pour désigner les *REST API*.

3. https://developer.twitter.com/en/docs/tweets/search/overview/basic-search
4. https://developer.twitter.com/en/docs/tweets/filter-realtime/overview.html

Les différentes APIs de Twitter utilisent toutes le format JSON[5] pour envoyer les tweets, ce qui permet de standardiser au maximum le traitement des données entrantes, indépendemment de l'API utilisée. Nous trouvons par exemple les champs contenant l'identifiant (*id*) de l'utilisateur qui envoie le tweet, le contenu du tweet ainsi que sa date et sa langue.

5.3.2/ UTILISATION DES APIS TWITTER

Nous utilisons la *Search API* pour récupérer :
- les tweets antérieurs au début de la collecte ;
- les tweets manqués en cas de panne au niveau de la *Streaming API*
- le fil d'actualité (*timeline*) des utilisateurs ;
- la liste des followers des utilisateurs.

Nous utilisons la *Streaming API* pour :
- les tweets arrivant en direct.

Le schéma 5.2 décrit le fonctionnement de la *Search API*. Elle repose sur un WebService REST[6]. Il nous faut d'abord définir un critère de sélection pour les tweets ①. Dans notre exemple, nous souhaitons avoir les tweets contenant le hashtag `#ep2014`[7], pour *European Parliament 2014*. L'envoi d'une requête *GET* ②, contenant l'adresse de la *Search API* ainsi que des paramètres, ici `%23ep2014` ainsi qu'un nombre de tweets, `count=100`, permet de demander à Twitter de nous envoyer l'ensemble des tweets ③, au format JSON ④, répondant aux critères passés en paramètres de la requête.

Le schéma 5.3 décrit le fonctionnement de la *Streaming API*. De la même manière que pour la *Search API*, nous définissons d'abord des critères pour la sélection des tweets ①, ici `#ep2014`, `Europarl_FR`[8] et `européennes`[9]. L'établissement d'une connexion permanente vers l'adresse de la *Streaming API* ②, et l'envoi d'une requête comprenant les paramètres correspondant aux critères de sélection permet de demander à Twitter de nous envoyer, au fur et à mesure de leur arrivée, les différents tweets répondant aux critères définis ③.

5.3.3/ LIMITATIONS DES APIS TWITTER

Les APIs fournies par Twitter sont soumises à des conditions d'utilisation et à des limitations qui ont un impact sur la collecte des données. Ces limitations sont susceptibles de changer. Pour pouvoir utiliser ces APIs, il est nécessaire de créer une application Twitter via une interface Web[10]. Twitter attribue alors différentes clés d'accès qui seront utilisées par les applications clients pour l'authentification lors des accès aux APIs. L'interface de gestion des clés permet aussi de choisir le type d'accès dont a besoin l'application.

Au niveau de la *Search API*, les données à disposition ne concernent que les 7 derniers jours. Différents types de requêtes sont possibles, dont les principaux types sont centrés

5. JavaScript Object Notation, format ouvert utilisé pour transmettre des données
6. Representational State Transfer, type d'API reposant sur le protocole HTTP et permettant d'accéder à une ressource, ici la *Search API*, puis d'effectuer diverses actions : lecture (*GET*), écriture (*POST*), modification (*PUT*), suppression (*DELETE*).
7. https://twitter.com/hashtag/ep2014
8. https://twitter.com/europarl_fr
9. https://twitter.com/search?q=europeeennes
10. https://apps.twitter.com/

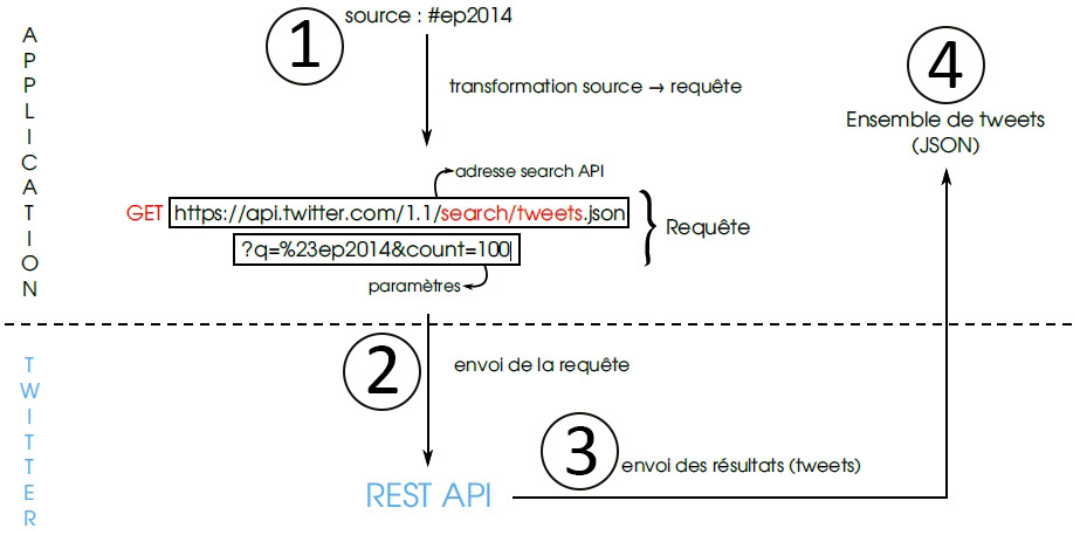

FIGURE 5.2 – Fonctionnement de la *Search API* de Twitter

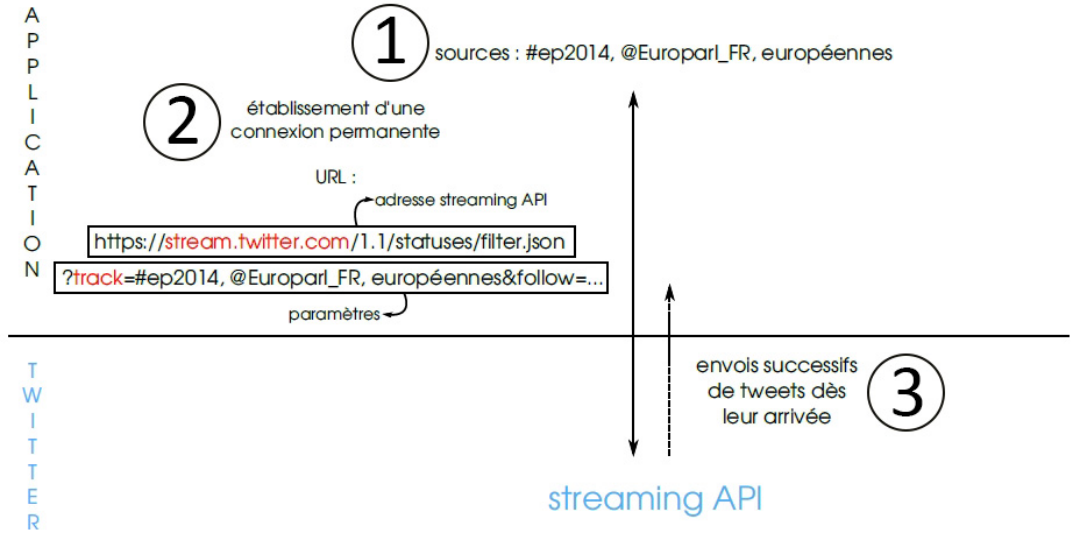

FIGURE 5.3 – Fonctionnement de la *Streaming API* de Twitter

sur : les hashtags, les comptes, les tweets. Concernant les hashtags, seuls les tweets indexés par Twitter sont disponibles. Cependant, même si Twitter ne communique pas sur les conditions d'indexation, J. Norblin [Norblin, 2014], en s'appuyant sur des collectes effectuées par les deux APIs, a estimé à 5% le nombre de tweets non indexés. Par ailleurs, concernant les requêtes centrées sur un utilisateur, 3200 tweets au maximum peuvent être récupérés dans sa timeline. Enfin, avec la *Search API*, il n'est pas possible d'envoyer plus de 450 requêtes en moins de 15 minutes.

Au niveau de la *Streaming API*, une application peut récupérer des tweets correspondant à une des conditions (un mot, une phrase, un hashtag, ou un compte d'utilisateur), que l'on nommera *query source*. Le nombre de *query sources* ne peut pas excéder 400. De plus, la *Streaming API* ne peut pas collecter plus de 1% du trafic global. Cette limitation peut être levée en développant un partenariat commercial avec Twitter.

Chaque compte Twitter peut définir plusieurs applications. Cependant, pour chaque adresse IP hébergeant une application, les précédentes limitations s'appliquent. Ainsi, il n'est pas possible de recueillir des données portant sur plus de 400 conditions avec une seule application hébergée par une machine. Il convient alors d'avoir recours à plusieurs comptes, utilisant la même application, mais avec des *tokens* d'identification différents et communiquant avec des adresses IP différentes.

La collecte des données se fait grâce au module de collecte de données de la plateforme SNFreezer, connecté aux *Streaming* et *Search APIs* de Twitter, via deux processus principaux. Deux autres processus sont utilisés pour collecter des informations complémentaires sur les utilisateurs de Twitter à des fréquences déterminées, et lors du lancement d'une collecte à récupérer tous les tweets auxquels on a accès sur la timeline des utilisateurs.

5.4/ MODE CLUSTER ET MÉCANISME DE REPRISE SUR PANNE

Les modes *cluster* et *reprise sur panne* ont été développés pour contourner certaines limitations des *Streaming* et *Search* APIs de Twitter expliquées dans la partie 5.3.3. Ils correspondent aux parties "Gestion du cluster" et "Gestion de la reprise sur panne" de la figure 5.1.

En particulier, la limitation la plus contraignante est la limite de 400 *query sources* (conditions) par adresse IP et porte donc sur la *Streaming API*. Certains projets nécessitant plus de 400 *query sources*, nous avons donc développé un module de gestion de cluster. Dans le mode cluster, plusieurs machines virtuelles, dans lesquelles des instances des processus de collecte de données sont déployées, peuvent être utilisées dans un projet pour collecter des tweets en utilisant des conditions sur un ensemble de plusieurs milliers de *query sources*. Ces dernières peuvent être définies sur chaque machine virtuelle en utilisant une interface Web ou peuvent être déployées automatiquement en utilisant des scripts et des fichiers d'importation utilisés à partir d'une des machines virtuelles qui servira de contrôleur central. Ce mode cluster part du principe qu'il y aura un gros volume de tweets à traiter. La couche de stockage peut alors utiliser un schéma non normalisé dans PostgreSQL, ou stocker les données dans une base MongoDB, ou encore écrire dans des fichiers au format JSON (exploitant le système de fichiers Hadoop HDFS) depuis le contrôleur central.

Dans le cas d'une interruption durant la collecte de données, par exemple des problèmes de réseau affectant la disponibilité des bases de données ou les appels vers les APIs Twitter, le processus en charge de la *Search API* peut récupérer des tweets sur la période des 7 derniers jours. Des emails de notification sont automatiquement envoyés à l'administrateur avec une caractérisation de la panne (stockage, API, etc.). Lorsque le processus de collecte redémarre, une comparaison est faite avec l'identifiant du dernier tweet collecté. Le processus en charge de la *Search API* commence alors à collecter les tweets manquants, par exemple en utilisant les *timelines* des utilisateurs [11]. Dans le cas d'une panne au niveau de la base de données PostgreSQL globale, chaque processus de collecte utilise une base MySQL locale disponible sur chaque machine virtuelle afin d'y ajouter les nouveaux tweets dans une table de cache. Quand la base de données PostgreSQL globale est à nouveau accessible, les tables de cache sont envoyées au serveur Post-

11. https://developer.twitter.com/en/docs/tweets/timelines/overview

greSQL.

Pour suivre l'état d'une collecte et le statut des différentes machines virtuelles instanciées sur le cluster, plusieurs petits outils ont également été développés. Ceux-ci permettent, au moyen d'une interface Web, de connaître le nombre de tweets, les hashtags les plus utilisés ou les utilisateurs les plus actifs sur une période, et d'envoyer des emails d'alerte aux administrateurs et chercheurs si besoin. Ils permettent aussi d'affiner la liste des critères de collecte. Cependant, dans le cas de gros volumes de données, la base de données globale doit être répliquée pour permettre le calcul des indicateurs des outils de suivi.

5.5/ STOCKAGE POLYGLOTTE

Pour traiter le problème de stockage des tweets, que ce soit en matière de performance, d'interopérabilité, c'est-à-dire de connectivité avec des outils tiers, mais aussi pour favoriser l'adéquation entre les algorithmes d'analyse et les structures de données, nous avons spécifié et développé une couche de stockage. Celle-ci correspond à la partie "Couche de stockage polyglotte des données" de la figure 5.1. Cette couche utilise plusieurs systèmes de stockage et comprend un système de gestion de bases de données relationnelles (SGBDR), de bases de données graphe (SGBDG), de bases de données orientée documents (SGBDD) ainsi qu'un système de stockage NoSQL sur Hadoop, permettant le passage à l'échelle. Les différents systèmes de stockage peuvent être utilisés simultanément. Ainsi, en fonction des types d'analyses prévus, plusieurs systèmes de stockage utilisant différents modèles et schémas seront utilisés.

Il est possible de sélectionner le ou les systèmes de stockage appropriés via un fichier de configuration. Cela permet de mettre en place différentes analyses en temps réel, c'est-à-dire pendant la collecte, sans avoir recours à des ETL [12] ou à se limiter à quelques algorithmes. Ainsi, il est possible d'effectuer des calculs de centralité, de trouver des communautés via des algorithmes utilisant une marche aléatoire, ou d'analyser l'opinion du contenu de tweets. En effet, la duplication des différents systèmes de stockage permet de ne pas lancer des scripts de transformation de modèle après la phase de collecte et permet de réduire le temps d'attente pour avoir accès aux jeux de données une fois la collecte terminée.

Cependant, la duplication des données ne peut pas être utilisée dans le cadre de la collecte de très grands volumes de données avec des fréquences de publication de tweets très élevées, à cause du goulet d'étranglement introduit par la transformation du JSON dans les trois modèles de bases de données à disposition.

La couche de stockage fournit également aux développeurs des interfaces pour ajouter des pilotes spécifiques à d'autres systèmes de stockage (figure 5.1).

Selon les besoins et le type d'information à analyser, il est donc possible de choisir entre plusieurs structures de stockage :

— Dans le cas d'information structurée, une base de données relationnelle est à disposition (figure 5.4).
— Dans cas de l'analyse de données liées (par exemple des structures de données

12. Extract-transform-load, middleware permettant d'effectuer des synchronisations d'information d'une source de données vers une autre.

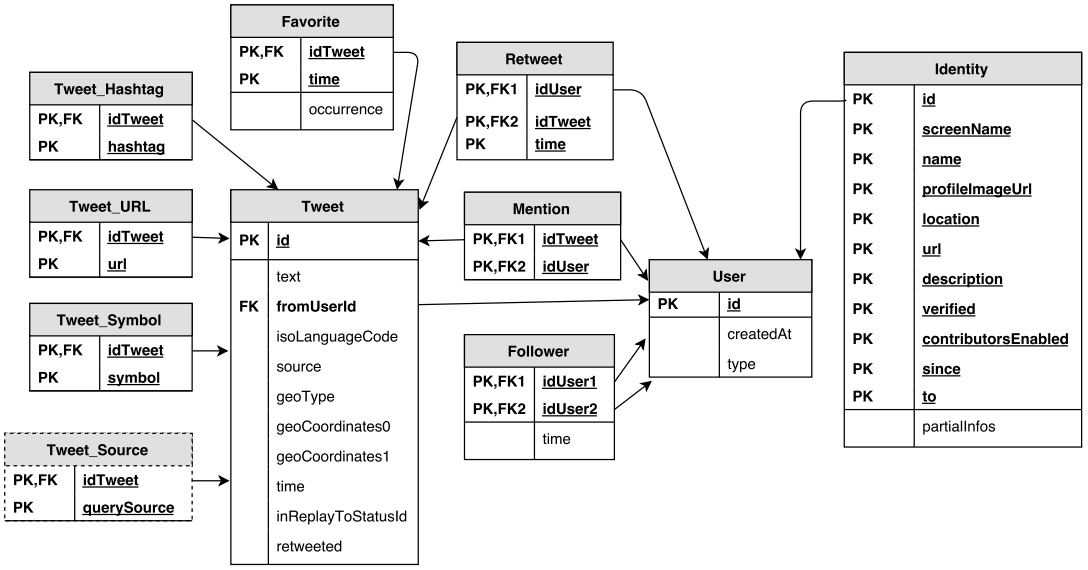

FIGURE 5.4 – Modèle logique de données relationnelles pour les tweets

comme celles des réseaux sociaux), une base de données graphe peut être plus appropriée (figure 5.5). Les objets (Tweet, User, Hashtag, etc.) sont des nœuds, et les relations sont décrites par des arêtes avec des propriétés (send, contain, mention, etc.). L'avantage du modèle graphe réside dans son efficacité à déduire de nouveaux liens de façon plus efficace que les jointures dans le modèle relationnel. C'est par exemple le cas des relations *use*, représentées par des flèches en pointillés, qui sont une composition de liens et qui apportent de l'information supplémentaire pour l'analyse des données. Par exemple, un utilisateur qui retweete un tweet contenant un hashtag sera considéré comme ayant utilisé cet hashtag. Ce schéma a été implémenté dans Neo4j [13].

— Dans le cas de gros volumes de données, trois choix sont possibles :
  — un schéma de base de données non normalisé, avec une table pour les tweets et quelques autres tables pour les followers. Dans ce cas, les contraintes d'intégrité des données ne sont pas vérifiées ;
  — des fichiers JSON stockés directement sur le serveur ou dans un système de fichier Hadoop (HFS) ;
  — une base MongoDB [14] accessible à distance.

Il est possible de cumuler les systèmes de stockage (par exemple des fichiers JSON avec une base de données relationnelle avec un schéma normalisé), néanmoins nous proposons aussi un ensemble d'outils implémentant les transformations de modèles de données pour transformer de manière asynchrone les données d'un système de stockage à un autre et pour vérifier les contraintes d'intégrité, ainsi que l'intégrité des données.

Une couche spécifique, nommée "Couche d'échange de données" sur la figure 5.1, permet les échanges de données entre les systèmes de stockage et les services dédiés aux applications, ainsi qu'avec les outils tiers qui sont connectés en fonction des analyses souhaitées, visibles sur la partie droite de la figure 5.1. Des connecteurs ont été

13. https://neo4j.com/
14. https://www.mongodb.com/

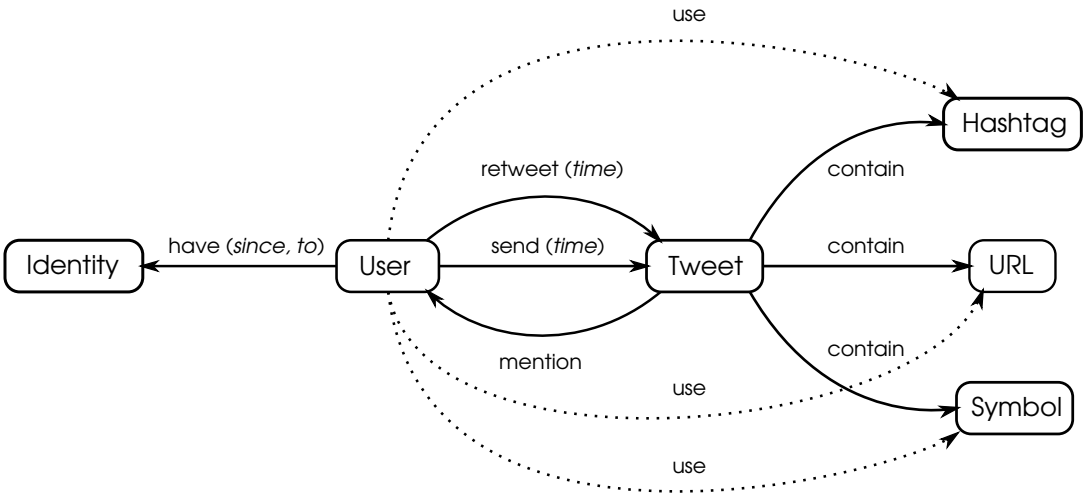

FIGURE 5.5 – Extrait d'un schéma de données sous forme de graphe pour les tweets

développés pour analyser les données avec des logiciels tiers tels que *R* et *igraph*[15], en particulier avec PostgreSQL et Neo4j. De plus, nous avons développé des connecteurs pour afficher avec *D3.js*[16] des données issues de PosgreSQL et Neo4j. Si un connecteur n'est pas disponible, cette couche fournit des fichiers d'export pour les différentes classes d'algorithmes tels que des fichiers contenant des matrices d'adjacence, des triplestore pour les graphes, des tableaux multidimensionnels ou des fichiers CSV (Comma-Separated Values) pour les tableurs.

5.6/ VALIDATION DE LA COLLECTE ET DU STOCKAGE SUR SNFREEZER

SNFreezer a été utilisé dans trois projets, les deux premiers pour valider le stockage, le mode cluster et la reprise sur panne. Le troisième, plus général, est essentiellement centré sur des analyses utilisant différentes modélisations des données. La validation des analyses sera étudiée plus en détail dans la partie 5.7.

5.6.1/ DESCRIPTION DES PROJETS

5.6.1.1/ TWITTER ET LES ÉLECTIONS EUROPÉENNES DE 2014

Le premier projet, *TEE 2014* (pour Twitter et les Élections Européennes de 2014), est une étude comparative internationale de l'utilisation de Twitter par les candidats aux élections européennes de mai 2014. Il a pour but d'étudier les tweets politiques, à travers six pays, pendant les élections européennes de 2014. Ce projet se concentre sur les tweets émis par tous les candidats, en Allemagne, Belgique, Espagne, France, Italie et au Royaume-Uni, ainsi que sur les messages qui leur sont adressés publiquement. Les tweets contenant les principaux hashtags associés avec les élections dans chaque

15. http://igraph.org/
16. http://d3js.org/

pays étudié sont également collectés. Ce projet est soutenu par le Ministère français des Affaires étrangères [17] et la Fondation allemande pour la recherche [18], au travers du programme de Partenariat Hubert Curien Procope [19], ainsi que par le Réseau national des Maisons des Sciences de l'Homme [20]. Ce premier projet a été utilisé pour valider le mode cluster, le passage à l'échelle du nombre de critères pour la collecte de données, le mécanisme de reprise sur panne, ainsi que le stockage dans une base relationnelle (normalisée ou non) et dans une base graphe Neo4j.

Les différentes *query sources* ont été spécifiées par les équipes du projet (6 équipes représentant 6 pays) et incluent :

— les comptes Twitter des candidats sur les listes de chaque pays. En fonction des pays, les comptes suivis constituent l'intégralité de la liste, ou alors seulement les n premiers candidats ;
— des comptes de média traditionnels (le journal *Le Monde* par exemple) ;
— des hashtags et mots clés reflétant les grandes thématiques de campagne.

5.6.1.2/ Coupe Du Monde de football de 2014

Le second projet concerne la Coupe du monde de football de 2014 (*CDM2014*), ayant eu lieu du 12 juin au 13 juillet 2014. Ce projet a été utilisé pour valider le passage à l'échelle du stockage d'une très grande quantité de tweets en mode cluster, ainsi que la reprise sur panne. En effet, la popularité très importante de cet événement, ainsi que son aspect mondial, ont généré un flux de données conséquent, qui a excédé la limite de 1% du trafic global imposée par la *Streaming API*.

Les différentes *query sources* représentent plus de 1600 critères :

— des comptes de différentes fédérations ;
— les comptes officiels des différents équipes et les hashtags désignant les équipes ;
— les noms des joueurs ou leurs comptes officiels ;
— les hashtags désignant les pays ;
— des hashtags des matchs, par exemple #FRAGER pour France - Allemagne. Certains hashtags ont été rajoutés pendant la compétition, principalement ceux correspondant aux matchs à élimination directe, dont les affiches ne sont connues qu'après le premier tour.

5.6.1.3/ Co-voiturage

Le troisième projet a pour but d'étudier le covoiturage et le comportement des utilisateurs de Twitter sur ce sujet. Il a été utilisé pour étudier des événements et des interactions entre utilisateurs, et détecter des communautés d'utilisateurs. Ce projet a eu lieu dans le cadre du développement de la plateforme DisCoCRM, issue de collaborations entre l'entreprise eb-Lab et le laboratoire Le2i.

Les différentes *query sources* incluent :

17. http://www.diplomatie.gouv.fr/fr/
18. http://www.dfg.de/en/index.jsp
19. http://www.campusfrance.org/fr/procope
20. http://www.msh-reseau.fr/

— des comptes des principales applications de co-voiturage (BlaBlaCar, idVROOM, Uber, etc.) ainsi que leurs comptes locaux s'ils existent (français, anglais, allemand, etc. ou par ville) ;
— les comptes des fondateurs des entreprises (par exemple Frédéric Mazzella pour BlaBlaCar) ;
— des hashtags et mots clés liés aux comptes (par exemple #BlaBlaCar).

5.6.2/ TEST DU MODE CLUSTER POUR LE PASSAGE À L'ÉCHELLE DES CRITÈRES DE COLLECTE

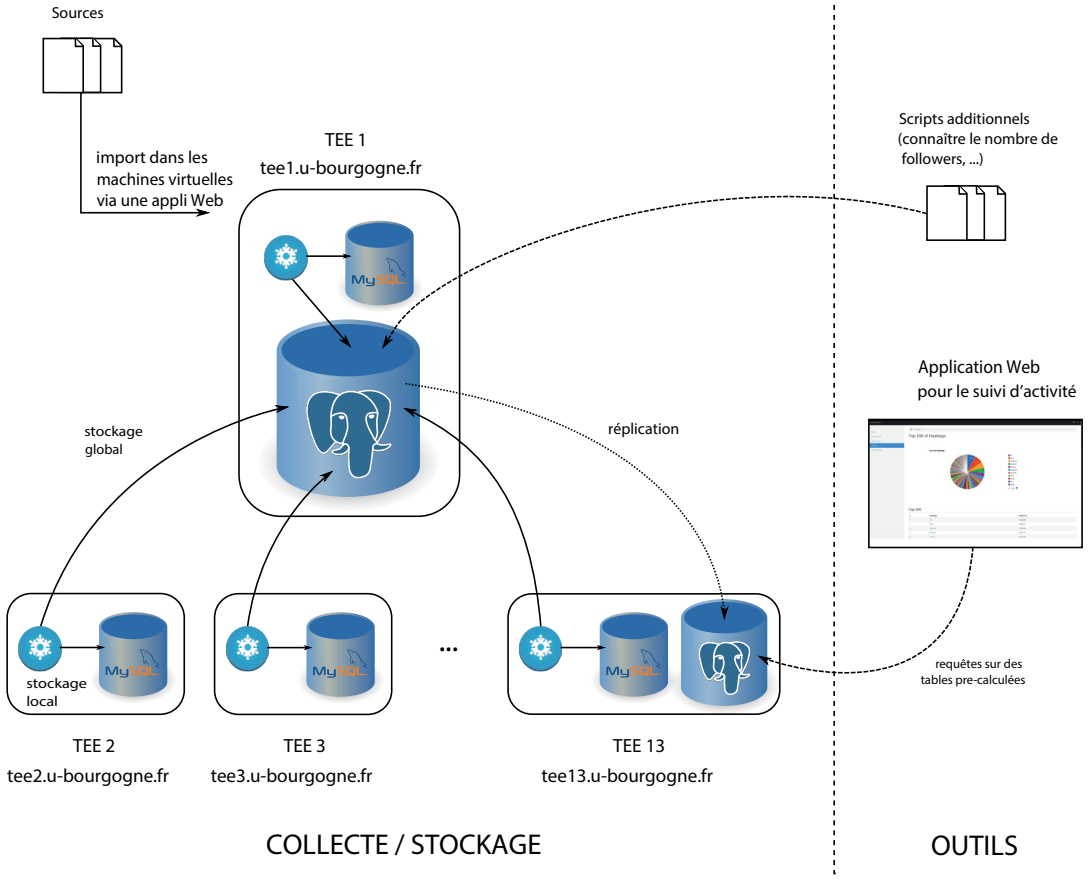

FIGURE 5.6 – Architecture utilisée pour le mode cluster dans le cadre du projet TEE 2014

Dans le cadre de la collecte des tweets du premier projet, *TEE 2014*, treize machines virtuelles ont été utilisées pour prendre en charge le nombre important de *query sources* (5 000 critères ont été définis) et les limitations de la *Streaming API*, comme le montre la figure 5.6.

Les différentes sources (comptes Twitter, mots clés et hashtags) sont importées dans la plateforme via une application Web. Les sources sont ensuite réparties entre les différentes machines virtuelles (*VM*) (`TEE 1` à `TEE 13` à raison de 400 sources par *VM*. Chaque *VM* possède sa propre instance de *SNFrezzer*, avec son propre stockage local. La machine `TEE 1` est aussi utilisée pour le stockage global des données, via une base `PostgreSQL`. La machine `TEE 13` contient elle aussi une base `PostgreSQL`, répliquant celle de la machine `TEE 1`. `TEE 13` est utilisée pour les tables pré-calculées servant à

l'application Web. Ces tables permettent d'effectuer le suivi de la collecte, et fournissent différents indicateurs, par exemple la fréquence des tweets par heure, la fréquence des hashtags et les comptes les plus cités.

La période de collecte s'est étalée sur un mois et 50 millions de tweets ont été récupérés. La taille de la base de données est d'environ 50 Go, que ce soit pour celle utilisant un schéma relationnel non normalisé, ou pour celle avec un schéma relationnel normalisé et des index. La couche de stockage a prouvé son bon fonctionnement en mode cluster. Nous avons aussi vérifié le bon fonctionnement de l'ajout dynamique de *query sources* au cours de la collecte, ainsi que de la reprise sur panne. Des tweets multilingues, incluant l'alphabet cyrillique, ont été récupérés afin d'étudier les liens entre les tweets politiques au sein de l'Union Européenne et en Ukraine[21]. D'après l'équipe technique des chercheurs en sciences de la communication de l'Université de Bonn, qui avait déjà réalisé des collectes avec YTK, le temps d'attente entre la collecte et le début des analyses sur les tweets a été significativement réduit (de quelques semaines à moins d'une semaine) grâce à l'utilisation de SNFreezer.

5.6.3/ PASSAGE À L'ÉCHELLE DU STOCKAGE ET REPRISE SUR PANNE

Le mode cluster combiné avec les modes de stockage des données proposés, c'est-à-dire dans une base NoSQL MongoDB et/ou dans des fichiers JSON, ont été validés par le deuxième projet *CDM2014*.

En moyenne, 1 million et demi de tweets ont été collectés par jour, avec des pics à la fin de l'événement ainsi que pour certains matchs en particulier. Au total, plus de 1,1 milliard de tweets ont été stockés, à la fois dans une base MongoDB et dans des fichiers JSON, ce qui représente une taille de 3.2 To de données JSON.

Par deux fois la limite de 1% du trafic global a été dépassée et le système de reprise sur panne s'est déclenché avec succès pour collecter les tweets manquants. En effet, le processus connecté à la *Search API* a collecté les anciens tweets manqués pendant que le processus connecté à la *Stream API* continuait la collecte des nouveau tweets.

De plus, le mécanisme de reprise sur panne a été activé avec succès à plusieurs reprises pour permettre des opérations de mise à jour du code de SNFreezer, des mises à jour système nécessitant un redémarrage des serveurs, des sauvegardes de bases de données et des changements de paramètres du noyau de PostgreSQL. Pour ces quatre cas, le mécanisme de reprise sur panne a utilisé le processus connecté à la *Search API* pour collecter les tweets manqués durant l'arrêt des machines ou des services.

Dans cette section, nous avons mis l'accent sur les propriétés non fonctionnelles de la plateforme. Dans la section suivante, nous nous intéressons aux aspects liés à l'analyse des données collectées et nous montrons comment mettre en œuvre une approche multi-paradigme itérative et incrémentale, c'est-à-dire une approche qui permet, au moyen de différents algorithmes, de produire par étapes de la connaissance en affinant les données.

21. Des élections présidentielles ont eu lieu en Ukraine le 25 mai 2014, après une période de troubles importants dans le pays.

5.7/ CONTRIBUTIONS AUX OUTILS D'ANALYSE DE SNFREEZER

Dans cette partie nous discutons pour chaque projet des résultats obtenus en appliquant plusieurs types d'algorithmes pour extraire différentes propriétés, comme des régularités ou singularités à partir d'un jeu de données.

Le réseau produit par les interactions des utilisateurs de Twitter est très grand et très dynamique. Il est basé sur l'instantanéité, ce qui rend compliqué le fait de vouloir l'étudier directement de manière brute et globale. Pour contourner la complexité des données à étudier, nous proposons une approche itérative qui consiste à filtrer de différentes manières les données brutes du graphe de Twitter pour les adapter au contexte de l'étude.

Dans cette approche, notre contribution permet :

- de réaliser des explorations interactives du jeu de données des tweets en détectant des intervalles de temps intéressants (qui peuvent correspondre à des événements réels) ;
- puis de caractériser ces intervalles en fonction des hashtags ou des communautés ;
- de rechercher des utilisateurs influents en fonction de leurs relations topologiques dans le graphe.

Pour cela, nous utilisons la couche d'échange de données de la plateforme afin de traiter les données en enchaînant plusieurs méthodes d'analyse. Il n'y a pas de limite au niveau des algorithmes utilisables, d'autres que ceux déjà intégrés pouvant être ajoutés sous forme de plug-ins à notre système.

Dans la suite de cette section, nous allons présenter les algorithmes existants, leurs limites, les adaptations que nous proposons et différentes analyses réalisées, en commençant par la détection d'événements, puis la détection d'utilisateurs influents.

5.7.1/ DÉTECTION EXPLORATOIRE D'ÉVÉNEMENTS

La détection d'événements ou d'anomalies ayant eu lieu pendant la période couverte par la collecte est intéressante sur plusieurs plans. Tout d'abord, elle permet de faire ressortir les différents événements importants ayant eu lieu. Ensuite, elle permet de cibler un ou plusieurs intervalles de temps à étudier, ce qui réduit la quantité de données à analyser. Ceci est particulièrement utile si la quantité de données brutes est importante, permettant ainsi de réduire les temps de traitement.

Pour cela, nous nous intéressons au nombre de tweets postés, et nous souhaitons trouver les points où une variation importante dans le nombre de tweets postés a lieu. Cela correspond à un changement de comportement des utilisateurs, et est probablement lié à un événement particulier.

Nous prenons l'horodatage (`timestamp`) du premier et du dernier tweet de la collecte. Entre ces deux `timestamp`, nous découpons la série temporelle en différents intervalles de longueurs identiques, par exemple par jour, par demi-journée, etc. Un vecteur est ensuite construit, composé d'une part des différents intervalles de temps, et d'autre part, pour chaque intervalle, du nombre de tweets postés. Nous détectons ensuite les changements à partir de ce vecteur.

	Figure 5.7	Figure 5.8	Figure 5.9
min.size	2	7	14
method	multi	multi	multi
beta	0.001	0.001	0.001
degree	0	2	0

TABLE 5.1 – Paramètres breakout

Les données peuvent être brutes, c'est-à-dire sans filtrage, ou filtrées en ne gardant que les tweets provenant de certaines *query sources*. Nous présentons dans les paragraphes suivants une série d'expériences ayant servi à construire un outil simple et efficace pour la détection et la caractérisation d'événements.

5.7.1.1/ ANALYSES AVEC L'ALGORITHME BREAKOUT

L'algorithme proposé et utilisé dans [James et al., 2014a] détecte des divergences dans la moyenne d'une distribution. Nous avons utilisé le package *R* `BreakoutDetection`, développé par Twitter[22]. L'intervalle de temps choisi est de 6 heures entre chaque `timestamp`, et le projet concerné est celui sur le co-voiturage. Nous avons filtré les données en ne gardant que celles concernant *Uber*.

Cet algorithme permet de définir les paramètres suivants :

- le nombre minimal d'observations entre chaque changement (`min.size`) et entre la détection de plusieurs points de changements (`method=multi`), ou d'un seul point de changement `method=amoc`[23] ;
- si une analyse pour des changements multiples a été retenue, des paramètres complémentaires peuvent être précisés. Ces derniers influent sur la fonction de pénalisation utilisée dans l'algorithme :
  - le degré de la fonction polynomiale de pénalisation, qui peut prendre les valeurs 0, 1 ou 2. Par défaut, la valeur est `1` ;
  - un nombre réel constant, `beta`, utilisé pour mieux contrôler le coût de la fonction de pénalisation. La valeur par défaut est `0.008` ;
  - un nombre réel constant, `percent`, qui spécifie le pourcentage minimum de changement dans la qualité de l'ajustement statistique pour ajouter un point de changement. Une valeur de `0.25` correspond à une augmentation de 25%. Il n'y a pas de valeur par défaut.
  - Si ni `beta` ni `percent` sont définis, la forme de pénalisation par défaut est appliquée avec `beta=0.008`.

Nous avons réalisé plusieurs expériences à partir des tweets collectés, en faisant varier les différents paramètres de l'algorithme. Nous avons choisi la méthode `multi` à chaque fois, afin de détecter plusieurs événements. Le tableau 5.1 présente les différents paramètres pour les figures 5.7, 5.8 et 5.9.

On remarque que les résultats sont très variables en fonction des paramètres choisis. Le paramètre le plus important est `min.size`. S'il est trop faible (figure 5.7), beaucoup de changements seront détectés ; s'il est trop élevé (figures 5.8 et 5.9), trop peu de chan-

22. https://blog.twitter.com/engineering/en_us/a/2014/breakout-detection-in-the-wild.html
23. At Most One Change (Au Maximum Un Changement)

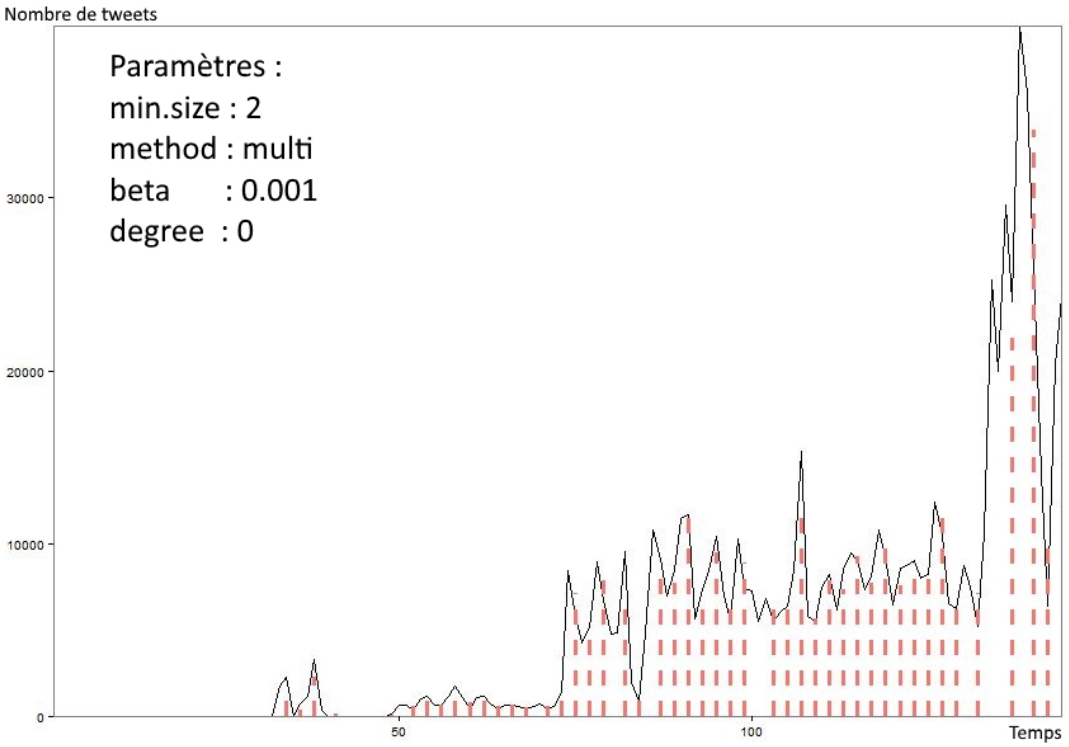

FIGURE 5.7 – Projet covoiturage - Uber - Algorithme `Breakout`

gements seront détectés. Trouver la bonne valeur dépend donc de la distribution des données étudiées.

5.7.1.2/ ANALYSES AVEC L'ALGORITHME PELT

Nous avons ensuite utilisé l'algorithme `PELT` [Killick et al., 2012b], qui intègre une fonction de vraisemblance paramétrée. L'implémentation de PELT est proposée via le package *R* `changepoint`[24]. L'intervalle de temps choisi est toujours de 6 heures entre chaque `timestamp`, et le projet concerné est celui sur le co-voiturage. Nous avons filtré les données en ne gardant que celles concernant *Uber*.

Nous avons réalisé trois expériences avec l'algorithme `PELT` à partir des tweets collectés : une première en utilisant la moyenne du nombre de tweets collectés par timestamp, une deuxième avec la variance et une dernière en utilisant les deux. Ces expériences sont décrites respectivement dans les figures 5.10, 5.11 et 5.12.

On remarque ici aussi beaucoup de différences entre les trois figures. Dans notre exemple, lorsque l'on se base uniquement sur la moyenne (figure 5.10), le nombre de changements détectés est très important et ces derniers ne correspondent pas à des événements réels. En effet, sur certaines périodes, l'algorithme détecte un changement à chaque `timestamp`, ici toutes les 6 heures, ce qui semble peu réaliste. Lorsque l'on utilise uniquement la variance (figure 5.11), trop peu d'événements sont détectés. Lorsque l'on utilise à la fois la moyenne et la variance (figure 5.12), les changements détectés semblent correspondre à des périodes d'activités distinctes.

24. https://cran.r-project.org/web/packages/changepoint/changepoint.pdf

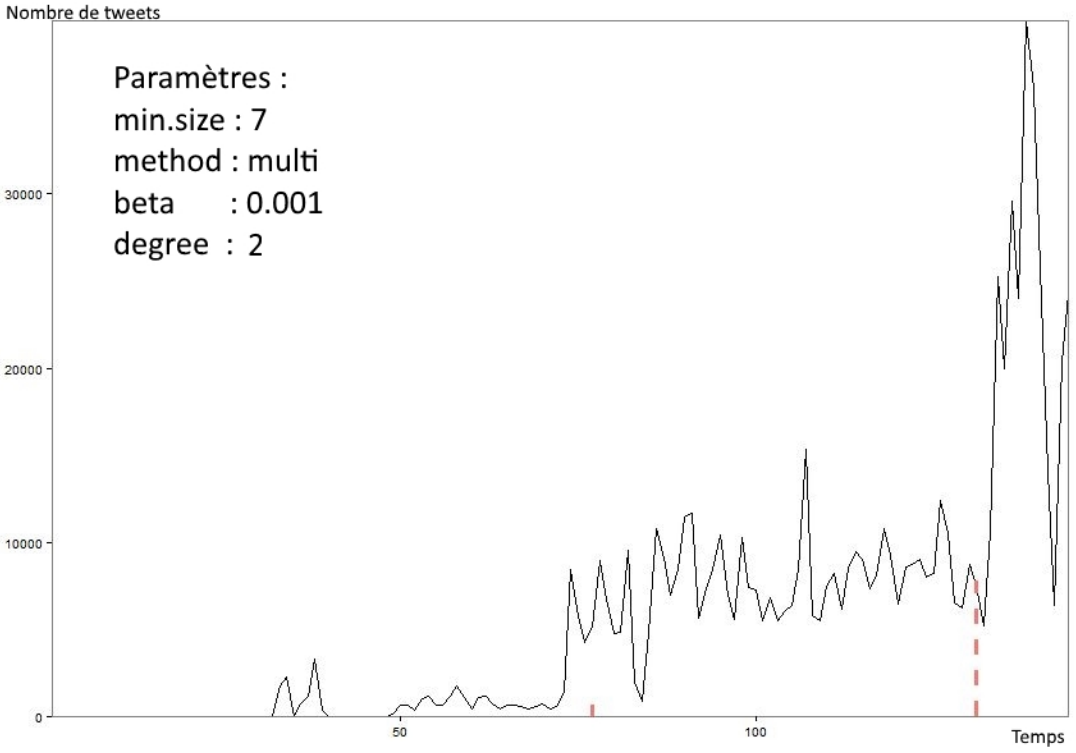

FIGURE 5.8 – Projet covoiturage - Uber - Algorithme `Breakout`

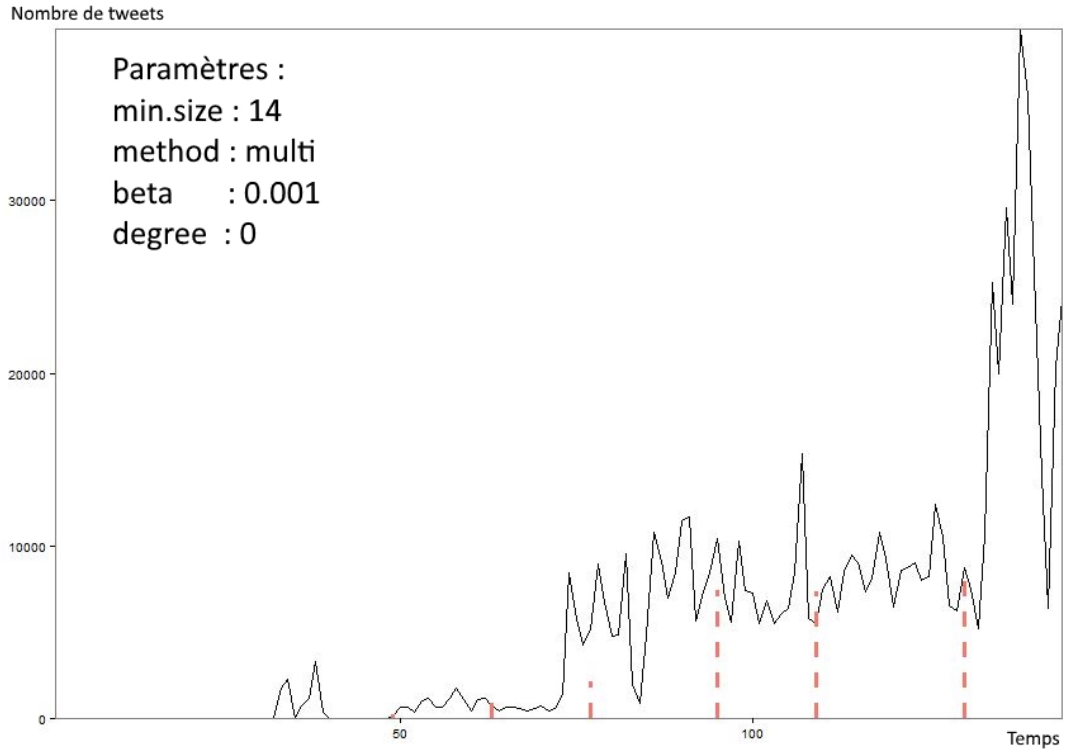

FIGURE 5.9 – Projet covoiturage - Uber - Algorithme `Breakout`

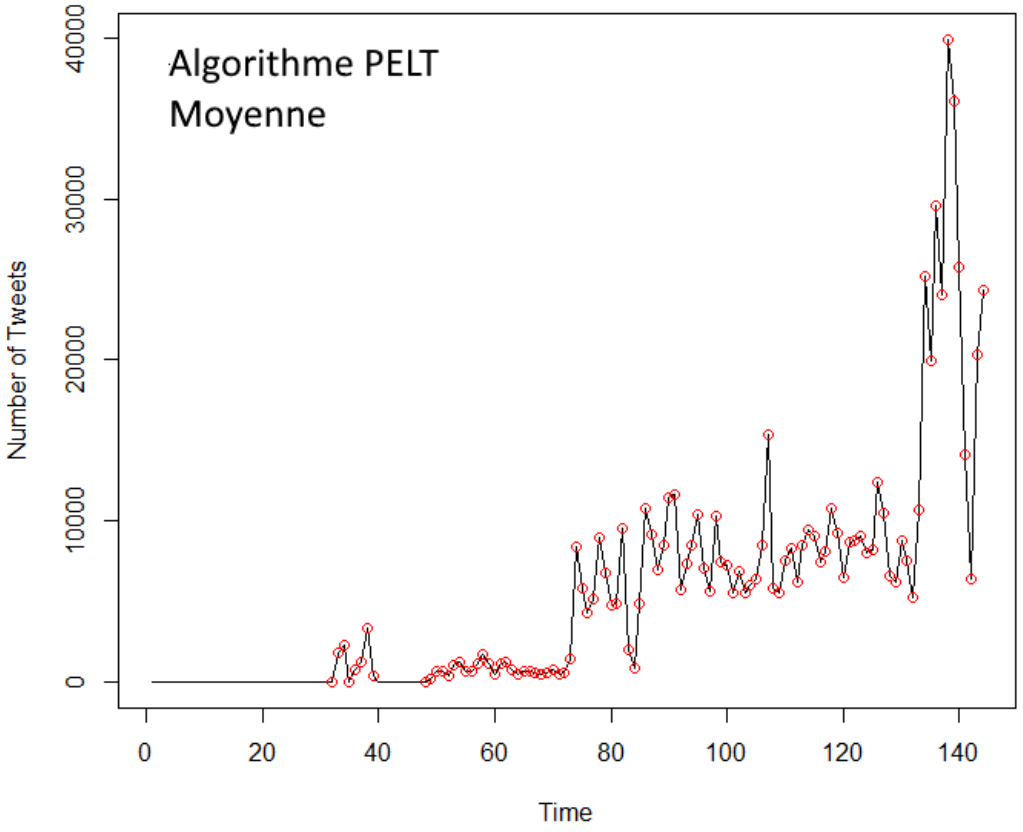

FIGURE 5.10 – Projet covoiturage - Uber - Algorithme PELT - Moyenne

La plupart des algorithmes de détection d'événements utilisent soit une analyse statistique d'une série temporelle [James et al., 2014b, Bai et al., 2003, Killick et al., 2012a], soit des techniques basées sur le traitement automatique du langage naturel [Atefeh et al., 2013, Guille et al., 2014].

Cependant, avant d'utiliser un algorithme basé sur une approche statistique, il est nécessaire de formuler des hypothèses sur la distribution a priori des données. Dans beaucoup de cas, il est difficile de connaître parfaitement cette distribution et de justifier les choix retenus. Cela nécessite beaucoup de connaissances sur la distribution à étudier, comme nous l'avons vu dans les parties 5.7.1.1 et 5.7.1.2, et n'est donc pas adapté pour des non experts.

5.7.1.3/ ANALYSES AVEC LA DENSITÉ TEMPORELLE

Pour palier ce problème et fournir un outil d'identification d'anomalies dans les séries temporelles pour des non experts, nous proposons une méthode basée sur une estimation de la densité par fonction noyau [Rosenblatt et al., 1956, Parzen, 1962].

La méthode proposée permet de trouver un minima, ou un maxima local dans la densité

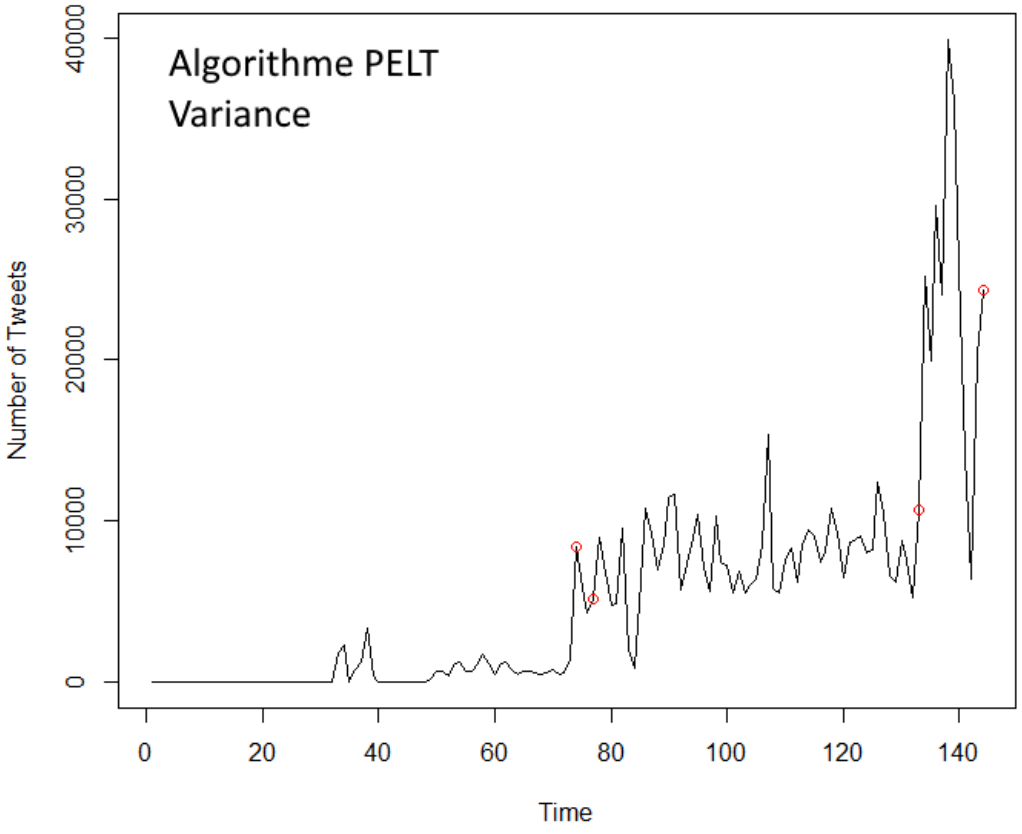

FIGURE 5.11 – Projet covoiturage - Uber - Algorithme PELT - Variance

temporelle des tweets. Elle permet de déterminer les moments après lesquels le nombre de tweets publiés commence à augmenter, ou à diminuer. L'avantage de la méthode est qu'elle ne possède qu'un seul paramètre facilement compréhensible, la bande passante, qui permet de changer de mode de détection des événements :

— quand la bande passante est faible, des micro événements sont détectés (des événements ayant eu lieu sur une courte période de temps) ;
— quand elle est élevée, seuls les macro événements (tendances) sont détectés (ceux ayant eu lieu sur une longue période de temps).

Le code source du programme de détection de minima et maxima locaux de densité temporelle est disponible sur github [25].

Cependant, la seule utilisation de cette méthode ne permet pas d'obtenir des descriptions de l'événement détecté, à part ses dates de début et de fin. Il n'y a aucune information disponible sur le contenu de l'événement, c'est-à-dire sur ses causes et ses conséquences. Une décomposition par hashtag des données permet d'aider à formuler des hypothèses sur le contenu d'un événement. La densité temporelle de chaque hashtag est calculée. La figure 5.13 montre qu'en regroupant toutes les densités calculées, il est possible de les comparer facilement et visuellement, et ainsi fournir un outil de visualisation simple

25. https://github.com/kerzol/Changepoint-pelt-vs-timedensity

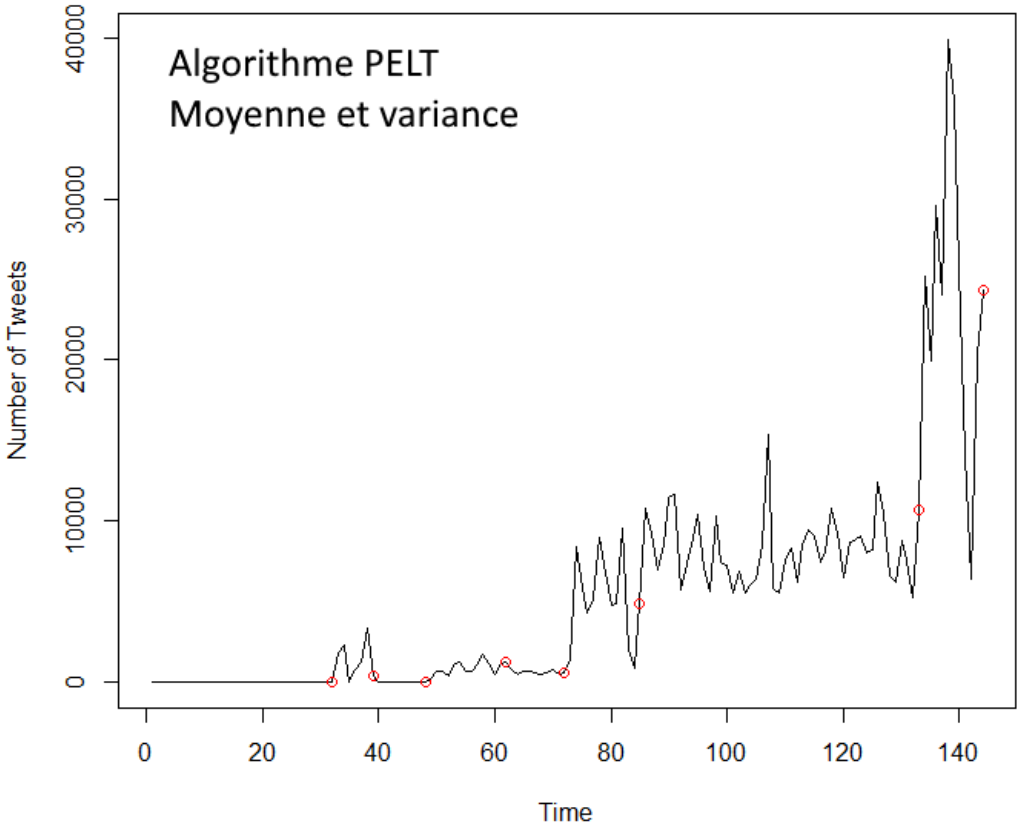

FIGURE 5.12 – Projet covoiturage - Uber - Algorithme PELT - Moyenne et variance

pour suivre l'évolution de l'utilisation des différents hashtags. Les différences dans les fréquences des hashtags peuvent servir de première approximation pour une description sémantique de l'événement.

Nous avons appliqué cette méthode sur le projet TEE 2014. La figure 5.13 montre une capture d'écran de l'application Web de détection et caractérisation d'événements, développée par Sergey Kirgizov [Kondrashova et al., 2016]. L'axe des abscisses représente le temps, et l'axe des ordonnées la fréquence des tweets contenant les hashtags intéressants. Les pics correspondent aux périodes d'utilisation des hashtags les plus fréquents, et chaque couleur représente la fréquence d'un hashtag particulier. Les deux ellipses qui ont été rajoutées sur la figure correspondent aux deux principaux événements de la campagne électorale :

— le vote (Election day) ;
— l'annonce des résultats (Result announcement day).

Des événements potentiels ont été détectés, ils sont matérialisés par les lignes grises. Ils correspondent aux moments où il y a eu un accroissement (ou une diminution) rapide du nombre de tweets.

À l'aide des deux barres verticales vertes, l'utilisateur peut sélectionner un intervalle de temps qui l'intéresse, et lancer une analyse détaillée des tweets émis durant cet inter-

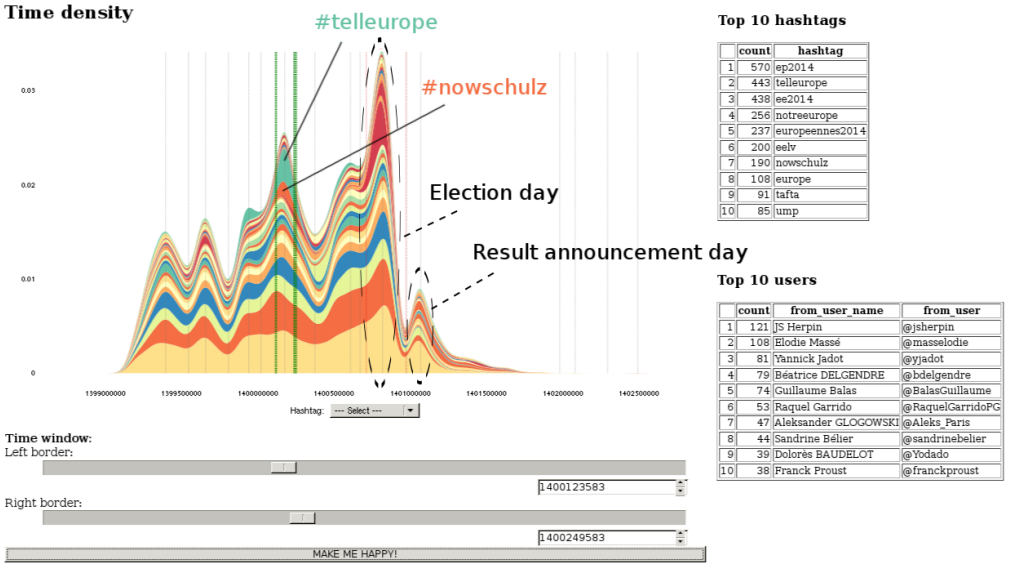

FIGURE 5.13 – Aperçu de l'interface Web pour la détection et la caractérisation d'événements à partir de données de Twitter (TEE 2014)

valle (bouton en bas à gauche). Les analyses peuvent être le calcul de fréquences, par exemple des 10 hahstags les plus utilisés, des 10 utilisateurs les plus prolifiques en nombre de tweets, ou bien un calcul des hubs et des authorities, etc.

De nombreuses informations utiles peuvent être extraites des données issues de Twitter en utilisant cette application Web. Par exemple, considérons le pic entre les deux barres verticales épaisses que l'on peut voir sur la figure 5.13. On peut remarquer que les hashtags #TellEurope et #NowSchulz sont devenus très populaires durant cette période, alors qu'en dehors de celle-ci ils ne sont quasiment jamais utilisés. Ce pic est dû au débat *Tell Europe* organisé par l'Union européenne de radio-télévision [26] et qui a eu lieu durant cette période et auquel Martin Schulz a participé.

5.7.2/ ÉVALUATION DE L'INFLUENCE

De nombreux travaux concernent l'influence dans les réseaux sociaux et abordent la question de deux grandes manières distinctes :

— selon la détection des personnes influentes dans un graphe ;
— selon l'estimation ou l'évaluation de l'influence, qui est ensuite utilisée pour un classement.

Pour les approches du second type, on se réfère souvent à une définition de l'influence qui consiste à mesurer son résultat, c'est-à-dire les liens qu'un utilisateur possède avec les autres utilisateur du réseau.

Pour les approches du second type, l'influence peut être définie comme la capacité d'un utilisateur à provoquer une action chez un autre utilisateur [Leavitt et al., 2009]. Le terme **action** désigne les différentes interactions possibles entre les utilisateurs. Par exemple les opérateurs liés à une action sur Twitter sont (*retweet*, *réponse*, *mention*, *suivre*). Un

26. https://www.ebu.ch/fr/home

utilisateur peut *suivre* un autre utilisateur, ce qui lui permet de voir les *tweets* et les informations de l'utilisateur qu'il suit. Un utilisateur peut *retweeter* un *tweet*, ce qui expose ce *tweet* à ses abonnés, qui peuvent à leur tour le *retweeter*. Un utilisateur peut *mentionner* un autre utilisateur en utilisant le préfixe @ s'il veut lui adresser le *tweet*. Enfin, un utilisateur peut *répondre* à un *tweet* et créer ainsi une conversation avec l'utilisateur du *tweet* initial. Cette définition est beaucoup plus complexe à modéliser que la première, en effet elle inclut une relation de causalité et un aspect dynamique. Très peu de travaux en tiennent compte et les méthodes développées reposent en fait sur une définition du premier type.

Dans notre cas, pour les besoins identifiés dans le contexte de l'entreprise nous nous concentrons sur l'identification des personnes influentes dans des données sociales modélisées sous la forme d'une graphe et nous étudions les mesures topologiques (par le nombre de liens ou chemins).

5.7.2.1/ MESURES DE CENTRALITÉ

Une mesure est un indicateur quantitatif qui est calculé à partir des composants du graphe, que ce soit des sommets, des arêtes ou des arcs. Il s'agit soit de mesures globales, soit de mesures locales (nœuds, chemins). Une mesure globale tente de résumer de manière simple la structure globale d'un graphe [Ducruet, 2010]. Une mesure locale permet de décrire un élément du graphe par rapport aux autres éléments [Ducruet, 2010]. Les mesures locales peuvent être scindées en deux catégories : les mesures locales de voisinage, ne s'occupant que d'un élément et de ses voisins immédiats ; et les mesures locales d'ensemble, prenant en compte un élément et tous les autres éléments de même nature.

Les mesures de centralité cherchent à prendre en compte des notions générales telles que la cohésion, le prestige, la notoriété, l'influence. Cependant, il n'existe pas de réel consensus sur une définition unique de la centralité et sur son interprétation en tant que phénomène du monde réel, principalement du fait de la dépendance au domaine, c'est-à-dire du lien implicite qu'il existe entre la modélisation du graphe et le domaine étudié. Cette notion est étroitement liée à l'importance d'un nœud ou d'une arête. On peut ainsi définir plusieurs mesures de centralité.

La centralité de degré DC, ou *degree centrality*, mesure l'intensité de la connexion d'un nœud à ses voisins. Elle correspond au degré du nœud en question, qui a déjà été évoqué dans la partie 3.2.2.

La centralité de degré peut aussi être normalisée afin de traduire un potentiel de communication, en calculant le rapport entre le degré du nœud et le degré maximal possible, c'est-à-dire, pour un nœud i et un nombre total de nœuds n :

$$DC_{norm}(i) = \frac{DC(i)}{n-1}$$

La centralité de proximité CC, ou *closeness centrality*, mesure la proximité d'un nœud à tous les autres nœuds. Plus la mesure est petite, plus le nœud est important. Plusieurs versions existent. Une définition utilisée souvent est : l'inverse de la somme des plus courts chemins d'un nœud i à tous les autres nœuds [Kepner et al., 2011].

$$CC(i) = \frac{1}{\sum_{i \in V} d(j,i)}$$

La centralité de stress SC, ou *stress centrality*, d'un nœud i, définie par [Shimbel, 1953], mesure le nombre de plus courts chemins passant par i. On note σ_{st} le nombre de plus courts chemins entre deux nœuds, appelés ici s et t, d'un graphe, et $\sigma_{st}(i)$ le nombre de plus courts chemins entre les deux nœuds s et t passant par le nœud i. Cette mesure donne une information sur la quantité de communication qui passe par un nœud donné, ce qui donne un niveau de *stress* du nœud, en partant du principe que toutes les communications se font en utilisant tout le temps le plus court chemin possible entre les différents nœuds. Si cette centralité est élevée pour un nœud, ce dernier peut faire office de "point de passage" pour relier d'autres nœuds. Cette mesure est applicable également aux arêtes.

$$SC(i) = \sum_{s \neq t \neq i \in V} \sigma_{st}(i)$$

La centralité d'intermédiarité BC, ou *betweeness centrality*, définie par [Freeman, 1977], est une autre mesure du nombre de plus courts chemins du graphe passant par chaque sommet. Elle peut être vue comme une version normalisée de la *stress centrality*.

$$BC(i) = \sum_{s \neq t \neq i \in V} \frac{\sigma_s t(i)}{\sigma_{st}}$$

La centralité de Katz CK, définie par [Katz, 1953], mesure l'influence relative d'un nœud, en ne prenant pas en compte les plus courts chemins, mais le nombre total de chemins entre deux nœuds. Pour la calculer, il est nécessaire de connaître la matrice d'adjacence A et de l'élever à la puissance k afin de connaître les chemins de degrés k entre deux nœuds i et j. Un paramètre α doit être choisi de manière à ce qu'il soit plus petit que l'inverse la valeur absolue de la plus grande valeur propre de A.

$$CK(i) = \sum_{k=1}^{\infty} \sum_{j=1}^{\infty} \alpha^k (A^k)_{ji}$$

La centralité de portée CR, ou *reach centrality*, définie par [Gutman, 2004], correspond au degré avec lequel n'importe quel membre d'un réseau peut atteindre les autres membres du réseau. Vue autrement, elle mesure le nombre de nœuds qu'un nœud i peut atteindre en k étapes ou moins. Pour $k = 1$, $CR = DC$.

D'autres mesures du même type existent. Il existe aussi d'autres manières d'aborder le problème du calcul de la centralité, par exemple la notion de vitalité, ou *vitality*, qui détermine l'importance d'un nœud ou d'une arête en faisant la différence entre une mesure quantitative caractérisant le graphe, et cette même mesure caractérisant le graphe privé du nœud ou de l'arête en question [Brandes et al., 2005]. Il existe plusieurs vitalités, dépendantes de la mesure choisie et du type d'entité étudié.

5.7.2.2/ HUBS ET AUTHORITIES : HITS

Le modèle proposé par Kleinberg [Kleinberg, 1999] découle d'une observation sur la création de pages Web. En effet, aux débuts du Web, certaines pages connues, nommées hubs, servaient de répertoires ou de catalogue d'informations pour conduire les utilisateurs à d'autres pages faisant autorité, ou authorities. En d'autres termes, un "bon hub" représente une page qui aiguille vers de nombreuses autres pages faisant autorité, et une "bonne autorité" est une page liée par de nombreux hubs différents. Cette définition croisée a servi de base à un algorithme de calcul des hubs et autorités.

L'algorithme HITS (*Hyperlink-Induced Topic Search*) repose sur ce modèle et calcule pour chaque nœud deux scores : un score d'autorité et un score de hubs. Ainsi, à l'issue de plusieurs itérations, il peut fournir deux listes ordonnées des nœuds : les *authorities* (nœuds recevant beaucoup de liens) et les *hubs* (nœuds à l'origine de nombreux liens vers des pages faisant autorité). Le score d'autorité d'une page est la somme des scores de hubs des pages qui pointent vers elle. Symétriquement, le score de hub d'une page est la somme des scores d'autorité des pages qu'il relie.

L'algorithme itératif pour calculer les scores hub et autorité de chaque nœud notés respectivement $auth()$ et $hub()$ est le suivant :

1. initialisation : $\forall v \in V, auth(v) = hub(p) = 1$
2. mise-à-jour des scores :
   1. $auth(v) = \sum_{v' \in N^-(v)} hub(v')$ où $N^-(v)$ représente l'ensemble des nœuds qui ont un lien vers v
   2. $hub(v) = \sum_{v' \in N^+(v)} auth(v')$ où $N^+(v)$ représente l'ensemble des nœuds vers lequel v a un lien
3. boucler sur l'étape 2 jusqu'à convergence des scores

La condition d'arrêt peut être difficile à expliciter. Il est possible de l'affaiblir et de ne considérer qu'une convergence du rang global, c'est-à-dire une valeur stable du rang de chaque nœud pour les vecteurs score d'autorité et de hub. Une autre possibilité est de normaliser les résultats à chaque itération et de fixer une valeur de seuil pour évaluer la convergence [Langville et al., 2005].

Une autre expression utilise une approche matricielle et une décomposition en valeur singulière. On considère la matrice d'adjacence A, de taille $n \times n$, les vecteurs h et a de taille n représentant respectivement les scores de hub et autorité pour les n nœuds du graphe. Ainsi $h = A.a$ et $a = A^\intercal.h$. En replaçant a et h dans les deux équations on obtient $a = AA^\intercal a$ et $h = A^\intercal Ah$. Par conséquent, déterminer les vecteurs propres de $AA^\intercal$ et $A^\intercal A$ permet de calculer les scores d'autorité et hub des nœuds du graphe. Ils peuvent être calculés par la méthode de la puissance itérée (*power iteration method*), ou par une décomposition en valeur singulières. Pour cette dernière, on pose $A = U\Sigma V^\intercal$ où U et V sont des matrices carrées orthogonales et Σ est une matrice diagonale contenant les valeur singulières de A qui, élevées au carré, sont les valeurs propres de A. Ce qui donne en remplaçant dans les équations précédentes :

$$AA^\intercal = (U\Sigma V^\intercal)^\intercal(U\Sigma V^\intercal) = V\Sigma^\intercal U^\intercal U\Sigma V^\intercal = V(\Sigma^\intercal\Sigma)V^\intercal = V\Sigma^2 V^\intercal$$

Ou encore $AA^\intercal V = V\Sigma^2$. Si V_j est la j^{ème} colonne de V, $AA^\intercal V_j = \sigma_j^2 V_j$. On obtient une expression symétrique pour $A^\intercal A$ et par conséquent, les vecteurs singuliers de V et U sont

les vecteurs propres de AA^T et A^TA. Le premier vecteur de chacune des deux matrices donnent respectivement le score de hub et d'autorité. La version utilisant la SVD a l'avantage de s'affranchir du critère de convergence de la version itérative de l'algorithme.

5.7.2.3/ GRAPHE DES RETWEETS : HUBS ET AUTHORITIES DANS LE PROJET TEE 2014

Cette expérience avait pour but d'effectuer des analyses afin d'identifier les utilisateurs influents, et de s'intéresser à la manière dont se diffuse l'information.

Cette analyse est basée sur l'algorithme `HITS` utilisé ici sur le graphe des retweets, c'est-à-dire un graphe pondéré et orienté, afin de mettre en évidence les hubs et authorities. Dans le contexte des tweets et retweets, les hubs sont les comptes diffusant beaucoup l'information, et les authorities sont les comptes considérés comme faisant référence au sein d'une communauté, sur un domaine donné, ce qui leur confère une certaine influence.

Pour appliquer l'algorithme `HITS`, nous construisons une matrice carrée d'utilisateurs grâce aux retweets. Nous avons construit la matrice des utilisateurs ayant retweeté plus de 20 fois des tweets émis par des comptes politiques identifiés. Les hubs et authorities calculés par l'algorithme ont ensuite été analysés par les chercheurs en sciences sociales partenaires du projet TEE. Les chercheurs ont identifié certaines spécificités qui n'ont pas été trouvées en utilisant des techniques traditionnelles d'analyse (comme la fréquence des hashtags et le nombre de retweets par utilisateurs).

En utilisant les techniques traditionnelles d'analyse, c'est-à-dire principalement des requêtes de comptage faisant intervenir une clause `GROUP BY`, les chercheurs en sciences humaines avaient conclu que les petits partis politiques se comportent globalement tous de la même manière. Cependant, les Top 10 des hubs et authorities sont majoritairement monopolisés par des membres et des personnalités politiques d'un petit parti.

Rang	Autorités	Hubs
1	Marine Le Pen	X
2	Florian Philippot	X
3	Louis Aliot	X
4	Jean-Marie Le Pen	Nathalie Germain
5	Steeve Briois	Brigade Patriote
6	Bruno Gollnisch	X
7	Gilles Lebreton	X
8	MC.Arnautu	MarinePrésidente2017
9	Alain Cadec	X
10	Arnaud Danjean	X

TABLE 5.2 – Projet TEE 2014 - Top 10 des hubs et autorités (X est utilisé pour anonymiser des comptes d'utilisateurs autres que des personnages publics ou des comptes des partis ou des personnes officielles des partis)

5.7.2.4/ HUBS ET AUTHORITIES DANS LE PROJET SUR LE CO-VOITURAGE

L'algorithme HITS appliqué sur un graphe orienté et pondéré a aussi été utilisé pour le projet sCRM de eb-Lab sur l'étude du co-voiturage. Nous avons étudié un événement en

particulier, identifié grâce à la méthode développée dans [Killick et al., 2012a] et décrite dans la partie 5.7.1.2. Cet événement était lié aux conducteurs de taxi bloquant certaines routes autour de Paris pour protester contre le service de VTC[27] *UberPop*. Nous cherchions à voir quelle était l'activité de l'entreprise *Uber* durant cet événement, et quels étaient les comptes influents.

Nous avons réalisé le calcul des hubs et authorities avec deux paramétrages différents.

Le premier calcul a été fait sans filtre. La matrice construite contenait donc l'ensemble des utilisateurs qui avait retweeté au moins une fois.

Le résultat du premier calcul (tableau 5.3), montre les comptes liés aux conducteurs et associations de taxis, et les personnes liées à cette profession dans le Top 10 des authorities : *Taxi_de_Paris, PierrePeyrard, MonPereCeTaxi* On trouve aussi une entreprise liée à la mobilité : *InnovMobi*, et un compte sur les VTC autre qu'*Uber* : *ASSOCIATIONVTC*. Les hubs sont aussi liés à la communauté des taxis (*CGT_TAXIS, taxiazur*) avec aussi plusieurs comptes robots (*999Hha, StepouneTest*). *Uber* et ses concurrents autres que les taxis n'apparaissent pas dans ces Top 10.

Rang	Autorités	Hubs
1	Taxi_de_Paris	999Hha
2	ThibaudDELETRAZ	CGT_TAXIS
3	PierrePeyrard	StepouneTest
4	MonPereCeTaxi	taxiazur
5	del_tass	94ALLADIN
6	InnovMobi	MoMoElTaCo
7	zaherinho	augenoux
8	ASSOCIATIONVTC	nabilakoff
9	nabilakoff	zaherinho
10	pumbijourdain	Le_Terrier

TABLE 5.3 – Projet co-voiturage - Top 10 hubs et autorités en utilisant tous les comptes disponibles

Le second calcul a été effectué en filtrant les comptes retweetés. Ces derniers, que l'on appelle *comptes suivis*, étaient les comptes officiels d'*Uber* et quelques concurrents français opérant à Paris.

Le résultat du deuxième calcul (tableau 5.4) montre certains *comptes suivis*, mais pas la totalité, comme hubs et authorities : *UberLyon, Uber_Lille, Uber_Paris, Uber_Cannes*, mais on ne retrouve pas le compte officiel d'*Uber France*, *UberFR*. Le premier hub est un compte robot (*ConcoursRetweet*), et seul un concurrent d'*Uber* apparaît dans les authorities, *idVROOM*. Le Top 5 est significatif, mais ensuite il n'y a pas assez de données pour que les résultats soient pertinents. Cela montre que les *comptes suivis* n'ont pas été suffisamment retweetés. Nous en avons conclu qu'ils étaient restés discrets et n'ont pas pris part aux discussions liées à cet événement.

5.7.2.5/ VISUALISATION DES INTERACTIONS D'UN COMPTE TWITTER

Afin d'extraire de la connaissance des hubs et des authorities, et d'analyser plus spécifiquement le comportement d'un utilisateur, plusieurs composants ont été

27. Véhicule de Transport avec Chauffeur

Rang	Autorités	Hubs
1	UberLyon	ConcoursRetweet
2	iDVROOM	UberLyon
3	Uber_Lille	Uber_Paris
4	Uber_Cannes	Uber_Lille
5	Uber_Paris	Uber_Cannes
6	Seatecawen	Seatecawen
7	patron_pme	patron_pme
8	nicolaslr	nicolaslr
9	Mbcustode	Mbcustode
10	MagalieBarreira	MagalieBarreira

TABLE 5.4 – Projet co-voiturage - Top 10 hubs et autorités avec une restriction sur les comptes retweetés

développés. La figure 5.14 montre les tweets d'un utilisateur, au centre de la figure, ici *jphuchon*, et qui les retweete. Les ronds rouges correspondent à des comptes Twitter, les ronds bleus à des tweets. Les flèches rouges partent d'un compte Twitter et vont vers les tweets émis par ce compte. Les flèches bleues indiquent qu'un compte a retweeté un tweet. Les flèches roses concernent les tweets retweetés par le compte étudié.

La figure 5.15 montre les tweets d'un utilisateur sur une période de temps donnée, sous forme de *timeline*. L'outil développé permet de sélectionner un tweet afin de voir son contenu.

5.7.3/ DÉTECTION DE COMMUNAUTÉS : LE RÉSEAU HASHTAG - UTILISATEURS

Cette expérience avait pour but d'identifier des communautés à partir d'un ensemble de comptes Twitter et d'utiliser des connaissances du domaine afin d'améliorer la visualisation et l'interprétation des résultats. Pour détecter les communautés, nous avons utilisé l'algorithme *Walktrap*, décrit dans l'état de l'art.

Nous nous sommes appuyés sur le projet *TEE 2014* et le jeu de données du corpus français afin de trouver des communautés formées par les comptes politiques officiels des candidats, en ne prenant en compte qu'un sous-ensemble des comptes concernés par la collecte. C'est-à-dire, dans le corpus français, une cinquantaine de comptes, généralement des têtes de listes, qui étaient également ressortis de analyses précédentes (hubs-authorities par exemple).

Le graphe produit contient les comptes Twitter sélectionnés, ainsi que les différents hashtags les plus fréquents, utilisés par ces même comptes. Afin d'éliminer les hashtags non significatifs, nous les avons filtrés grâce à un test binomial. Une ontologie du domaine politique étudié a été définie, incluant les partis politiques, leur région, leur pays, les noms des candidats et de leurs comptes officiels ainsi que la position de chaque candidat dans sa liste. La figure 5.16 en présente un extrait.

Une fois les communautés identifiées grâce au *Walktrap*, nous avons comparé les membres des communautés trouvées par l'algorithme avec les affiliations décrites par l'ontologie. Quelques singularités ont été détectées. Pour aider les chercheurs en sciences sociales à analyser et interpréter ces singularités, nous avons réalisé une visualisation en utilisant différentes couleurs en fonction des membres et des communautés.

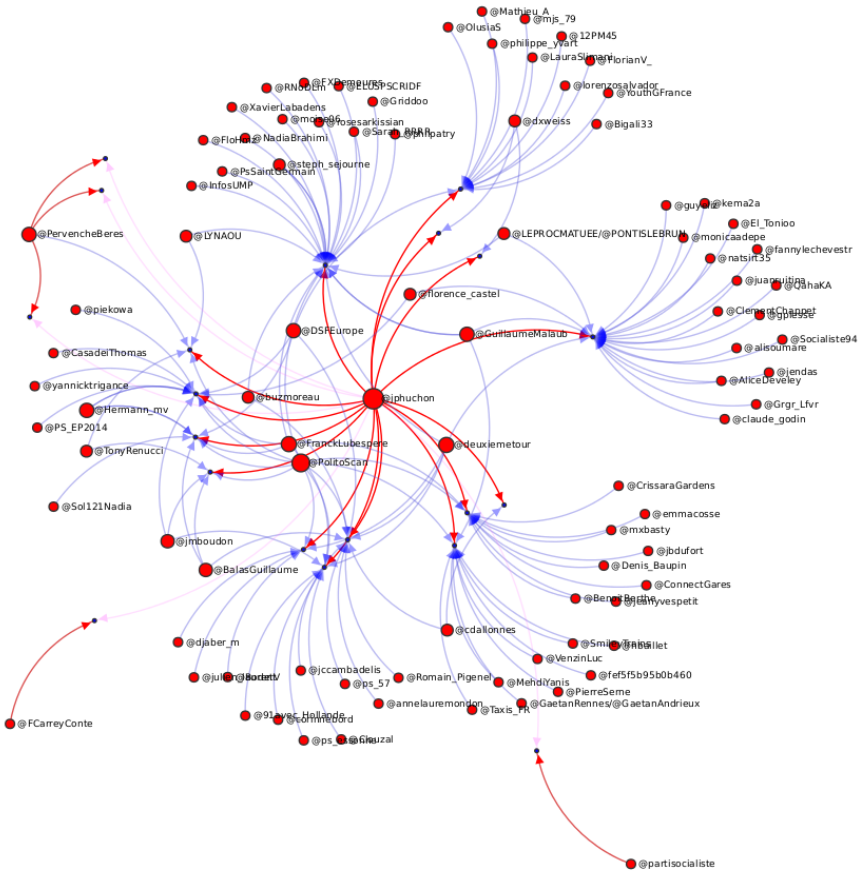

FIGURE 5.14 – Tweets et retweets d'un utilisateur en particulier

Le résultat est décrit dans la figure 5.17, les hahstags étant représentés par des disques jaunes, et la couleur des utilisateurs dépendant de leur parti politique. Les nœuds de la même communauté sont entourés par une seule et même couleur. Les hashtags présents dans les différentes communautés donnent une caractérisation de ces communautés. Nous pouvons remarquer que, la plupart du temps, les utilisateurs d'un même parti ont tendance à utiliser les mêmes hashtags et se retrouvent dans une communauté qui correspond assez bien au parti. Cependant, la figure met en évidence une singularité par exemple, en bas à droite, un membre (le nœud orange dans la communauté jaune) d'un parti politique a utilisé les mêmes hashtags que les membres d'un autre parti politique (les nœuds rouges).

5.8/ COMPARAISON DE SNFREEZER ET DES PLATEFORMES EXISTANTES ET CONCLUSION

Nous allons maintenant comparer les différentes fonctionnalités de SNFreezer avec quelques projets concurrents. Nous avons étudié le mode de stockage (polyglotte ou non), son indépendance, la présence d'un mode cluster, d'une fonctionnalité de reprise sur panne, d'une interface utilisateur, ainsi que la possibilité d'utiliser les différentes APIs de Twitter et d'étudier les données avec des analyses séquentielles déjà disponibles ou

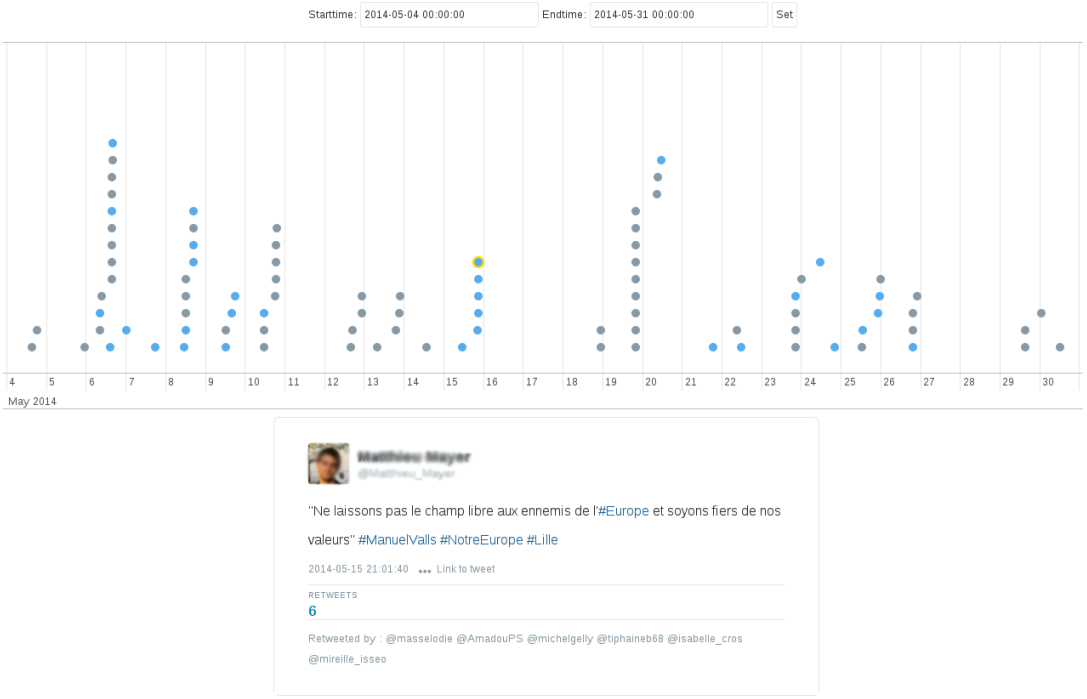

FIGURE 5.15 – Tweets d'un utilisateur en particulier sur une période de temps, montrés sous forme de frise chronologique

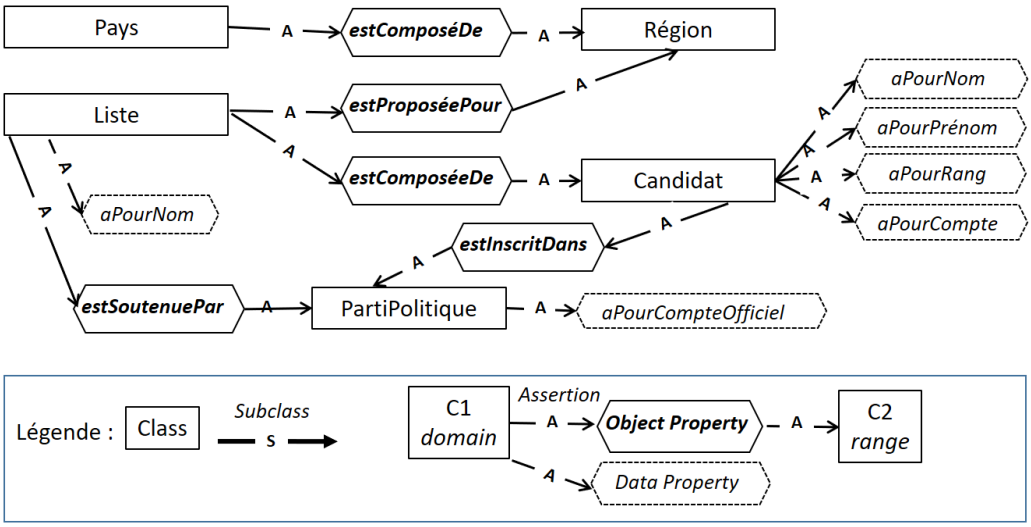

FIGURE 5.16 – Extrait du schéma de l'ontologie présenté en G-OWL[Héon et al., 2013]

avec des analyses pluggables. Le tableau 5.5 présente les différents projets et critères de comparaison retenus.

Le projet Sciences Po Medialab [28] fournit des composants individuels pour collecter, stocker, analyser et afficher des graphes sociaux. C'est un ensemble d'outils largement indépendants et difficiles à connecter entre eux sans connaissance en programmation

28. http://tools.medialab.sciences-po.fr

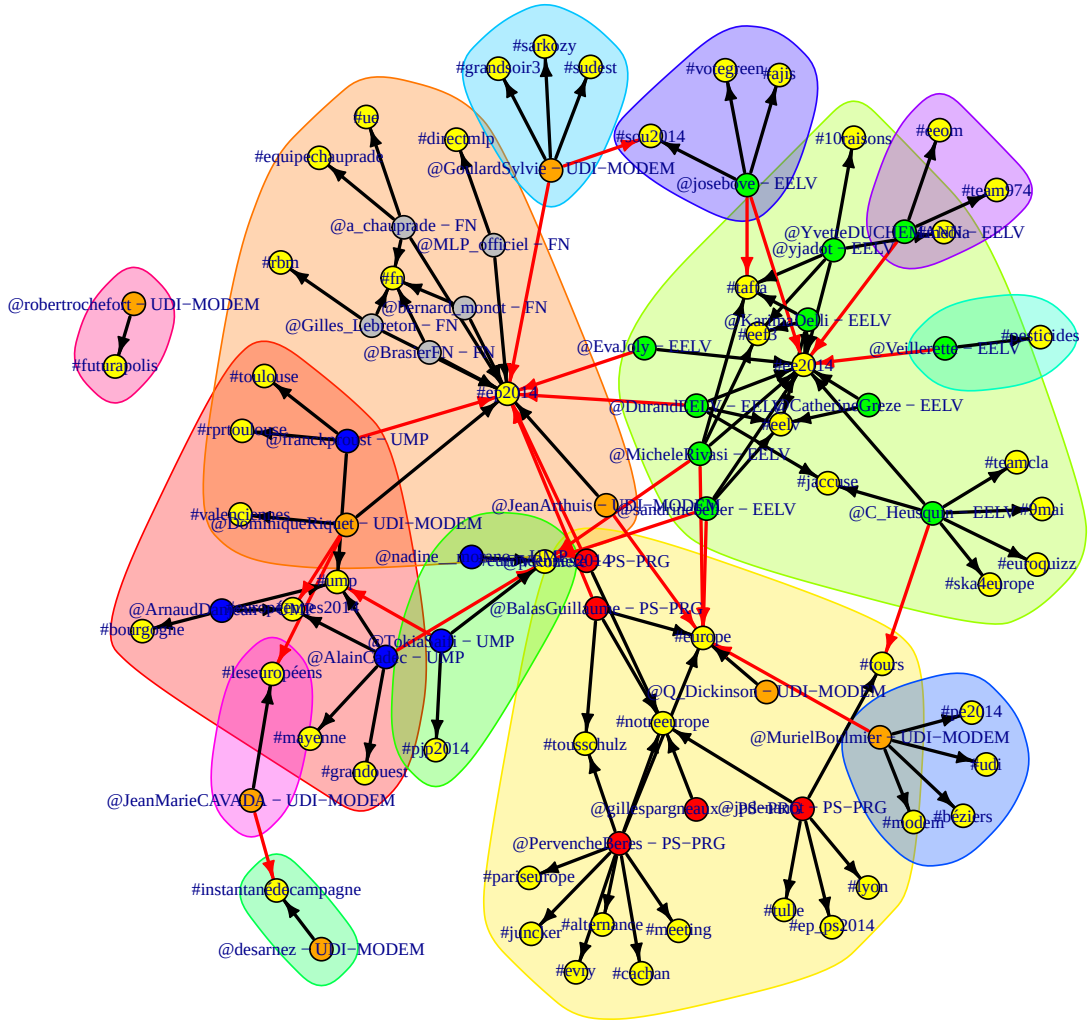

FIGURE 5.17 – Communautés et singularités détectées dans le corpus français de TEE 2014

informatique. L'outil de collecte de tweets permet de se connecter aux différentes APIs de Twitter, mais ne propose pas de mode cluster ou de reprise sur panne. De plus, le stockage n'est pas polyglotte, la couche de stockage n'est pas indépendante et il n'y a pas d'interface utilisateur intégrée. Au niveau des analyses, il est possible de rajouter des outils, mais le projet ne dispose pas de séquençage des analyses.

Dans [Burnap et al., 2014], les auteurs décrivent un service intégré pour analyser les médias sociaux à la demande. Leur plateforme COSMOS propose d'utiliser un ensemble d'outils open-source pour des analyses de textes et de réseaux via une interface utilisateur intégrée. Ils se concentrent sur des analyses de l'opinion, autour d'un événement significatif. Ils utilisent d'autres sources que Twitter, mais n'utilisent que la *Streaming API* pour collecter des tweets. De plus, ils ne proposent pas de stockage polyglotte ni d'indépendance de la couche de stockage, pas de mode cluster ou de reprise sur panne et pas de possibilité de rajouter des outils pour l'analyse des données.

Dans [Li et al., 2012], les auteurs décrivent un système spécialisé pour la détection d'événements. Leur plateforme TEDAS fournit une interface de visualisation, mais leur outil de collecte est très simple, sans stockage polyglotte, ni d'indépendance de la couche

de stockage, ni de mode cluster ou de reprise sur panne. Leur but principal étant la détection d'événements, d'autres types d'analyse ne sont pas encore disponibles.

Ces plateformes ont toutes des buts spécifiques, mais ne couvrent pas un large domaine comme peut le faire SNFreezer. Cependant, ce dernier ne comporte pas d'interface utilisateur intégrée ; car l'objectif de la plateforme est de fournir un framework multi paradigmes avec de nouveaux outils facilement intégrables qui peuvent être utilisés de différentes manières (ensemble, séparément, etc.). Nous pensons que l'interaction avec les utilisateurs dépend du domaine d'utilisation de la plateforme et nous laissons le choix de l'interface utilisateur et de la combinaison des différents outils aux utilisateurs de notre framework afin qu'ils puissent construire ce qu'on appelle un *observatoire*.

Nous ne connaissons pas d'approche mixant un outil de collecte de tweets avec des fonctionnalités avancées telles qu'un mode cluster pour contourner les différentes limitations des APIs Twitter, un mécanisme de reprise sur panne afin de s'assurer qu'aucun tweet ne soit oublié durant la collecte, avec la possibilité de rajouter différents modules d'analyses pouvant être utilisés de manière séquentielle afin de réaliser des analyses incrémentales et itératives.

Le développement et les différents tests effectués sur SNFreezer nous ont permis de valider le concept de framework et de plateforme que nous avions imaginé. Le temps de développement a été de 80 jours homme. Cependant, cet outil reste limité pour ce qui est de l'évolutivité. En effet, la partie stockage n'est pas complètement indépendante du reste de l'application. De plus, les outils d'analyse ne sont pas totalement intégrés à la plateforme. Celle-ci prend plus la forme d'une application Web avec des outils d'analyse à part et dont le couplage est réalisé de manière ad-hoc par un format pivot qui consiste à extraire des bases de données, pour chaque type de relation, des triplets : item1, item2, intensité. Par exemple, si il s'agit du nombre de retweets entre utilisateurs, on obtiendra des triplet de la forme (U1,U2,nbRT). Par ailleurs, aucun moteur de workflow n'est disponible dans SNFreezer. Le séquencement des algorithmes d'analyse est fait "à la main" au moyen de scripts shell ou R et aucun contrôle n'est apporté par la plateforme quant à la pertinence de la séquence d'analyse réalisée. L'orientation collecte, stockage et analyse de données Twitter de SNFreezer ne s'intègre pas tout à fait avec la méthode proposée dans la chapitre 4 qui utilise des données provenant d'une plateforme collaborative pour établir le profil utilisateur et enchaîne différents algorithmes de détection de communautés. Ces différentes limitations nous ont conduit à envisager le développement d'une autre plateforme pour le compte d'eb-Lab afin de partir sur une base technique plus homogène pour une implémentation et une utilisation industrielle tout en bénéficiant de l'expérience acquise lors du développement de SNFreezer.

	SNFreezer	Medialab	COSMOS	TEDAS
Interface Utilisateur Intégrée	Non	Non	✓	✓
Stockage Polyglotte	✓	Non	Non (Détection du langage)	Non (Prédiction de la localisation)
Indépendance de la couche de stockage	✓	Non	Non (MongoDB, distribution des données avec Hadoop)	Non
Mode Cluster	✓	Non	1 connection à Twitter, analyses en parallèle	Non
Plusieurs APIs Twitter	✓	✓	Non (Streaming API et autres sources que Twitter)	Non
Reprise sur panne	✓	Non	Non	Non
Analyses séquentielles	✓	Non	9 types d'analyses disponibles (tweet ou au niveau du corpus), mais pas de séquençage	Uniquement détection et analyse d'événements
Plusieurs types d'analyses pluggables	✓	✓	Non	Uniquement détection et analyse d'événements

TABLE 5.5 – Comparaison de SNFreezer et des plateformes existantes

6

Implémentation de la plateforme DiscoCRM, évaluation et retour d'expérience en entreprise

Sommaire

6.1/ INTRODUCTION

6.1.1/ CONTEXTE DU PROJET

Le développement du projet DisCoCRM [1] a commencé au second semestre 2014. Il s'inscrit dans la démarche des sociétés eb-Lab et Teletech International d'intégrer les réseaux sociaux dans leur offre d'outils CRM (appelée `Nest CRM` [2]), initiée avec le début de ma thèse. Une première expérimentation, sur le réseau social Facebook, a été réalisée à travers l'application Social Buddies [3]. Développée par eb-Lab, cette application s'intègre dans une page Facebook pour y ajouter un aspect relation client, sous forme de questions-réponses entre les utilisateurs, ainsi qu'avec les administrateurs de la page. Le concept de cette application est : 1) de s'affranchir des espaces imposés par Facebook pour les discussions en proposant un espace de discussion entre *fans* d'une page, avec la possibilité de voter pour les réponses les plus pertinentes et 2) de proposer un forum de discussion et un endroit dédié aux personnes ayant aimé une page. Cette application s'est limitée à ce seul réseau social.

L'intégration de plusieurs réseaux sociaux est une perspective importante pour l'offre social-CRM (s-CRM). Une première orientation a consisté à développer pour Twitter l'équivalent de l'application Social Buddies afin de permettre à un *community manager* de gérer plusieurs comptes Twitter et ainsi de pouvoir répondre aux différentes catégories de questions des utilisateurs.

L'orientation qui concerne plus particulièrement mon projet et mon travail de thèse est très différente de celle de Social Buddies. À partir des résultats obtenus en 2013 sur les données du projet Vitagora, présenté dans le chapitre 4, l'objectif a été de réaliser une implémentation industrielle d'une plateforme qui servira de socle aux différentes applications de s-CRM d'eb-Lab et de Teletech International. La plateforme doit ainsi permettre la collecte et le stockage des données issues des réseaux sociaux, et proposer des algorithmes pour les analyses de ces données, en s'appuyant sur l'expérience de SNFreezer présentée dans le chapitre 5 et en s'adaptant aux contraintes de l'entreprise. L'outil s'adresse principalement à des *community manager* et des conseillers clientèle, ainsi qu'à des services marketing pour le suivi des campagnes marketing sur les réseaux sociaux.

Le regroupement des différents utilisateurs présents sur les réseaux sociaux en communautés thématiques, ainsi que la détection des personnes influentes, permettront d'élargir l'offre d'applications s-CRM en proposant une analyse plus en profondeur des réseaux autour d'une marque ou d'un produit.

Ce chapitre est organisé de la manière suivante : tout d'abord, les sections 6.1.2 et 6.1.3 précisent les objectifs du projet DisCoCRM et ses fonctionnalités requises. Ensuite, la partie 6.2 présente les différents cas d'utilisation de la plateforme, ainsi que son architecture technique et les différences par rapport à SNFreezer. La section 6.3 présente l'organisation et les phases du projet ainsi que l'environnement technique. Puis la section 6.4 décrit l'implémentation de l'outil de collecte de données issues des réseaux sociaux. La section 6.5 précise les contraintes et la réalisation de l'entrepôt de données.

1. *Discovering Communities for CRM*
2. Plateforme CRM de Teletech International, développée pour fournir un ensemble d'outils assemblés afin d'offrir une solution CRM spécifique aux besoins des clients
3. http ://www.social-buddies.fr/

La section 6.6 présente l'intégration des algorithmes et outils d'analyse au sein de la plateforme. Enfin, la section 6.7 décrit l'implémentation de l'application Web de contrôle permettant de faire le lien entre l'outil de collecte de données, l'entrepôt de données et les algorithmes et outils d'analyse.

6.1.2/ OBJECTIFS COMMERCIAUX DU PROJET DISCOCRM

L'objectif général du projet est de fournir des produits commercialisables basés sur une plateforme permettant d'intégrer des outils d'analyse de données issues des réseaux sociaux pour différents besoins. Trois positionnements commerciaux ont été identifiés.

Le premier positionnement concerne la préparation, le lancement et le suivi d'une campagne marketing sur les réseaux sociaux. Pour cela, la plateforme permet d'identifier les utilisateurs influents et pertinents à cibler pour s'en servir comme relais d'information ; ainsi que les hashtags à utiliser pour maximiser la visibilité de la campagne et le nombre d'utilisateurs potentiellement touchés. Aussi, la plateforme permet de suivre l'audience et l'efficacité de la campagne, ainsi qu'analyser des communautés d'utilisateurs qui se seront créées ou auront évolué suite à cette campagne.

Les deux autres positionnements concernent l'étude des discussions et des communautés existantes autour d'une marque ou d'un produit. Cela peut se faire soit de manière ponctuelle, soit sur la durée. Pour cela, la plateforme permet d'identifier les discussions autour d'une marque ou d'un produit et à en extraire et à caractériser les communautés s'agrégeant autour.

En fonction du positionnement, plusieurs types d'utilisateurs seront amenés à utiliser le produit : des *community manager*, des conseillers clientèle, des *data scientists* et des développeurs chez eb-Lab et Teletech International, ainsi que des chefs de projet fonctionnels liés à la marque ou au produit étudié.

Afin de répondre à ces besoins, plusieurs objectifs principaux ont été identifiés pour le développement de la plateforme DisCoCRM, qui devra permettre :

1. de supporter différentes catégories d'utilisateur :
   — chefs de projet fonctionnels, développeurs, *data scientists*, *community managers*, etc.
2. de disposer d'un ensemble d'algorithmes d'analyse et d'outils génériques pour ajouter de nouveaux algorithmes ;
3. d'avoir une vision globale d'un client ou d'un prospect sur les différents réseaux sociaux ainsi que dans les outils de gestion de la relation client, c'est-à-dire :
   — avoir un profil unique par personne identifiée dans la base de données des clients et des prospects ;
   — pouvoir accéder aux différentes informations présentes dans le CRM ainsi que dans ses profils sociaux.
4. de gérer plusieurs sources de données (différents réseaux sociaux, sites Internet, bases de données issues du CRM, etc.) et aussi de permettre l'ajout de nouvelles sources sans remettre en cause l'architecture globale de la plateforme.

La figure 6.1 donne un aperçu très général du positionnement de la plateforme DisCoCRM au sein de `Nest CRM`.

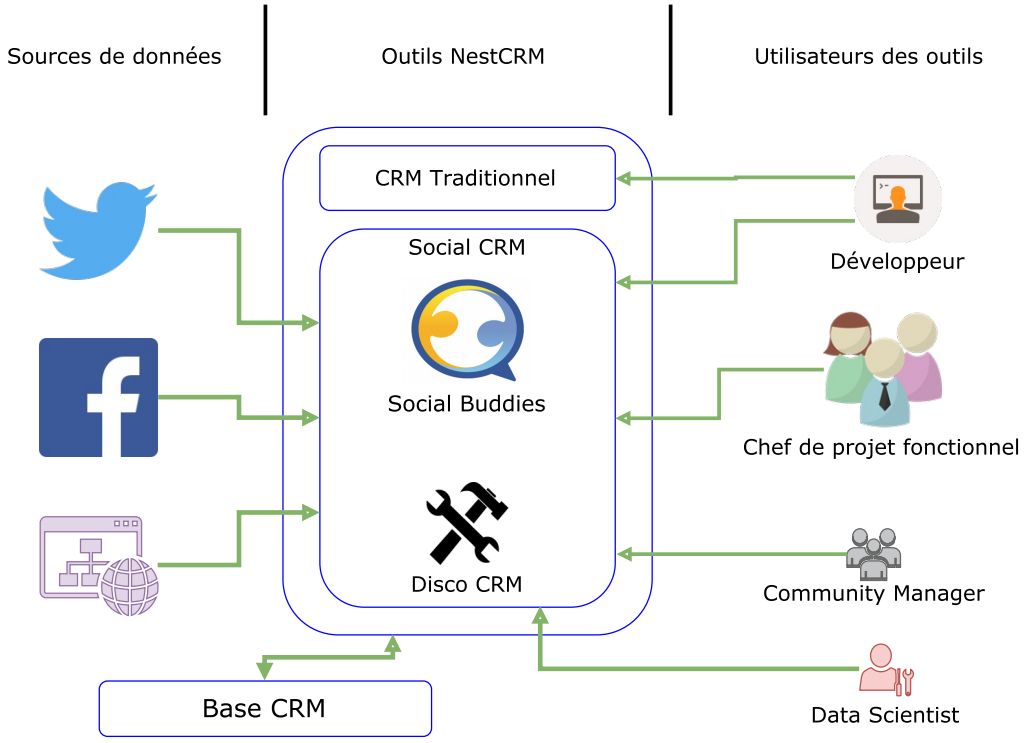

FIGURE 6.1 – Aperçu général du positionnement de DisCoCRM dans l'offre de Teletech International

Les figures 6.2 et 6.3 illustrent un des objectifs de la plateforme DisCoCRM : avoir une vision globale d'un client ou d'un prospect, que ce soit à travers les informations contenues dans la base CRM[4] de l'entreprise ou à travers ses différents profils sur les réseaux sociaux. Pour cela, nous construisons d'abord un profil unique pour un utilisateur des réseaux sociaux (figure 6.2), à partir des données collectées. Cette fusion des différents profils sociaux d'un même utilisateur peut se faire soit à la main, soit grâce à des algorithmes permettant de mettre en correspondance les différents profils. Ensuite, l'appariement entre le profil unique d'un utilisateur des réseaux sociaux, disponible dans DisCoCRM, et un client ou un prospect contenu dans la base CRM (figure 6.3) peut se faire de différentes manières : via son nom, un numéro de commande, ou via un conseiller clientèle au téléphone demandant à son interlocuteur son profil sur un des réseaux sociaux.

6.1.3/ FONCTIONNALITÉS DE LA PLATEFORME

En fonction du public ciblé par les différents positionnements de la plateforme, plusieurs fonctionnalités essentielles ont été identifiées.

Tout d'abord, des tableaux de bord et des rapports d'activité sont nécessaires afin de fournir diverses informations et statistiques via des écrans de suivi. Ils permettent de suivre l'activité liée à la marque ou au produit étudié, de suivre une campagne marketing sur

4. La base CRM est une base de données relationnelle contenant des informations sur les clients, les prospects, leurs interactions avec le service client, etc.

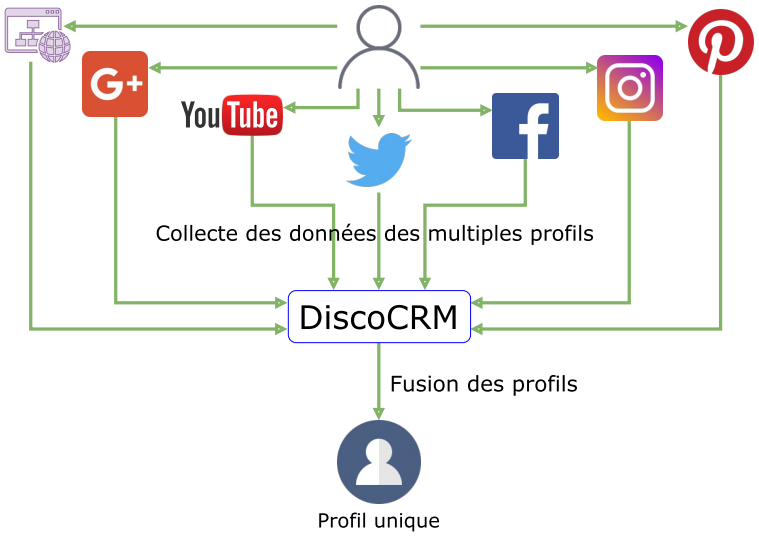

FIGURE 6.2 – Fusion des profils sociaux des internautes

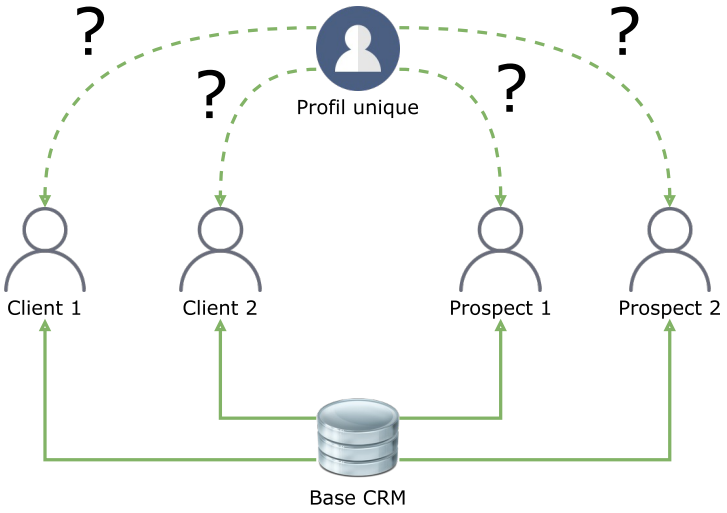

FIGURE 6.3 – Appariement du profil d'un internaute avec les données de la base CRM

les réseaux sociaux, ainsi que l'évolution de cette activité dans le temps, via différentes statistiques affichées : nombre de tweets, retweets, popularité des hashtags, utilisateurs influents, nombre de posts, etc. Ces dernières peuvent être calculées sur des périodes de temps différentes : par exemple sur une période donnée, ou à un instant t depuis le début de la collecte de données.

Ensuite, la détection et la caractérisation des communautés sur les réseaux sociaux constituent le cœur de la plateforme, tout particulièrement l'analyse du regroupement d'utilisateurs. Il est possible de caractériser les regroupements de différentes manières. Dans notre cas, nous en considérons deux :

— Les utilisateurs parlant beaucoup entre eux. Pour cela, en prenant comme thème commun un nom de marque ou de produit, il y a deux critères mesurables :
  — les re-tweets : les utilisateurs se retweetant beaucoup sont considérés comme

parlant entre eux ;
— les mentions du compte d'une marque, ou de personnes identifiées travaillant pour la marque / le produit.
— Les utilisateurs aux intérêts similaires. Pour cela, les hashtags contenus dans les tweets sont considérés comme une marque d'intérêt.

Enfin, la notion d'influence étant très pertinente d'un point de vue marketing, il est intéressant de détecter les différents utilisateurs influents, via par exemple le calcul des *hubs* et *authorities* présenté dans la partie 5.7.2.3. Cela permet par exemple, pour le lancement d'une campagne marketing, de cibler les utilisateurs pouvant servir de relais de la communication autour d'une marque ou d'un produit.

Les différentes caractéristiques statistiques peuvent être calculées à un instant donné, comme c'est le cas pour le deuxième positionnement des objectifs commerciaux présentés dans la partie 6.1.2. Cependant, elles évoluent au fil du temps et il est important d'offrir une fonctionnalité de suivi de cette évolution.

6.2/ Présentation de la plateforme DisCoCRM

À partir des objectifs et des fonctionnalités identifiées pour DisCoCRM, il est possible de retenir différents cas d'utilisation, et de définir l'architecture de la plateforme.

6.2.1/ Cas d'utilisation de la plateforme

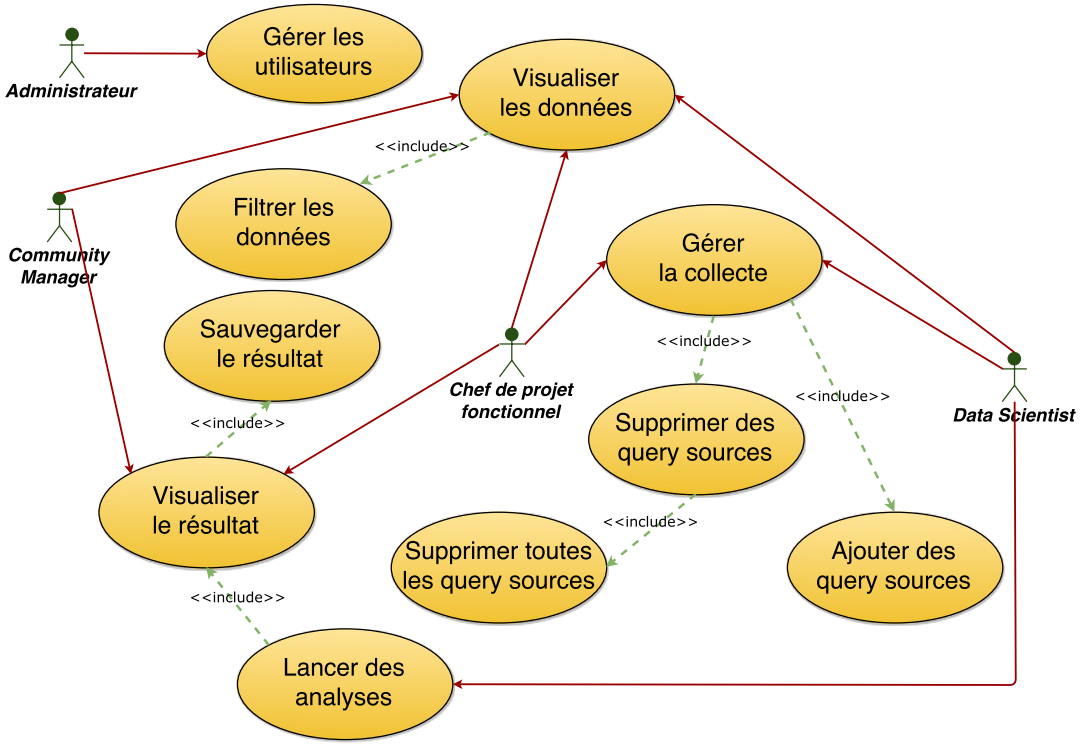

FIGURE 6.4 – Cas d'utilisation de DisCoCRM

La figure 6.4 présente les cas d'utilisation de la plateforme DisCoCRM. Les différentes catégories d'utilisateurs reprennent celles identifiées dans la partie 6.1.2.

Le chef de projet fonctionnel, travaillant pour le client d'eb-Lab et de Teletech International, peut visualiser les données issues de la collecte, ainsi qu'influer sur la collecte, en modifiant les différents critères (*query sources*). Un *data scientist*, travaillant pour eb-Lab et Teletech International, se charge de la conception des analyses en fonction des besoins du client. Il peut lui aussi gérer la collecte, et visualiser les données. Un *community manager*, travaillant pour eb-Lab et Teletech International, peut visualiser les données, ainsi que le résultat des analyses. La gestion des utilisateurs, c'est-à-dire des différents comptes et de leurs droits d'accès, est confiée à un administrateur.

6.2.2/ ARCHITECTURE GLOBALE

La figure 6.5 présente l'architecture de DisCoCRM et son intégration dans l'offre logicielle s-CRM de Teletech International.

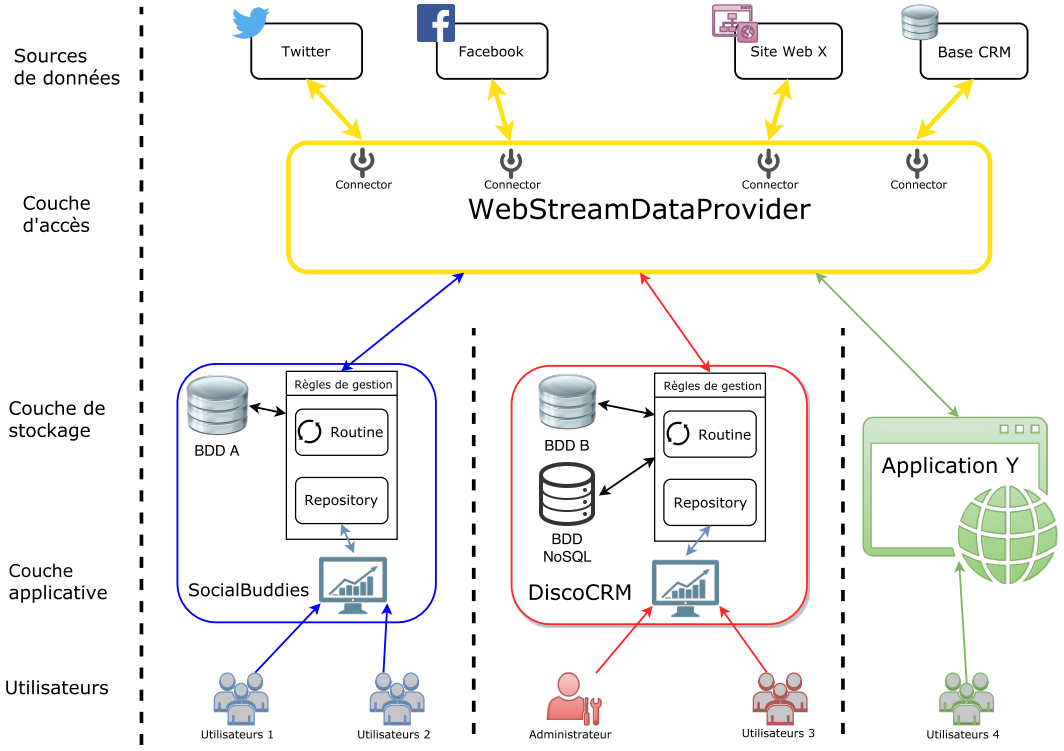

FIGURE 6.5 – Architecture de DisCoCRM

Plusieurs applications développées par eb-Lab et Teletech International ont besoin de collecter des données issues de sources multiples, que ce soit les réseaux sociaux (Twitter et Facebook sur la figure), la base CRM de l'entreprise et des sites Web pertinents. L'application `WebStreamDataProvider`, qui constitue la *couche d'accès* présentée dans la figure 6.5, a été développée afin de fournir un point d'accès unique à ces différentes applications, permettant d'unifier le processus de collecte, ainsi que le traitement des données sur toutes les applications et ainsi faciliter le développement et la maintenance.

Chaque application développée par Teletech International et eb-Lab se connecte à la

`WebStreamDataProvider` afin d'accéder aux données issues des réseaux sociaux, et dispose de sa propre *couche de stockage* permettant de stocker les différentes *règles de gestion*, les routines spécifiques à l'application, les données concernant les utilisateurs et leurs droits, ainsi que les données nécessaires au fonctionnement de l'application. Dans le cas de DisCoCRM, cette base de données correspond à la *BDD B* sur la figure 6.5 et appelée base de données interne dans la suite du chapitre. De plus, pour DisCoCRM, les données issues des réseaux sociaux sont stockées dans un entrepôt. Il s'agira d'une base de données NoSQL, notée sur la figure *BDD NoSQL*, qui doit apporter la flexibilité requise, compte tenu du caractère hétérogène des différentes sources de données.

6.2.3/ Positionnement et différences par rapport à SNFreezer

L'expérience acquise lors du développement de SNFreezer, qui est une plateforme similaire pour la gestion de la collecte et l'analyse des données issues de Twitter, sert de base pour la conception et le développement de DisCoCRM.

Cependant, les contraintes techniques et les fonctionnalités sont différentes ; ainsi que les utilisateurs finaux de la plateforme, des chercheurs dans le cas de SNFreezer, des *community manager*, gestionnaires de projet et *data scientists* dans le cas de DisCoCRM.

Plusieurs scripts d'analyse intégrant différents algorithmes ont déjà été développés dans le cadre de SNFreezer. Cependant, ils ne sont pas facilement utilisables par les utilisateurs ciblés. En effet, ils sont peu paramétrables, c'est-à-dire que la gestion des paramètres se fait dans le code du script avant chaque exécution. Les scripts ont été développés dans le langage *R* grâce à *R Studio*, cependant, afin de pouvoir exécuter ces scripts depuis une application Web, l'utilisation de *R Studio* est impossible. Il faut donc modifier en conséquence les différents scripts, ainsi que récupérer les résultats renvoyés par les scripts pour les afficher, ce qui n'était pas le cas avec une exécution dans *R Studio*.

Aussi, les objectifs des deux projets sont différents. Pour SNFreezer, il est de tester des algorithmes et de nouvelles méthodes d'analyse. Pour DisCoCRM, il est d'utiliser des algorithmes déjà éprouvés et de les rendre facilement utilisables par les utilisateurs ciblés.

6.3/ Organisation du projet

Une fois l'architecture spécifiée, l'organisation du projet peut être précisée, en commençant par les différents choix techniques résultant des contraintes liées au développement en entreprise, à savoir une facilité de développement et de maintenance, ainsi que l'utilisation de technologies déjà maîtrisées chez eb-Lab et Teletech International ; puis en découpant le projet en plusieurs phases.

6.3.1/ Environnement technique

Le framework `.NET`[5] est utilisé pour la grande majorité des développements réalisés par eb-Lab et Teletech International. Aussi, la plupart des développeurs présents chez eb-Lab

5. https ://www.microsoft.com/net

et Teletech International maîtrisent ce framework et n'utilisent que celui-ci. Afin de faciliter le développement et la maintenance des applications, DisCoCRM utilisera ce framework. L'environnement technique qui sera utilisé pour les développements sera articulé autour de ce framework.

Pour le développement des applications, nous utilisons `Visual Studio Community 2015`[6] qui est un environnement de développement intégré (IDE), développé par Microsoft. Cet IDE permet d'utiliser le framewok `ASP.NET`[7], développé par Microsoft, afin de développer des applications Web. `Nuget`[8], une extension de `Visual Studio`, est utilisée pour gérer l'installation, la désinstallation et la mise à jour des packages installés sur les différents projets.

Afin de pouvoir travailler en équipe et également dans le souci de réaliser des sauvegardes régulières du travail réalisé, ainsi que de suivre les modifications et les versions des applications, le système de gestion de version `SVN`[9] est employé. Nous utilisons `AnkhSVN`[10], une extension de `Visual Studio`.

Du point de vue infrastructure, nous utilisons `IIS`[11], qui est un serveur Web, développé par Microsoft. Une version *Express* est utilisée par `Visual Studio` lors du déploiement des sites créés pour les tests en local.

Pour le stockage des données nécessaires au fonctionnement et à la sécurité des applications, `SQL Server`[12], qui est un système de gestion de bases de données relationnelles, développé par Microsoft, est utilisé par les applications développées (hors *data warehouse*). Lors du développement, les différentes bases de données spécifiques à chaque projet sont utilisées en local et peuvent être administrées directement depuis `Visual Studio`, ou depuis `SQL Server Management Studio`[13].

Pour la couche d'accès aux données, Microsoft fournit aux développeurs un mappeur objet/relationnel (ORM) afin de manipuler facilement des objets plutôt que d'accéder aux données (tables) par des requêtes et ainsi limiter l'écriture de code permettant l'accès aux données à écrire. Cet ORM, `Entity Framework`[14], offre trois types d'utilisation en fonction des besoins et des contraintes de développement :

1. *Code First*, qui permet de n'écrire que du code, avec des classes et des propriétés qui serviront à générer une base de données ;
2. *Model First*, qui permet de créer un modèle qui générera une base de données ;
3. *Database First*, qui permet de générer un modèle de données avec des classes et des propriétés correspondant aux tables et colonnes d'une base de données existante.

Nous avons retenu l'approche *Code First*, qui présente dans notre cas plusieurs avantages :

6. https ://www.visualstudio.com/
7. https ://www.asp.net/
8. https ://www.nuget.org/
9. https ://subversion.apache.org/
10. https ://ankhsvn.open.collab.net/
11. Internet Information Services, https ://www.iis.net/
12. https ://www.microsoft.com/en-us/cloud-platform/sql-server
13. https ://msdn.microsoft.com/fr-fr/library/mt238290.aspx
14. https ://msdn.microsoft.com/fr-fr/data/ef.aspx

— *Code First Migrations* permet, lorsque le modèle de données change, d'automatiser le déploiement des changements du schéma de la base de données vers les sites de production. Les deux autres approches n'offrent pas cette fonctionnalité et les changements doivent être gérés par les développeurs ;
— il n'y a aucun code généré automatiquement et il est difficile à modifier si besoin, ce qui permet un contrôle complet du code de l'application ;
— la logique métier est contenue uniquement dans le code hébergé par le serveur d'application, et non dans la base de données, qui ne sert qu'au stockage ;
— nous n'avons aucune base de données pré-existante.

En conclusion, cette approche nous permet d'être rapidement productifs dans le développement du projet.

Profitant de l'expérience acquise avec SNFreezer, nous nous sommes concentré dans un premier temps, sur la spécification et le développement du connecteur Twitter. Afin de gérer la communication avec les différentes APIs mises à disposition par Twitter, nous utilisons le driver `C# Tweetinvi` [15]. Il permet d'établir la connexion et d'envoyer une requête à Twitter afin d'obtenir les données, converties ensuite en objets pouvant être manipulés par le langage `C#`.

Afin de gérer la *Stream API* de Twitter, nous utilisons `SignalR` [16], qui est une librairie pour `ASP.NET`, développée par Microsoft, implémentant le standard `WebSocket`. Il permet de gérer la communication entre des clients et des serveurs. `SignalR` propose une alternative, via un système de *push*, au fonctionnement classique entre un client et un serveur. Ce dernier consiste en une succession de requêtes du client et de réponses du serveur et est inadapté à une gestion en temps réel des dialogues client-serveur.

Concernant le système de *push*, un client s'inscrit à une méthode présente sur un serveur, puis le serveur le contacte (une ou plusieurs fois). En effet, lorsque ce dernier possède des données à envoyer, il les poussera aux clients inscrits qui s'occuperont de traiter les données. L'ensemble obtenu est ainsi très réactif ; et utilisé par exemple pour gérer un système de chat, ou des notifications d'événements. Ainsi, avec le système de *push*, l'application permet une véritable gestion en temps réel des dialogues entre clients et serveurs. Pour mettre en place ce système de *push*, la première étape consiste à créer un `Hub`, qui est la partie visible du serveur. Il permet d'exposer aux clients les méthodes qu'ils peuvent appeler ou auxquelles ils peuvent s'abonner. Dans le cas d'un chat, le client a deux actions à faire : s'abonner à la méthode lors de sa connexion pour pouvoir recevoir des données ; et, quand il veut communiquer, appeler la bonne méthode en incluant le message qu'il souhaite transmettre. Ainsi, lorsque `SignalR` reçoit le message, il peut le renvoyer à tous les clients connectés. On peut aussi choisir de le transmettre à :

— tous les abonnés ;
— tout le monde sauf l'appelant ;
— seulement l'appelant.

Ceci est très utile dans notre cas pour transmettre et savoir où enregistrer les tweets récoltés.

L'architecture mise en place pour collecter des données en direct, par exemple via la *Stream API* de Twitter, est reprise dans le schéma 6.6. Il reprend le `Hub` présenté plus tôt,

15. https ://tweetinvi.codeplex.com/
16. http://www.asp.net/signalr : technologie de Microsoft pour le développement de fonctionnalités en temps réel pour une application `ASP.NET`

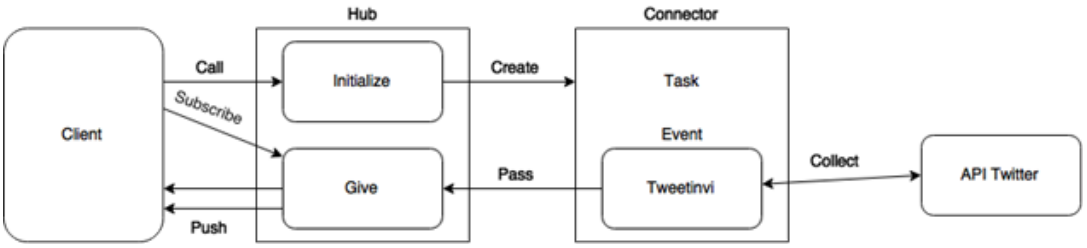

FIGURE 6.6 – Fonctionnement de `SignalR` au sein de la `WebStreamDataProvider`

mais s'intéresse aussi au cheminement complet de l'information. Tout d'abord, le client appelle la méthode `Initialize` du `Hub`, qui crée une tâche dans le `ConnectorTwitter` représentant le flux. Le client s'abonne aussi au `Hub` afin de recevoir les tweets collectés. Ensuite, le driver `Tweetinvi` se charge de communiquer avec l'API de Twitter. Lorsqu'un tweet est reçu, c'est-à-dire que la *StreamAPI* nous renvoie un tweet correspondant aux critères définis, `Tweetinvi` crée un événement, qui sera passé à `SignalR`. Ce dernier *pushera* le tweet au client abonné.

`SignalR` nous permet donc de créer une communication en temps réel entre la *StreamAPI* de Twitter et un client qui se chargera de traiter les données collectées.

6.3.2/ PHASES DU PROJET

À partir des spécifications techniques et des spécifications fonctionnelles, nous avons identifié trois grandes phases, chaque phase pouvant être développée indépendamment des autres et pouvant donner lieu à des phases de recette indépendantes.

L'application de collecte des données, `WebStreamDataProvider` sur la figure 6.5, qui sera utilisée par DisCoCRM ainsi que d'autres applications développées par eb-Lab et Teletech International, sera développée en premier. En effet, il est indispensable de la développer en premier étant donné que les autres parties de DisCoCRM l'utilisent.

L'entrepôt de données permettant de stocker les données collectées grâce à la `WebStreamDataProvider` sera développé dans un second temps.

Enfin, l'application Web de contrôle, ainsi que l'intégration des scripts d'analyse et outils de visualisation des résultats seront développés en dernier.

La figure 6.7 présente les différentes briques logicielles qui composent la plateforme DisCoCRM ainsi que les interactions principales entre ces briques.

Dans la suite de ce chapitre seront détaillées les briques développées pour mettre en place l'outil d'analyse de données issues des réseaux sociaux :

1. un outil pour collecter les données (`WebStreamDataProvider`) ;
2. un entrepôt de données (flexible et scalable) pour stocker les données (figure 6.5) ;
3. l'intégration d'outils d'analyse et algorithmes déjà existants ;
4. la création d'interfaces pour interagir avec les différentes briques (Application Web de contrôle).

L'analyse et la conception visant à mettre en place l'ensemble de ces modules ainsi que les différents choix techniques sont présentées ci-après.

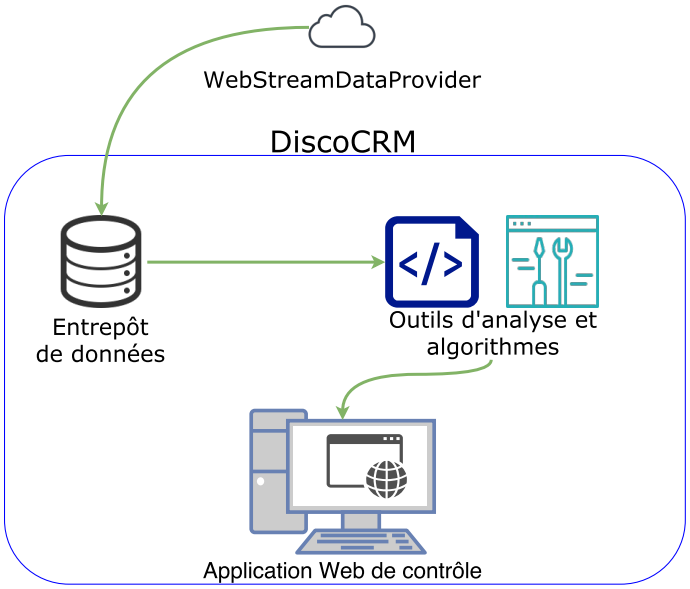

FIGURE 6.7 – Briques logicielles de DisCoCRM et interactions

6.4/ OUTIL DE COLLECTE DE DONNÉES

L'outil de collecte de données est la première brique qui a été conçue et développée dans la plateforme. En effet, cette application est indispensable au bon fonctionnement des outils et algorithmes d'analyse, et elle est aussi utilisée par un autre projet développé chez eb-Lab, l'équivalent de *Social Buddies* sur Twitter.

Le module de collecte de données répond à deux exigences :

1. être accessible par plusieurs applications ;
2. pouvoir intégrer plusieurs sources de données (Twitter, Facebook, etc.) et offrir la possibilité de rajouter des sources de données sans remettre en cause son architecture.

L'architecture générale de l'outil de collecte est composée de trois modules : 1) un Web service de collecte de données, 2) une base de données interne et 3) un Web service pour l'authentification.

6.4.1/ RÉALISATION D'UN WEB SERVICE DE COLLECTE DE TWEETS

Aucun outil de collecte de tweets n'avait été développé par eb-Lab. Il a été envisagé d'utiliser SNFreezer, mais celui-ci comporte plusieurs limitations générales, pour une intégration dans un environnement de production, que nous avons déjà évoquées dans la partie 5.8, ainsi que des limitations techniques pour une intégration au sein d'eb-Lab. En effet, SNFreezer a été développé en `PHP`, et aucun développeur des équipes d'eb-Lab n'a de compétence dans ce langage puisque la grande majorité des développements actuels utilisent le framework `.NET` avec le langage `C#`. Ainsi, l'évolutivité du code de SNFreezer

est limitée et sa maintenance impacterait fortement les méthodologies de développement mises en place au sein d'eb-Lab.

Nous avons donc choisi de réaliser une application sous forme de Web service grâce au framework `ASP.NET Web API` [17] permettant de concevoir des Web service `RESTful` [18]. Nous avons appelé cette application `WebStreamDataProvider`, et elle correspond à la *couche d'accès* visible sur la figure 6.5.

Le développement sous forme de Web service permet de fournir un seul point d'accès pour les différentes applications de Teletech International et d'eb-Lab, permettant ainsi une cohérence dans le traitement des données, et une facilité accrue de développement d'applications distribuées, d'utilisation et de maintenance.

L'accès à ce Web service est sécurisé, avec l'utilisation de comptes pouvant être révoqués. Il est aussi utilisable par des clients non développés avec le framework `.NET`, grâce à des échanges standardisés de type `XML` et/ou `JSON` entre le Web service et les différentes applications l'utilisant.

Afin d'assurer sa réutilisabilité et de pouvoir ajouter ou modifier de nouvelles sources de données sans impacter les fonctionnalités déjà développées, le Web service a été conçu de manière la plus générique possible et les différentes fonctionnalités ont été découplées au maximum.

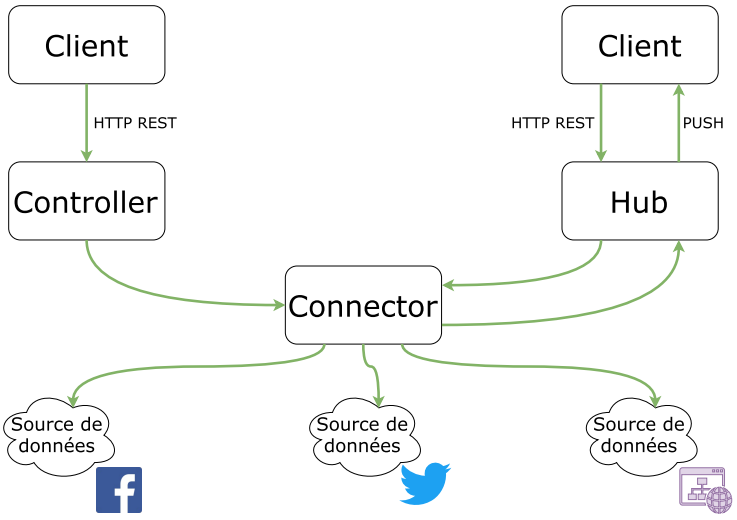

FIGURE 6.8 – Schéma global de la `WebStreamDataProvider` de DisCoCRM

Le schéma 6.8 présente une vue globale de la `WebStreamDataProvider` qui se conforme à un découpage classique d'une application `ASP.NET Web API`. Il y a deux manières d'interagir avec cette application.

La première manière consiste pour un client à faire appel à une méthode du `Controller` via une requête `HTTP`. Ce dernier va se charger d'effectuer les traitements afin de lui répondre. Dans le but d'intégrer n'importe quelle source de données (`Data Source` sur le schéma), nous avons développé un module supplémentaire, nommé `Connector`, permettant de récupérer les flux, soit depuis la totalité des sources implémentées, soit seulement

17. http://www.asp.net/web-api
18. Un Web service `RESTful` fait appel à des requêtes `HTTP` pour obtenir (`GET`), placer (`PUT`), publier (`POST`) et supprimer (`DELETE`) des données

depuis celles demandées par le client.

De plus, afin d'intégrer la *Stream API* de Twitter, nous avons conçu une deuxième manière d'interagir avec l'application grâce à un `Hub` en utilisant la technologie `Signal R`. Un client s'enregistre sur le `Hub`, et définit, au moyen de requêtes `HTTP`, ses critères de sélections à suivre (dans notre cas des comptes, hahstags, etc.). De la même manière que le `Controller`, ce `Hub` interagit avec le module `Connector` afin de contacter les différentes sources de données souhaitées.

Le module `Connector` est le cœur de cette application. En effet, il encapsule les différentes sources de données et réalise l'abstraction de leurs APIs. Ce module a donc été développé de façon à ce que le `Controller` ou le `Hub` n'aient qu'un appel à faire pour pouvoir récupérer les données depuis toutes les sources. Son architecture est exposée par le schéma 6.9.

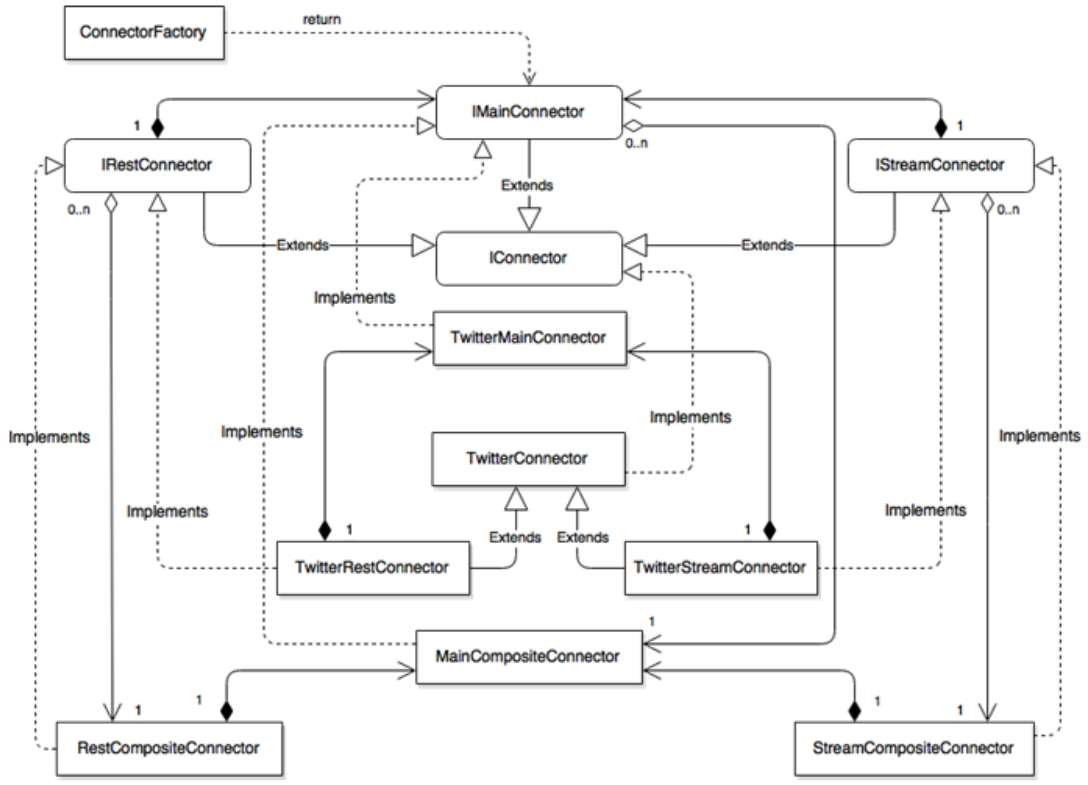

FIGURE 6.9 – Schéma du module *Connector* de la `WebStreamDataProvider`

Du point de vue de l'implémentation, plusieurs interfaces et classes structurent l'application (figure 6.9). Elles sont divisées en trois groupes :

1. Les classes et interfaces contenant les méthodes de récupération de données «simples»
2. Les classes et interfaces contenant les méthodes de récupération de données «avancées»
3. Les classes et interfaces où l'on retrouve les méthodes communes aux entités `Rest` et `Stream`

Les interfaces (dont le nom commence par un `I`) permettent de définir les méthodes utilisables par le `Controller` et le `Hub`, mais aussi celles à implémenter dans les

différents modules permettant la communication avec les sources de données. Il s'agit, sur la figure 6.9, des interfaces : `IConnector`, `IRestConnector`, `IStreamConnector`, `IMainConnector`.

Concernant les méthodes de récupération de données « simples », le nombre de résultats est fini et connu au moment de l'exécution. Ces entités correspondent à celles contenant le mot `Rest` dans leur nom et sont utilisables via le `Controller` par des requêtes et réponses `HTTP`.

Concernant les méthodes de récupération de données « avancées », le nombre de résultats n'est pas connu au moment de l'exécution. En effet, il dépend des critères de sélection définis, mais aussi de leur pertinence pour correspondre à des flux réels futurs. Ces entités correspondent à celles contenant le mot `Stream` et sont utilisables via le `Hub` par des requêtes `HTTP` afin de définir les critères de sélection. Lorsqu'un flux correspondant à ces critères est détecté, un événement est remonté jusqu'au `Hub` afin de pouvoir, grâce à la technologie `SignalR`, envoyer une réponse `HTTP`, contenant entre autres le message, en *push*, du serveur vers le client.

Les entités permettant de manipuler les différentes sources de données doivent donc implémenter les interfaces précédemment mentionnées afin de pouvoir être utilisées convenablement. Ainsi, pour Twitter qui est actuellement la seule source de données implémentée dans l'application, on dispose de quatre entités implémentant chacune une des interfaces.

Le troisième groupe est la partie *centrale* correspondant aux méthodes communes aux entités `Rest` et `Stream`. Elle contient aussi un objet représentant ces dernières et permet d'accéder à ces méthodes. Ces entités centrales, et plus particulièrement l'interface `IMainConnector`, sont les seules manipulées par le `Controller` et le `Hub`. En effet, lorsque le `Controller` ou le `Hub` reçoit une demande d'un client concernant une ou toutes les sources disponibles, un appel vers la classe `ConnectorFactory` va permettre de créer un objet `MainConnectorComposite`. Cet objet implémente l'interface `IMainConnector` et est utilisable en tant que tel. Il contient une liste d'entités implémentant cette interface, correspondant à la liste des sources demandées, et permet d'utiliser cette liste comme un seul et même objet.

Le `MainConnectorComposite` donne accès à deux entités : `RestConnectorComposite` et `StreamConnectorComposite`, permettant de manipuler les interfaces `IRestConnector` et `IStreamConnector` de chaque source comme un seul et même objet de la même manière que pour l'interface `IMainConnector`.

Cependant, la construction et l'utilisation de cette structure peut s'avérer coûteuse en ressources serveur et entraîner des ralentissements dans les réponses aux différents clients. Afin de limiter ce problème, nous avons mis en place un *Lazy-Loading* pour le chargement des `MainConnector` correspondant à chaque source de données. Cette technique permet de ne charger entièrement l'objet que lorsque l'on en a réellement besoin. Cela améliore grandement les performances, que ce soit au niveau des ressources nécessaires ou des temps de chargement.

6.4.2/ BASE DE DONNÉES INTERNE

La `WebStreamDataProvider` n'est utilisée que pour collecter les données issues de plusieurs sources, et non pour les stocker. Elle utilise la base de données interne de Dis-

CoCRM (figure 6.5) afin de réguler et faciliter son utilisation. Le schéma 6.10 illustre la modélisation de cette base.

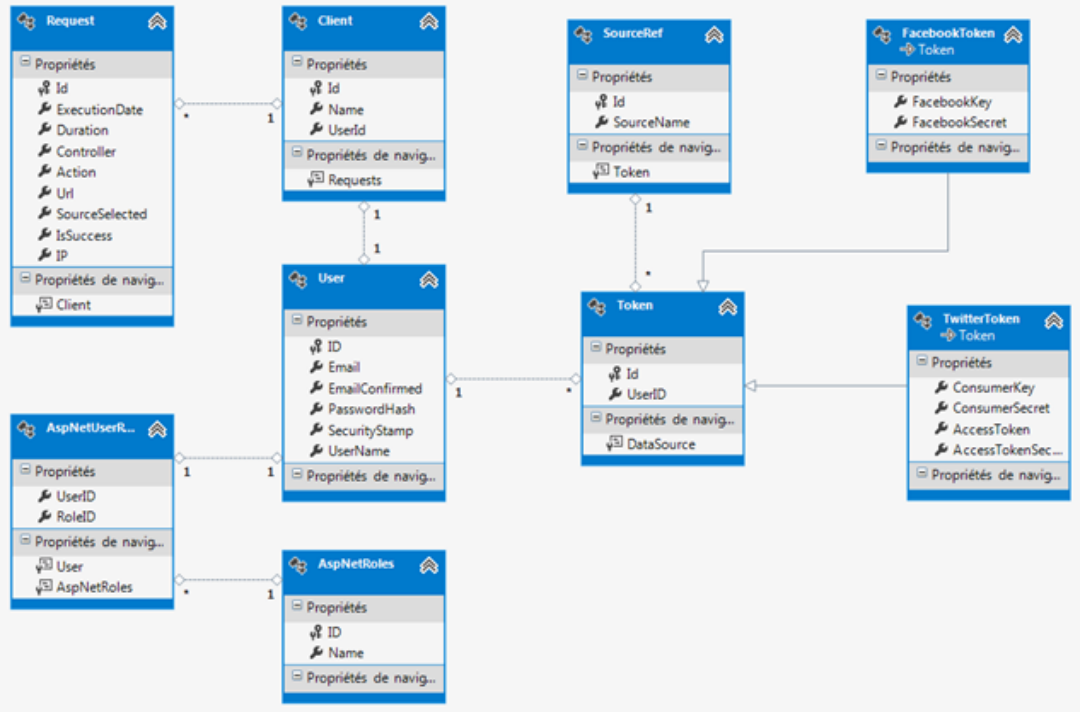

FIGURE 6.10 – Modélisation de la base de données locale de la `WebStreamDataProvider`

Cette base de données permet de stocker des informations sur les utilisateurs de l'application ainsi que sur leurs droits d'accès sur les différentes sources de données. En effet, ces dernières demandent des jetons d'accès, appelés *tokens* et pouvant être obtenus par n'importe quel utilisateur, afin de pouvoir collecter des données. Les sources de données offrent aussi la possibilité de restreindre les droits liés à des *tokens*, par exemple en autorisant la lecture de données mais pas l'écriture.

Chaque source de données proposant sa propre représentation des *tokens*, il est nécessaire d'implémenter une table pour chacune des sources, ici `TwitterToken` et `FacebookToken`. Cependant, et toujours dans un souci de généricité et de facilité d'utilisation, toutes ces tables héritent de la même entité `Token`, afin que les *tokens* des différentes sources soient manipulés de la même manière dans l'application. Enfin, la table `SourceRef` gère les jetons d'accès permettant d'associer un *token* à une source de données.

Pour la gestion des utilisateurs, nous avons utilisé les outils mis à disposition par le framework `ASP.NET` permettant l'enregistrement et l'authentification des utilisateurs sur l'application comme décrit dans la partie 6.4.3. De plus, ce framework fournit une gestion des rôles permettant de spécifier facilement les autorisations des utilisateurs et les ressources auxquelles ils ont accès. Cependant, cette gestion des rôles n'est pas implémentée par défaut. Nous avons permis la définition de rôles et l'affectation de ces derniers aux utilisateurs via les tables `AspNetRoles` et `AspNetUserRoles`.

Afin d'éviter les utilisations abusives et en trop grand nombre de la `WebStreamDataProvider`, un système d'enregistrement des requêtes effectuées par un client a été mis en place, via la table `Request`. Il permet de limiter le nombre

de requêtes possibles par un même client sur une période donnée. Il permet aussi de tracer, grâce à l'adresse IP du client, les requêtes ou encore la ressource demandée afin de faciliter la maintenance en cas de problème, mais aussi d'assurer la sécurité en vérifiant la provenance des requêtes pour éviter les usurpations de compte.

6.4.3/ GESTION DE L'AUTHENTIFICATION

L'authentification à la `WebStreamDataProvider` est réalisée à travers un système de *tokens*. En effet la communication se faisant en `HTTP`, la meilleure manière pour sécuriser les connections est l'emploi de *tokens*. La mise en place de l'authentification nécessite plusieurs étapes et utilise plusieurs paquets :

1. `OWIN` [19], qui permet de définir une couche d'abstraction entre un serveur et une application `ASP.NET`, afin de découpler le serveur de l'application. Cela permet de pouvoir porter une application d'un serveur pour l'héberger vers un autre type de serveur. Une application écrite en respectant les standards définis par `OWIN` ne sera pas fortement couplée à `IIS` comme cela est le cas avec les applications `ASP.NET` classiques.
2. `Microsoft.Owin`, qui implémente la spécification `OWIN`. Ce projet, soutenu par Microsoft, vise à faciliter l'utilisation de librairies et d'applications compatibles avec `OWIN` sur la plateforme Windows. Il permet la gestion et le support du protocole d'authentification `OAuth2` [20], qui est utilisé pour les comptes utilisateurs de la `WebStreamDataProvider`.
3. `Microsoft ASP.NET Identity Owin`, qui permet d'affiner la gestion des processus de création et d'enregistrement de comptes utilisateur. Il permet d'avoir le contrôle sur les comptes stockés dans l'application, afin de choisir les informations que l'on souhaite enregistrer.

Création d'un compte utilisateur

Afin de pouvoir accéder aux fonctions proposées par la `WebStreamDataProvider`, il est nécessaire de créer un compte qui sera enregistré dans la base de données interne de la `WebStreamDataProvider`. Pour cela, il suffit d'appeler la méthode `Register` qui se trouve dans le `Controller Account`.

Le listing 6.1 décrit un exemple de requête `HTTP` permettant de créer un compte utilisateur. Le *body* de l'appel `HTTP`, en rouge dans cet exemple, doit respecter la syntaxe des méthodes `POST` et contenir trois éléments :

1. le nom de l'utilisateur, ici `alice@example.com` ;
2. son mot de passe, ici `Password":"Password1!` ;
3. la confirmation de celui-ci, ici `ConfirmPassword":"Password1!`.

Listing 6.1 – Exemple de création de compte utilisateur

```
1 POST https://localhost:49269/api/Account/Register HTTP/1.1
```

19. http://owin.org/
20. https://oauth.net/2/ ; protocole permettant d'autoriser une application, un site Web ou un logiciel à accéder à une API d'un autre site Web

```
Host: localhost:49269
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; rv:32.0) Gecko/20100101 Firefox
    /32.0
Accept: */*
Content-Type: application/json; charset=utf-8
X-Requested-With: XMLHttpRequest
Referer: https://localhost:49269/
Content-Length: 84

{"Email":"alice@example.com","Password":"Password1!","ConfirmPassword":"
    Password1!"}
```

Demande et utilisation d'un *token* d'accès

Une fois le compte utilisateur créé, ce dernier doit demander un *token* d'accès afin d'être reconnu par l'application sans avoir à fournir ses identifiants à chaque action. Pour cela, il suffit d'envoyer une requête HTTP POST à l'adresse https://adressedelapplication/Token en ajoutant ses identifiants dans le *body* de l'appel HTTP.

Listing 6.2 – Exemple de demande de *token* d'accès

```
POST https://localhost:49269/Token HTTP/1.1
Host: localhost:49269
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; rv:32.0) Gecko/20100101 Firefox
    /32.0
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Referer: https://localhost:49269/
Content-Length: 68
grant_type=password&username=alice@example.com&password=Password1!
```

Le listing 6.2 est un exemple de requête HTTP permettant la demande d'un *token*. Cette requête contient les identifiants de l'utilisateur, dans notre exemple `alice@example.com` et `Password1!`. Il est donc préférable d'utiliser le protocole HTTPS afin d'assurer la sécurité[21]. Si la requête est complète et que les identifiants de l'utilisateur sont valides, le serveur renvoie une réponse HTTP contenant le *token* d'accès.

Listing 6.3 – Exemple de réponse à une demande de *token* d'accès

```
HTTP/1.1 200 OK
Content-Length: 669
Content-Type: application/json;charset=UTF-8
Server: Microsoft-IIS/8.0
Date: Wed, 01 Oct 2014 01:22:36 GMT

{
"access_token":"imSXTs2OqSrGWzsFQhIXziFCO3rF...",
"token_type":"bearer",
"expires_in":1209599,
"userName":"alice@example.com",
".issued":"Wed, 01 Oct 2014 01:22:33 GMT",
".expires":"Wed, 15 Oct 2014 01:22:33 GMT"
}
```

Le listing 6.3 montre le contenu de la réponse au format JSON, permettant de la réutiliser facilement dans n'importe quelle application. Plusieurs informations sont disponibles :

21. Pour un fonctionnement `stateless` HTTPS est suffisant ; dans le sens où l'adresse, les paramètres de la requête et son contenu (que ce soit en GET ou POST) sont cryptés après la négociation faite entre le client et le serveur

— le *token* d'accès ;
— son type ;
— sa durée de vie ;
— l'utilisateur l'ayant demandé.

Le type de *token*, ainsi que sa durée de vie sont paramétrables par le développeur dans le fichier `App_Start/Startup.Auth.cs`.

Une fois le *token* d'accès récupéré, l'utilisateur peut accéder aux fonctionnalités de l'application nécessitant une autorisation. Pour cela, il lui suffit d'ajouter son *token* d'accès dans le champ `Authorization` du `Header` de sa requête `HTTP`, comme le montre le listing 6.4.

Listing 6.4 – Exemple de demande d'accès à une ressource

```
1 GET https://localhost:49269/api/stream HTTP/1.1
2 Host: localhost:49269
3 User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; rv:32.0) Gecko/20100101 Firefox
      /32.0
4 Accept: */*
5 Authorization: Bearer imSXTs2OqSrGWzsFQhIXziFCO3rF...
6 X-Requested-With: XMLHttpRequest
```

Dans ce listing, l'utilisateur tente d'accéder à une ressource nécessitant une autorisation (`api/stream`). Il a donc renseigné son *token* précédé de `Bearer`, qui est le type d'authentification utilisé dans notre application.

6.5/ ENTREPÔT DE DONNÉES

Le stockage des données collectées n'est pas géré par la `WebStreamDataProvider`, mais par chaque application utilisant le Web service. Ce découplage permet de choisir la méthode de stockage la plus appropriée selon les besoins. Nous avons étudié les différentes contraintes de DisCoCRM liées au stockage des données collectées depuis les réseaux sociaux afin, dans un premier temps de choisir la solution la plus adaptée aux projets d'eb-Lab, tout en laissant ensuite la possibilité d'implanter un stockage polyglotte, rendu possible par le la couplage faible.

6.5.1/ CONTRAINTES DE L'ENTREPÔT DE DONNÉES

Dans le cas de notre plateforme, nous devons faire face à plusieurs contraintes :

— le stockage des données joue un rôle primordial puisqu'il est au centre de l'application, stockant les données issues des réseaux sociaux, et obtenues via le Web service décrit dans la partie 6.4, afin que les outils d'analyse puissent en extraire de la connaissance ;
— nous devons traiter des données à grandes dimensions, en volume et en variété ;
— les temps de réponse doivent être raisonnables ;
— la compatibilité avec une application `.NET`, et avec le langage utilisé pour implémenter les outils et algorithmes d'analyse, `R`, est importante, via l'existence de drivers ;
— la maintenance applicative doit pouvoir être assurée par les équipes d'eb-Lab et de Teletech International.

6.5.2/ Choix du système de stockage

Les bases de données relationnelles classiques sont mal adaptées à ces contraintes. La structure de la base étant assez rigide, la modifier devient compliqué et coûteux. Pour réaliser l'entrepôt dont l'application a besoin, il a été décidé de se tourner vers une solution **NoSQL** Les bases de données NoSQL présentent l'avantage d'avoir une structure beaucoup plus flexible, s'affranchissant des contraintes des bases relationnelles tout en ayant une meilleure scalabilité. En effet, elles sont plus souples et mieux adaptées aux données sociales hétérogènes que ne le sont les bases de données relationnelles classiques. Cependant, il existe plusieurs catégories de bases NoSQL :

— clé-valeur ;
— orientées colonnes ;
— orientées documents ;
— orientées graphes.

Pour chaque catégorie de base NoSQL, il existe des implémentations différentes, avec chacune son langage de requête et ses APIs spécifiques [Khazaei et al., 2016]. L'équipe de développement d'eb-Lab et de Teletech International possède peu de compétences dans les bases NoSQL. Cependant, la phase de développement des applications utilisant cette base de données permettra de faire monter en compétences plusieurs développeurs. Suite à l'étude des différentes offres, en tenant compte de la communauté utilisant le système, de sa vitalité, et de ses caractéristiques techniques, il a été retenu deux systèmes de gestion de base de données NoSQL :

1. MongoDB (orienté documents) ;
2. Apache Cassandra (orienté colonnes).

Après avoir testé les deux systèmes en prenant en compte les contraintes définies dans la partie 6.5.1, l'orientation choisie a été MongoDB pour les raisons suivantes :

— MongoDB utilise des documents sous format `JSON` sans schéma prédéterminé, c'est-à-dire le même format que celui utilisé par la `WebStreamDataProvider` pour la collecte de tweets ; alors que Apache Cassandra utilise un type spécifique (`CQL313`) ;
— MongoDB possède de nombreux drivers pour faciliter son intégration dans une infrastructure de système d'information, contrairement à Apache Cassandra qui fige plus le type d'interaction entre composants ;
— MongoDB est la base NoSQL la plus utilisée [22], avec une documentation complète et une communauté active.

6.5.3/ Conception de l'entrepôt de données

Dans la plateforme DisCoCRM, les collectes de données issues des réseaux sociaux sont organisées sous forme de campagnes. La base de données MongoDB est donc organisée de la même manière. Ainsi, chaque campagne possède sa propre base de données, ce qui permet d'éviter le mélange des données entre les différentes campagnes et les différents clients. Toutes les bases de données possèdent les mêmes collections [23], illustrées sur la figure 6.11. Ces collections sont :

22. http://db-engines.com/en/ranking_trend
23. MongoDB est un système de gestion de bases de données NoSQL orienté documents, composé d'un ensemble de bases de données indépendantes entre elles et agissant comme n'importe quelles autres

1. `MainStreamData`, contenant tous les tweets collectés ;
2. `Language`, contenant toutes les langues des tweets collectés (inférées par Twitter) ;
3. `QuerySource`, contenant les mots requêtés ainsi que les comptes suivis.

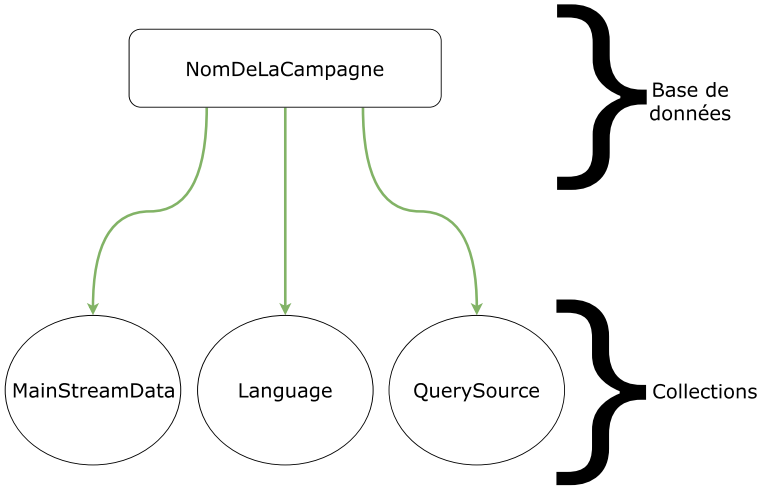

FIGURE 6.11 – Database et collections dans MongoDB

Il est ainsi possible de mettre en place une généralisation des traitements puisque toutes les campagnes possèdent des collections identiques. Par exemple, pour une requête sur les langues, il suffit d'indiquer le nom de la campagne afin d'accéder à la bonne base de données, puis de requêter la collection `Language`.

6.5.4/ SCHÉMA DES SOURCES DE DONNÉES

Le listing 6.5 montre la structure des documents qui nous sont renvoyés par la `WebStreamDataProvider` et qui seront stockés dans la base de données MongoDB. Cet exemple concerne un tweet collecté via la *StreamAPI*.

Listing 6.5 – Structure des documents renvoyés par la `WebStreamDataProvider` pour Twitter

```
1 {
2 "_id" : ObjectId("55b9f13532615b13e46fb0c3"),
3 "ID" : "626688977467117568",
4 "Message" : "Coca-Cola Consolidates Its Leadership,
5 Articles | THISDAY LIVE: http://t.co/7BJoDqujj4",
6 "PublicationDate" : ISODate("2015-07-30T09:41:10Z"),
```

bases de données dans un SGBD relationnel classique. À l'intérieur de celles-ci, on retrouve des *collections*, que l'on peut comparer à des tables dans un SGBD relationnel, une collection étant un ensemble de documents de même nature. Les collections vont contenir l'ensemble des données et les différentes opérations seront effectuées sur elles [Sadalage et al., 2012].

La différence majeure avec les SGBD relationnels réside dans le fait qu'il n'y ait pas de schéma prédéfini pour une collection : on peut y stocker ce que l'on souhaite sans avoir des attributs à remplir obligatoirement. Par exemple, dans une collection *Personne*, il est par exemple possible d'avoir des professeurs et des étudiants ; alors que dans un SGBD relationnel, il est d'usage de créer une table *Personne* et deux tables *Étudiant* et *Professeur* qui en héritent.

```
7  CreatedBy" : "BPI Solutions",
8  "ScreenName" : "BPIsolutions",
9  "ProfileImage" : "http://pbs.twimg.com/profile_images/57002038375
       5997184/8TE4esz-_normal.png",
10 "RetweetCount" : 0,
11 "FavoriteCount" : 0,
12 "Lang" : "English",
13 "InReplyTo" : null,
14 "InRetweetTo" : null,
15 "Hashtags" : [],
16 "URLs" : ["http://www.thisdaylive.com/articles/coca-cola-
       consolidates-its- leadership/215987/#.VbnxHKNOffU.twitter"],
17 "UsersMentions" : [],
18 "Symbols" : [],
19 "QuerySources" : ["coca"],
20 "Medias" : null,
21 "APIFlag" : "StreamAPI",
22 "timestamp" : 1438249270
23 }
```

À part le dernier champ (`timestamp`), tous les autres sont issus de Twitter. On retrouve par exemple l'identifiant du tweet, sa date, son créateur, sa langue, le nombre de retweet, etc. Chaque champ correspond à un besoin utile, que ce soit pour les outils et algorithmes d'analyse ou l'interface client.

6.6/ Intégration des algorithmes et des outils d'analyse

Plusieurs algorithmes d'analyse pour la détection d'événements et de communautés ont été développés et validés dans le cadre d'analyses sur des données collectées avec SNFreezer. Cependant, leur intégration dans DisCoCRM est différente de celle de SNFreezer. En effet, ils doivent pouvoir être lancés depuis l'application Web servant d'interface pour la plateforme. Ce fonctionnement diffère complètement de celui de SNFreezer où la plupart des scripts sont exécutés manuellement depuis *R Studio*. Il est impossible d'utiliser *R Studio* avec des applications Web `ASP .NET`.

La solution retenue pour les tests est d'utiliser `R.NET`[24], librairie permettant d'accéder aux fonctionnalités du langage `R` dans une application utilisant le framework `.NET`.

Liens avec les autres briques et les outils de visualisation

Afin de pouvoir réaliser les analyses demandées par le client, les outils et algorithmes d'analyse doivent accéder aux données stockées dans la base MongoDB.

Cela peut se faire de deux manières :

1. L'application Web requête la base de données MongoDB, puis transmet les données à des scripts `R` implémentant les algorithmes d'analyse. Le problème de cette solution vient de la nécessité de transformer les données `JSON` extraites

24. https://rdotnet.codeplex.com/

de la base MongoDB en objet C#, puis de transformer cet objet en données compréhensibles par R.

2. La seconde solution est que les scripts R, implémentant les algorithmes d'analyse, accèdent eux-mêmes aux données stockées dans la base de données MongoDB. Cela évite des intermédiaires et des transformations de données ralentissant le processus.

MongoDB possédant une large communauté, il existe un package permettant à un programme écrit en R d'accéder à une base MongoDB. On peut voir les liens entre les scripts R implémentant les algorithmes d'analyse, la base de données MongoDB et l'application Web de contrôle dans la figure 6.12.

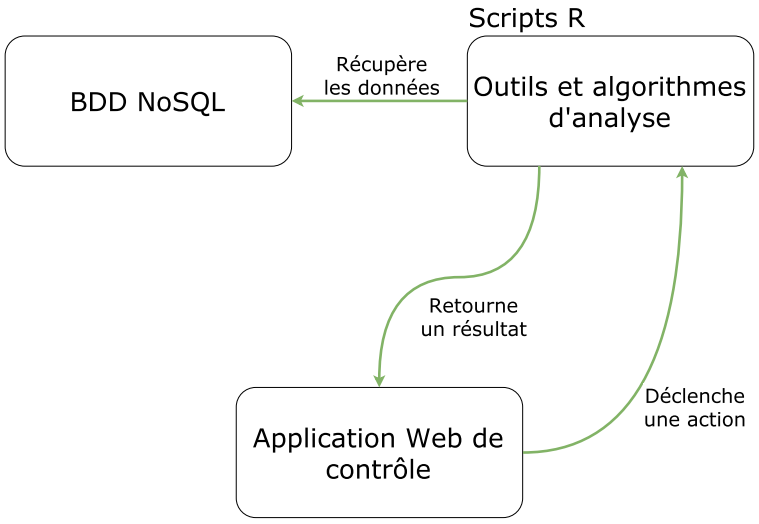

FIGURE 6.12 – Liens entre les scripts et les autres briques

Le client, à travers l'interface de l'application Web, va appeler un ou plusieurs scripts R. Ces derniers vont établir une connexion avec la base de données MongoDB afin de requêter les données. Ensuite, le ou les scripts R exécutent l'algorithme d'analyse et retournent à l'application Web le résultat calculé afin qu'il soit affiché.

Outils de visualisation

L'affichage des résultats étant une partie importante d'un produit commercialisé, il doit permettre de visualiser les informations importantes, ainsi que de parcourir les résultats et les données contenues dans la base MongoDB.

La librairie JavaScript D3JS[25] a déjà été utilisée dans le cadre de SNFreezer et offre des fonctionnalités intéressantes tout en étant facile à mettre en place et à maintenir. Nous l'avons donc retenue comme outil de visualisation des données et des résultats des analyses.

25. https://d3js.org/

6.7/ Application Web de contrôle

Afin d'avoir une interface permettant d'accéder aux données stockées dans les différentes bases MongoDB, de gérer la collecte de tweets, les utilisateurs et leurs droits ainsi que lancer des analyses et visualiser les résultats, la mise en place d'une application de contrôle est nécessaire. Cette dernière est destinée à être utilisée par plusieurs catégories d'utilisateurs (chef de projet fonctionnel, *community manager*, *data scientist*) appartenant potentiellement à plusieurs entreprises (Teletech Internationel, eb-Lab ainsi que leurs clients). Une application Web à accès sécurisé est donc une solution idéale comparée à une application client lourd nécessitant d'être déployée sur toutes les machines de tous les utilisateurs.

Du point de vue des contraintes techniques, l'application a été développée afin d'être utilisable sur plusieurs supports (ordinateurs et tablettes principalement) et donc d'avoir un affichage de son contenu qui s'adapte en fonction des capacités du client Web. La solution retenue est l'utilisation du *Responsive Web Design*[26]. Les choix techniques pour l'application découlent des contraintes présentées au début du chapitre.

Les technologies retenues pour le développement de l'application sont :

- `ASP.NET MVC`, framework de développement Web utilisé par eb-Lab et Teletech International
- `SQL Server`, le moteur de base de données utilisé par eb-Lab et Teletech International
- le framework `Bootstrap`[27].

6.7.1/ Architecture de l'application

Le schéma 6.13 présente l'architecture de l'application Web permettant de gérer les campagnes et d'interagir avec les autres briques et modules composant la plateforme DisCoCRM. On retrouve les différents composants du modèle MVC, avec les modèles transformés en `ViewModels`. En effet, ils contiennent les données dont les vues ont besoin pour l'affichage. Ces dernières sont réalisées en `HTML`, et font appel au framework `Bootstrap` et à `JavaScript` pour améliorer les interactions utilisateurs et faciliter le développement.

En bas du schéma on retrouve les différentes classes correspondant chacune à une couche à contacter :

- `BDD locale`, qui gère les actions vers la base de données interne implantée avec `SQL Server` ;
- `BDD NoSQL`, qui communique avec MongoDB ;
- `Client Web API`, qui gère les appels `HTML` pour la collecte des données ;
- `Client R`, qui demande les analyses implémentées grâce au langage `R` et reçoit le résultat.

Ce découpage permet d'apporter une architecture claire et facilement maintenable. Ainsi, si une des couches venait à être changée, par exemple utiliser une autre base NoSQL,

26. technique permettant de réaliser des sites Web dont le contenu visuel s'adapte automatiquement à l'espace disponible sur l'écran.

27. http://getbootstrap.com/ : Framework `CSS`, `HTML` et `JavaScript`, créé par Twitter, permettant de développer rapidement et facilement l'interface d'un site ou application Web, en fournissant des feuilles de style pour des composants standards comme des boutons, des formulaires, etc. Il est aussi *open source*, sous licence MIT, donc modifiable afin d'obtenir le rendu visuel que l'on souhaite.

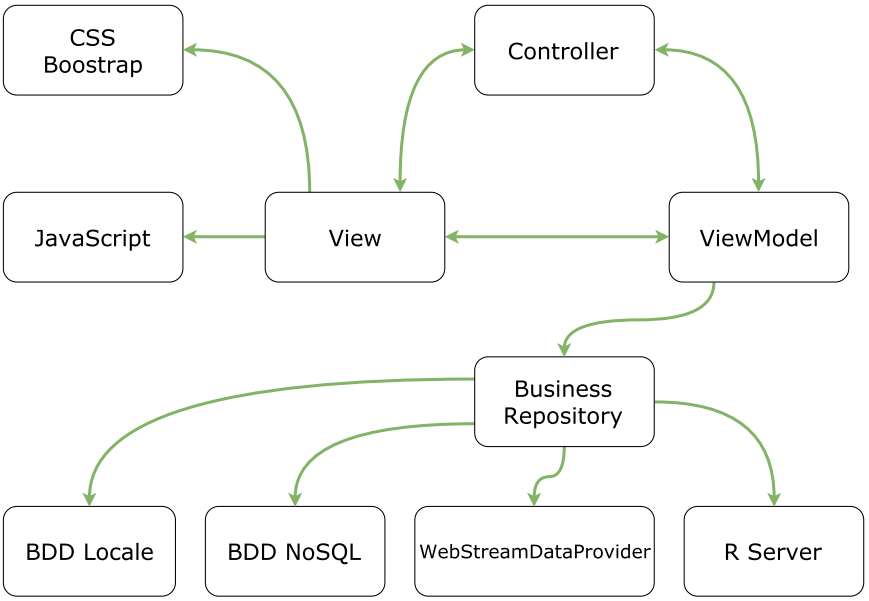

FIGURE 6.13 – Architecture de l'application Web de contrôle

seul le fichier C# correspondant à cette couche, dans notre exemple le fichier BDD NoSQL, devra être modifié et redéployé.

La couche Business Repository permet de faire le lien entre toutes les classes. Par exemple, si une action a besoin d'appeler la WebStreamDataProvider puis de stocker les résultats dans la base de données NoSQL, la couche Business Repository se chargera d'appeler les bonnes méthodes de la couche concernant la WebStreamDataProvider puis de la couche concernant la BDD NoSQL.

Un action du client dans l'application Web déclenche un appel au Controller, qui va construire le ViewModel. Pour cela, le Controller demande au Business Repository de lui transmettre les informations requises. Enfin, le Controller transmet le ViewModel créé, remplit la View et l'affiche.

6.7.2/ BASE DE DONNÉES INTERNE

Les données utilisées par l'application sont en grande majorité stockées dans la base MongoDB. Cependant, afin de coordonner les interactions entre les différentes couches de l'application, une base de données interne et locale à l'application Web doit être réalisée, dont le schéma est visible sur la figure 6.14.

On retrouve les classes standards mises à disposition par le framework .NET pour l'identification des utilisateurs sur le site Internet, c'est-à-dire :

- Users, où se trouvent les identifiants testés lors de la phase de login,
- AspNetRoles et AspNetUserRoles, système de gestion de rôles que l'on peut affecter à un utilisateur.

La partie la plus importante à mettre en place est celle gérant les campagnes. Chaque recherche ou analyse effectuée par un utilisateur est liée à une campagne. Cependant, un utilisateur peut avoir une ou plusieurs campagnes en cours à un instant donné. Il faut ainsi gérer les campagnes, avec leur statut (en cours, terminée, en attente, etc.) ; ainsi qu'une

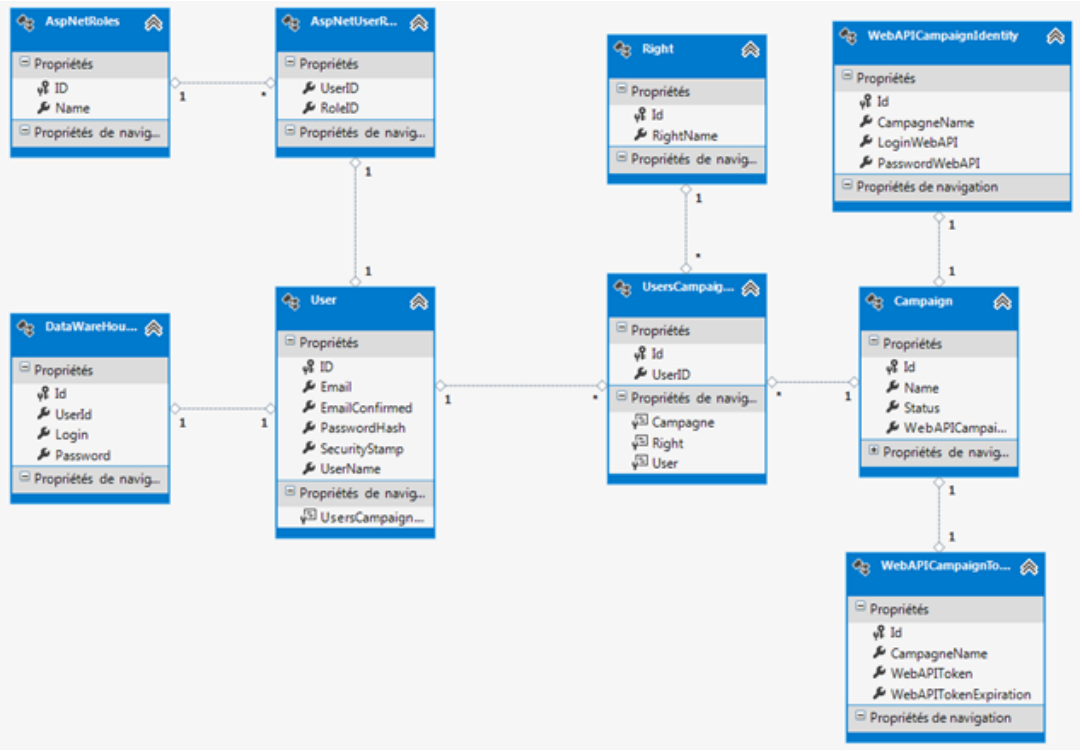

FIGURE 6.14 – Base de données interne de l'application Web de gestion des campagnes

table d'association permettant de connaître les droits d'accès aux différentes campagnes pour tous les utilisateurs (`UserCampaignRight`).

Les campagnes étant inscrites dans la base de données interne de la `WebStreamDataProvider`, nous stockons également le login et le mot de passe nécessaires pour demander le token d'authentification. Ce dernier ayant une durée de validité, nous stockons sa date d'expiration afin d'éviter un appel inutile à la `WebStreamDataProvider` redemandant le token pendant sa durée de validité.

6.7.3/ ACTIONS DE L'UTILISATEUR ET INTERFACE DE L'APPLICATION

L'application Web comporte plusieurs pages avec chacune ses actions spécifiques, permettant aux utilisateurs d'interagir avec les différents modules de la plateforme. Exceptée la page principale et la page de login, toutes les pages concernent une seule campagne à la fois.

Page Principale

La figure 6.15 présente la page principale, qui est le cœur du site. C'est la page s'affichant par défaut lors de la connexion d'un utilisateur. Elle permet d'accéder aux différentes fonctionnalités et aux interfaces de gestion. On peut aussi l'atteindre via le lien intitulé "Vos Campagnes", présent en haut de la page.

L'utilisateur connecté peut visualiser sur cette page l'ensemble des campagnes sur lesquelles il possède un droit. L'affichage de ces campagnes se fait sous forme de tableau, construit à partir de la base de données interne de l'application Web. Ce tableau synthétise les différentes informations sur les campagnes : leur nom et leur statut (en

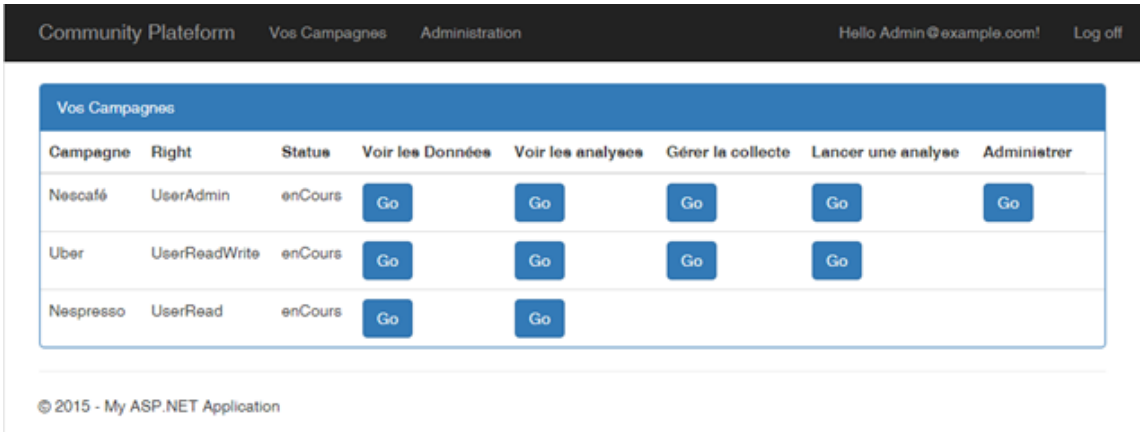

FIGURE 6.15 – Page principale de l'application Web de gestion des campagnes

cours, terminée ou suspendue), ainsi que le droit de l'utilisateur dessus, qui détermine les actions possibles ou non et donc l'apparition des boutons y conduisant. En fonction de ses droits, un utilisateur peut ensuite accéder aux différentes pages liées à une campagne. Les principaux droits sont :

- `UserAdmin` : possède tous les droits sur une campagne ;
- `UserReadWrite` : peut influencer sur toutes les données et leur collecte mais pas sur l'administration de la campagne (gestion des utilisateurs ayant un droit sur cette campagne) ;
- `UserRead` : accède seulement aux données enregistrées ;
- `RevokedUser` : l'utilisateur avait un droit avant mais maintenant n'en a plus aucun : il est indiqué comme `RevokedUser`, pour le retrouver et lui redonner un droit si besoin.

Visualisation des données

Cette page permet de visualiser les différentes données liées à une campagne et stockées dans la base de données MongoDB. Elle est composée de deux parties. Dans la partie supérieure de l'image, on retrouve les filtres et les options de tri pouvant être appliqués :

- `Croissant`, `Décroissant` et `Colonne Name` : classe (`-1` par ordre décroissant, `1` par ordre croissant) les données par rapport à la colonne choisie ;
- `Skip Documents` : n'affiche pas les `n` premiers documents ;
- `Limiter le nb de Documents` : affiche seulement les `n` premiers documents ;
- `User Name` : recherche les données suivant le nom du créateur du tweet ;
- `Date Min` et `Date Max` : donne les données présentes dans l'intervalle défini par les deux dates ;
- `Language` : menu déroulant contenant les langues des données stockées dans la collection MongoDB du même nom ;
- `Query sources` : idem que pour `Language` mais pour les *query sources*.

Le formulaire est complété par deux boutons. Le premier lance la recherche et rafraîchit seulement le tableau résultat, grâce à `JavaScript` qui utilise des requêtes `Ajax`[28] afin de créer des vues partielles qui seront intégrées dans la page `HTML`. Le deuxième bouton

28. *Asynchronous JavaScript and XML* : ensemble de technologies destinées à réaliser de rapides mises à jour du contenu d'une page Web, sans qu'elles nécessitent le moindre rechargement visible par l'utilisateur.

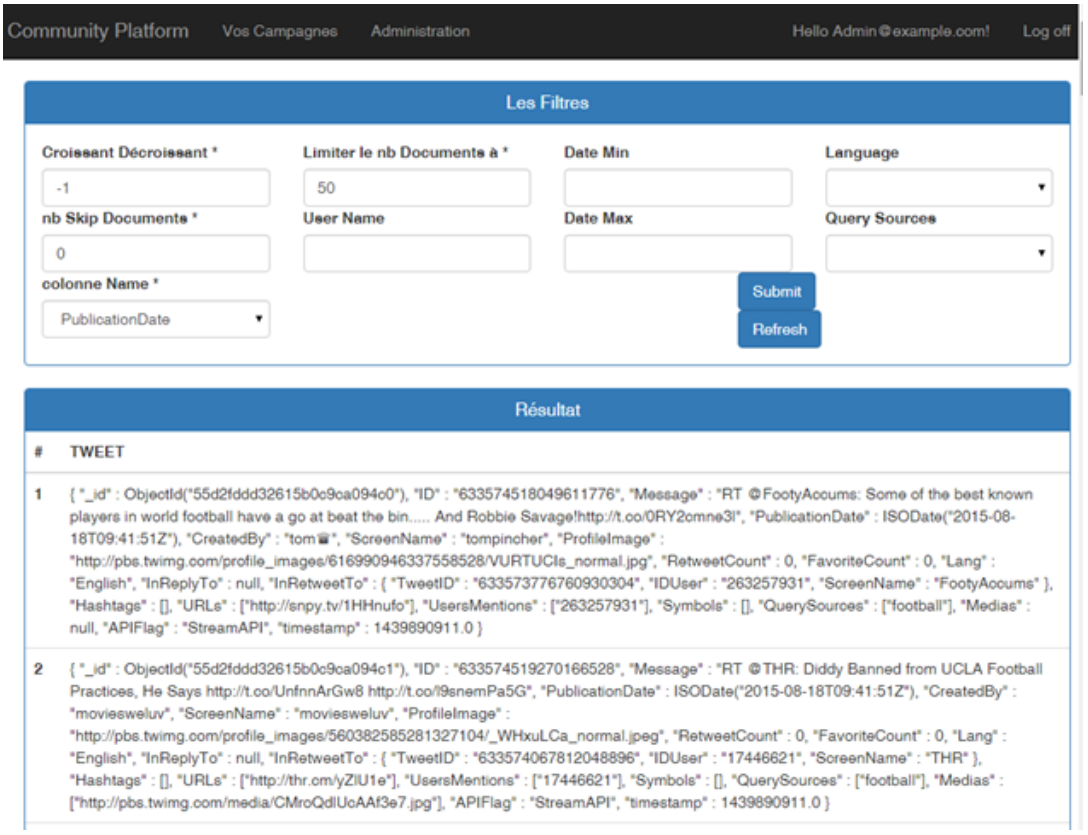

FIGURE 6.16 – Page de visualisation des données de l'application Web de gestion des campagnes

rafraîchit les filtres en réinitialisant les champs du formulaire également avec `JavaScript`. Ce renouvellement ciblé permet de ne pas rappeler toute la page et donc d'optimiser l'utilisation des ressources.

La partie inférieure concerne l'affichage des données, filtrées ou non.

Gestion de la collecte

La gestion de la collecte est un module central dans l'application. En effet, c'est lui qui communique avec la `WebStreamDataProvider` afin de collecter des données issues des réseaux sociaux, et qui communique avec la base de données MongoDB afin de stocker ces données, qui seront ensuite utilisées par les autres composants de l'application.

Pour les collectes sur Twitter, il est possible de sélectionner des tweets ayant le mot recherché, ou les tweets d'un utilisateur que l'on suit. Ainsi, en bas de l'interface, on retrouve deux parties qui ont des actions similaires : ajouter (`Add`), supprimer (`Suppress`) et tout supprimer (`Clear All`). La partie de droite gère les mots et la partie de gauche gère les utilisateurs Twitter. La collecte est effectuée en direct, via la *Stream API*. De plus, l'application offre également la possibilité, via les cases `Données antérieures`, de rechercher les données antérieures à la date de lancement de la collecte, via la *Search API*[29].

Les boutons `Play` et `Stop`, situés en haut à gauche de la page, permettent de démarrer

29. Dans les limites imposées par Twitter, c'est-à-dire 200 tweets par appel ; avec 450 appels en 15 minutes et la possibilité de remonter sur les 7 derniers jours uniquement

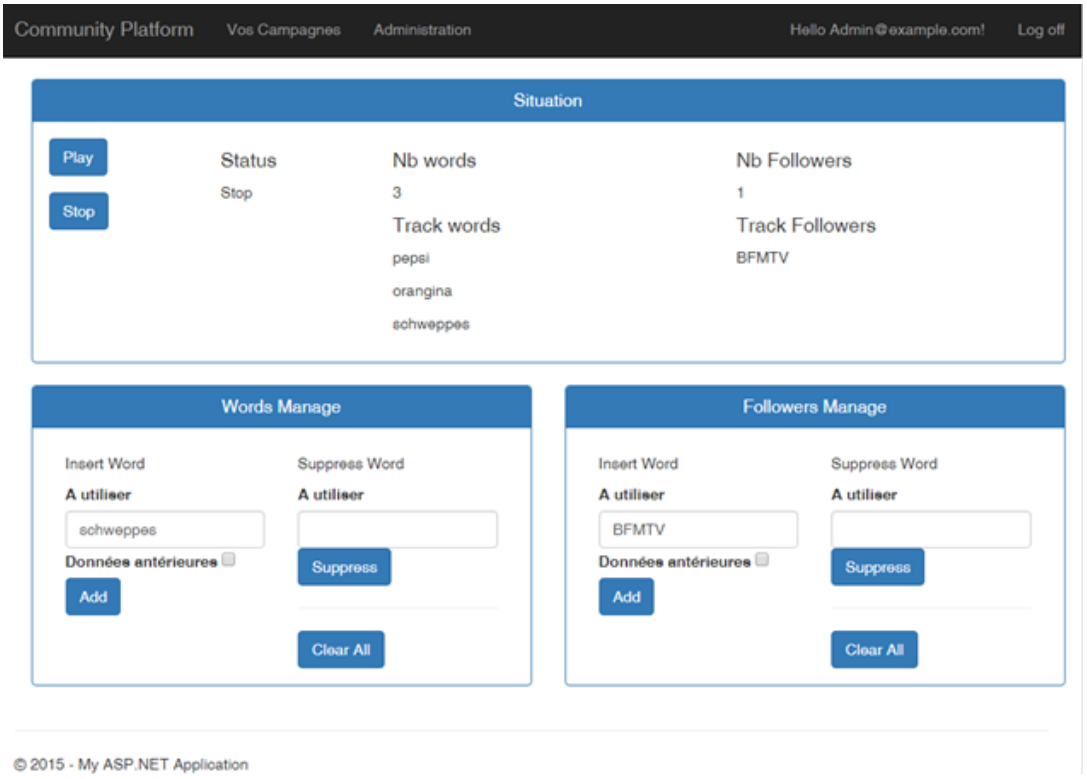

FIGURE 6.17 – Page de gestion de la collecte des données de l'application Web de gestion des campagnes

ou d'arrêter le processus de collecte. Sur leur droite est affiché le statut de la collecte (en pause, en cours, arrêtée), mis à jour automatiquement grâce à `SignalR`. Les *query sources* déjà saisies sont rappelées via les champs `Nb words`, `track words`, `Nb Followers` et `Track Followers` .

Exceptée la recherche de données antérieures au lancement de la collecte, qui est réalisée avec la *Search API* de Twitter, tous les autres appels sont traités par le `Hub` avec `SignalR` dont les méthodes sont :

- Ajouter des mots ou utilisateurs
- Supprimer un mot ou un utilisateur
- Supprimer tous les mots ou tous les utilisateurs
- Lancer la collecte
- Arrêter la collecte
- S'abonner à la méthode de rafraichissement du statut
- Récupérer les mots recherchés et leur nombre
- Récupérer les utilisateurs Twitter suivis et leur nombre

La collecte a donc une relation directe avec la `WebStreamDataProvider` et met en place le lancement du système de processus s'occupant de la collecte puis du stockage des données collectées dans la base MongoDB.

Administration d'une campagne

Cette page (figure 6.18) concerne l'administration d'une campagne ; à ne pas confondre avec l'administration générale du site présente dans le menu. Ici, les administrateurs de la campagne peuvent changer les droits des utilisateurs via un tableau répertoriant tous

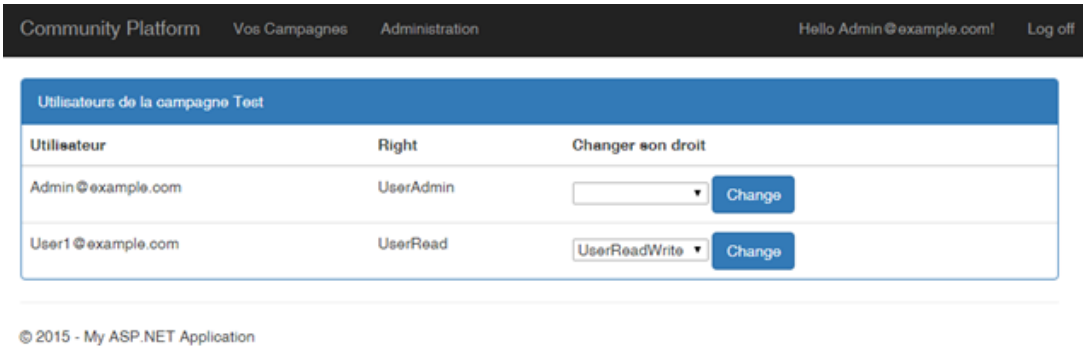

FIGURE 6.18 – Page d'administration d'une campagne de l'application Web de gestion des campagnes

les utilisateurs ayant un droit sur la campagne. Il suffit de sélectionner le nouveau droit à appliquer et de cliquer sur le bouton `Change` pour valider les modifications.

Gestion des analyses

La gestion des analyses s'effectue sur deux pages distinctes.

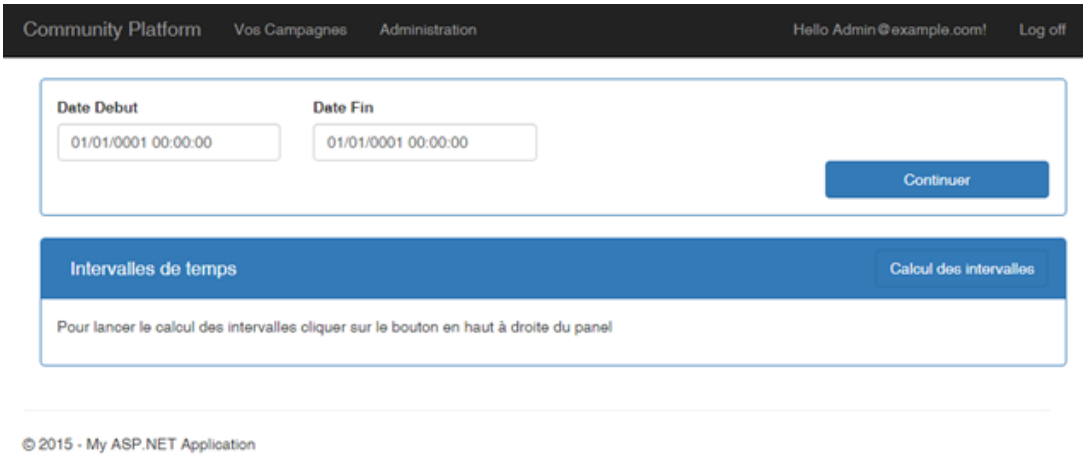

FIGURE 6.19 – Première page de gestion des analyses des données de l'application Web

Dans la première page, présentée dans la figure 6.19, l'utilisateur détermine l'intervalle de temps sur lequel lancer les analyses. Cet intervalle permet de limiter la quantité de données sur laquelle les analyses seront lancées, ou d'étudier des événements réels dont la période de temps est connue. Il y a deux manières de déterminer cet intervalle de temps.

Si l'utilisateur connaît déjà l'intervalle à utiliser, il lui suffit de rentrer les dates souhaitées et de cliquer sur le bouton `Continuer`, qui permet de naviguer vers la deuxième page de gestion des analyses.

Sinon, sur la partie basse de la page, le bouton `Calcul des intervalles` lance le script permettant de détecter les intervalles de temps les plus pertinents pour les analyses, grâce à l'algorithme présenté dans la partie 5.7.1.2, et construit, dans la partie `Intervalle de temps`, un graphique les rendant visibles. Ainsi, grâce à ce graphique, l'utilisateur peut choisir les dates de début et de fin de l'intervalle de temps.

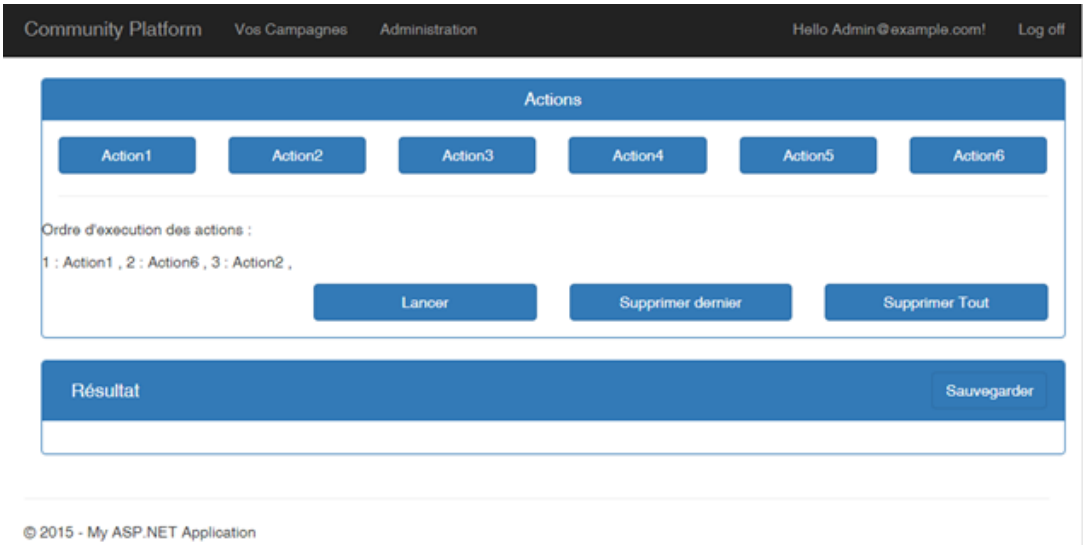

FIGURE 6.20 – Deuxième page de gestion des analyses des données de l'application Web

Sur la deuxième interface, l'utilisateur détermine les actions [30] à effectuer. La figure 6.20 présente une version générique, pour tout type d'analyse, de cette interface. Notre approche des analyses étant itérative, cette page offre la possibilité d'enchaîner plusieurs scripts. Les actions (qui sont encore à définir précisément) sont placées en haut de la plage, sous la forme de boutons. Lorsque l'utilisateur sélectionne une action, cette dernière est ajoutée à l'`ordre d'exécution des actions`, afin de garder une trace des actions et de leur enchaînement. Ensuite, après avoir sélectionné une ou plusieurs actions, l'utilisateur a trois possibilités :

- lancer la ou les actions, et visualiser le résultat ;
- supprimer la dernière action sélectionnée ;
- supprimer toutes les actions sélectionnées, afin de recommencer le processus à zéro.

La deuxième partie de l'interface permet d'afficher les résultats renvoyés par l'exécution des scripts. Cet affichage reste encore à déterminer suivant le rendu final souhaité. Un bouton `Sauvegarder` permet de stocker les résultats produits dans la base de données MongoDB.

6.8/ BILAN ET CONCLUSION

Cette partie résume les différentes sections de ce chapitre et présente l'état actuel de la plateforme DisCoCRM. Plusieurs personnes ont participé à ce projet :

- 1 responsable, chef de projet R&D
- 1 expert technique, chef de projet opérationnel
- 2 stagiaires pour le développement :
  - 1 pour le développement de la `WebStreamDataProvider`
  - 1 pour le reste de la plateforme DisCoCRM

30. Une action peut être définie par un ou plusieurs scripts `R`

Le temps passé pour les différentes phases a été de :

- 3 mois pour les spécifications techniques et fonctionnelles
- 4 mois pour le développement de la `WebStreamDataProvider`
- 6 mois pour le développement du reste de la plateforme DisCoCRM.

Dans le but de faciliter la maintenance des applications et pour d'éventuelles évolutions, la plateforme a été spécifiée sous la forme de trois briques fonctionnelles (figure 6.12). Comme les différentes architectures présentées le montrent, le fil conducteur des spécifications est que chaque brique soit indépendante. Cette architecture permet, en cas de changement (par exemple de la base de données NoSQL), de ne modifier qu'un seul composant (figure 6.13).

La première brique à avoir été développée est la `WebStreamDataProvider`. Celle-ci ne subira plus de grands changements, à part l'ajout de nouveaux connecteurs afin de communiquer avec d'autres sources de données (par exemple Facebook). L'interface créée sur l'application Web de contrôle afin de communiquer et gérer la collecte des données est elle aussi terminée. La gestion de la collecte de données est donc en place et fonctionnelle.

Afin d'intégrer de nouvelles sources de données, le schéma des données renvoyées par la `WebStreamDataProvider` est susceptible d'être modifié. En effet, chaque source de données ayant son propre schéma, il est préférable de stocker directement dans la base de données MongoDB les données dans leur format natif, une transformation étant trop coûteuse lors de la collecte de gros volumes de données. Lorsque l'application Web de contrôle, ou les scripts `R` souhaitent accéder aux données, la réalisation de vues permettra de n'avoir que les informations nécessaires.

La deuxième brique concerne l'entrepôt de données, réalisé avec MongoDB, un SGBD extérieur à l'application. Sa mise en place et son fonctionnement sont actuellement fonctionnels. De plus, la possibilité pour l'utilisateur de visualiser les données (filtrées ou non) est aussi en place. On peut donc considérer cette brique comme terminée.

Une première application Web de contrôle, utilisant le patron de conception `MVC`, a été réalisée afin de permettre les tests de la plateforme.

Concernant la couche des analyses, l'ensemble de l'architecture est en place et est fonctionnelle. L'intégration des scripts écrits en `R` est en cours, leur visualisation est encore en train d'être définie. La solution `R.NET` est fonctionnelle pour les tests, cependant les retours d'utilisateurs sur le Web montrent que cette librairie est assez sensible aux modifications apportées par de nouvelles versions du langage `R`. Pour éviter les problèmes à ce niveau, il est préférable d'utiliser un Web service afin d'exécuter les différentes scripts `R`. Pour cela, l'utilisation de `R Server`[31] par exemple semble préférable.

La mise en commun des différents profils sociaux avec celui contenu dans la base CRM, pour une seule et même personne, n'est pas encore implémentée dans DisCoCRM.

Les principales difficultés et écueils techniques étaient liées à la liaison entre la base de données MongoDB et les applications `ASP.NET` ; l'apprentissage du langage de requête de MongoDB et la familiarisation avec cette base de données NoSQL ; l'intégration des scripts `R` dans une application `ASP.NET` ; ainsi que l'enchaînement des scripts et la visualisation des résultats renvoyés par les scripts `R`.

Le développement de la plateforme DisCoCRM a permis de passer d'un prototype réalisé

31. https://www.microsoft.com/fr-fr/cloud-platform/r-server

pour un *proof of concept* (SNFreezer) à une application respectant les standards des entreprises eb-Lab et Teletech International, et favorisant les évolutions par un découplage des différentes briques.

7

CONCLUSION

Pour conclure ce manuscrit, nous résumerons tout d'abord les travaux que nous avons présentés en mettant en avant les contributions et leur utilité dans le cadre de l'entreprise partenaire de la thèse, puis nous terminerons en décrivant les principales perspectives de recherche ouvertes par nos travaux.

7.1/ BILAN

Nous avons montré, au travers d'exemples issus de la collaboration avec les entreprises eb-Lab et Teletech International, que la gestion de la relation client (CRM) doit évoluer vers le *Social CRM* et intégrer une dimension sociale grâce à une intégration des réseaux sociaux numériques comme nouveau canal de communication entre les entreprises, les marques et les consommateurs. Plus qu'une simple intégration des réseaux sociaux, la valorisation des données qu'ils produisent est un enjeu primordial pour la gestion de la relation client. L'état de l'art montre que de nombreux outils algorithmiques existent pour analyser les données des réseaux sociaux grand public et des réseaux sociaux d'entreprise. Toutefois, la valorisation des données issues des réseaux sociaux ne peut se faire que par l'interprétation des résultats de ces algorithmes par un expert de la gestion de la relation client. Les algorithmes ayant des conditions d'application et produisant parfois des artefacts non représentatifs, cette valorisation des données n'est pas une tâche facile. Par exemple certaines marches aléatoires ne sont pas adaptées aux graphes bipartis ; et les algorithmes utilisant la modularité pour la détection de communautés ont des limites de résolution qui ont pour conséquence la détection de communautés non pertinentes, car de taille inférieure au seuil de résolution. Aussi, la quantité de données issues des réseaux sociaux peut rendre les temps d'exécution des algorithmes particulièrement élevés. Ces difficultés font que l'intégration des réseaux sociaux comme source de données permettant de générer de la valeur pour les entreprises nécessite l'élaboration d'outils adaptés, faciles d'accès pour les experts de la gestion de la relation client, et capables de fournir des résultats pertinents.

En lien avec l'entreprise partenaire de la thèse, eb-Lab, et son client principal Teletech international, nous avons identifié les fonctionnalités essentielles auxquelles doit répondre un outil de type *Social CRM*. Nous avons mis en évidence deux priorités : 1) la construction d'une plateforme de collecte et de stockage des données issues des réseaux sociaux et 2) le développement d'outils d'analyse orientés utilisateur pour la détection de communautés et d'événements.

Les problématiques abordées dans la thèse correspondent aux deux orientations prioritaires. Au niveau de la plateforme nous devons être en capacité de collecter des données en provenance des principales applications de réseaux sociaux, et sur une période qui peut être longue. Nous devons donc être en mesure de traiter un flux important de données. Afin de faciliter les analyses et d'améliorer leur rapidité de mise en œuvre, il convient de limiter le recours aux *ETL* et donc de proposer un entrepôt de données dont le modèle de données est proche des structures attendues par les algorithmes. Au niveau des outils d'analyse, il est essentiel de développer des méthodes permettant de faciliter ou de garantir l'interprétabilité des résultats produits par les algorithmes. En effet, un des objectifs est de construire des outils pour des utilisateurs qui sont experts de la relation client et non pas des *data-scientist*. Ainsi, nous nous sommes concentrés sur la détection et la caractérisation de communautés et d'évènements.

Les contributions ont été développées dans les chapitres 4, 5 et 6 de ce document. Elles concernent tout d'abord la modélisation d'un profil thématique d'utilisateur afin de représenter la sémantique de son domaine d'activité. Ce profil est ensuite inclus dans un mécanisme de détection de communautés. Pour cela, nous avons utilisé en premier lieu l'algorithme *k-means* ; puis montré que l'algorithme de Louvain donne qualitativement de meilleurs résultats. Cette validation empirique, sur des données d'un réseau social d'entreprise, a été réalisée en s'appuyant sur une connaissance fine des utilisateurs. Nous avons ensuite développé une chaîne de sémantisation. Nous avons utilisé des éléments de la connaissance du domaine, modélisés par une hiérarchie de termes, afin de contextualiser les résultats de l'algorithme de Louvain sous la forme de communautés d'utilisateurs et de hashtags. Comme le montrent les expériences, associée aux profils thématiques, l'injection de connaissance du domaine dans les données améliore la modularité et met donc mieux en évidence la structure communautaire. Nous avons adopté une démarche similaire pour la détection d'événements. Tout d'abord, nous avons proposé une méthode, qui ne requiert pas de paramètre hormis la notion de bande passante, pour détecter des événements, qu'ils soient courts ou plus longs, à partir de séries temporelles. Ensuite, pour caractériser sémantiquement un événement, nous l'avons contextualisé par d'autres séries temporelles sur les hashtags.

Les contributions concernant la plateforme sont d'une part un prototype de plateforme de collecte, de stockage et d'analyse de données issues de Twitter (*SNFreezer*) ; et d'autre part une plateforme générique s-CRM (*DiscoCRM*) correspondant au standard de développement de l'entreprise eb-Lab. Pour la plateforme *SNFreezer*, nous avons contribué au développement d'un mode cluster et d'un mécanisme de reprise sur panne au niveau de la collecte de données. L'étude de l'état de l'art montrait que les plateformes existantes ne répondaient pas à ces deux critères, qui étaient essentiels pour satisfaire les besoins d'une gestion de la relation client utilisant des données des réseaux sociaux. Ces deux mécanismes ont été validés dans différents projets ; y compris la collecte des données de la coupe du monde de football 2014 (3,2To pour 1,1 milliard de tweets). Le stockage des données et la construction d'un entrepôt ont suivi une approche multi-paradigmes. Cette approche permet de répondre à l'objectif de stocker les données dans le ou les formats les plus proches des algorithmes d'analyse, ce qui facilite leur utilisation et réduit leur temps d'exécution. Une couche d'abstraction et des connecteurs vers PostgreSQL, HDFS/JSON, Neo4j, MongoDB ont été développés et validés lors de collectes. À partir de l'expérience acquise avec *SNFreezer*, nous avons spécifié puis développé la plateforme générique *DiscoCRM*, s'appuyant sur des services Web, afin que l'entreprise eb-Lab puisse la vendre à ses clients sous différentes formes (marque blanche, produit

autonome, infogérance). Une attention particulière a été portée à l'architecture logicielle de la plateforme qui a été spécifiée en termes de services Web afin de diminuer les couplage des différents composants et plus spécifiquement au niveau des mécanismes de collecte de données provenant des réseaux sociaux et leur stockage dans un entrepôt. Un entrepôt MongoDB et les mécanismes de collecte de données Twitter ont été mis en production ainsi que la partie back-office d'administration de la plateforme et la partie front-office dédiée à l'administration par les clients (*community manager, data scientist*) des campagnes collectes de données. L'incorporation des algorithmes d'analyse dans un workflow et leur gestion au moyen d'une interface Web étaient encore à l'état de prototype lors du rachat de Teletech International et l'arrêt des activités de recherche et de développement de la société eb-Lab qui s'en est suivi.

7.2/ PERPECTIVES

Nos travaux ouvrent de nouvelles perspectives dans la prise en compte de la sémantique dans la détection de communautés. Nous souhaitons poursuivre l'introduction d'une composante contextuelle pour constituer une chaîne de sémantisation, depuis la collecte des données jusqu'à l'interprétation des résultats, en formalisant l'utilisation d'ontologies pour enrichir les données ou leur visualisation. Au travers des différentes analyses sur les données réelles (co-voiturage, élections, etc.), nous nous sommes aperçus que le processus de création de connaissance est incrémental et itératif. Incrémental, dans le sens où des algorithmes permettent d'obtenir des informations générales sur les données (centralité, communautés non recouvrantes, etc.). Et itératif à partir de ces éléments, dans le sens où il est possible de formuler des hypothèses qui vont donner lieu à de nouvelles expérimentations sur des données éventuellement plus précises. De plus, ce processus inclut le recours à des ontologies comme outils de contextualisation des données ou des résultats. Toutefois, peu de tentatives de formalisation ont été développées. Nous avons commencé à exploiter les liens entre les termes sous la forme d'une hiérarchie de généralisation/spécialisation et nous estimons que les techniques de *graph embedding* [Goyal et al., 2017], permettant de passer d'une modèle graphe à un modèle vectoriel, peuvent permettre de prendre en compte une partie de la richesse des liens offerts par une ontologie de domaine ou d'application.

De même, la richesse des différents liens produits par les actions des utilisateurs sur les réseaux sociaux doit être mieux prise en compte dans les données. Les modèles de réseaux multi-couches ou multi-relationnels ouvrent des perspectives importantes, les principaux types d'algorithmes commençant à exploiter les réseaux multi-couches. Cependant, les questions de l'interprétation des résultats, de la sélection et de la contextualisation des données d'entrée sont encore plus importantes dans ces nouvelles approches. Par exemple, la modélisation tensorielle des profils d'utilisateurs que nous avons ébauchée au chapitre 4 peut être utilisée pour représenter des hypergraphes. Il est ainsi possible d'appliquer des algorithmes de détection de communautés, soit par extension des méthodes de clustering, soit par décomposition tensorielle. Dans les deux cas, la modélisation des données en entrée est dépendante du contexte métier et de la question qui motive l'analyse. De plus, l'interprétation des résultats est rendue encore plus difficile par la complexité des outils mis en œuvre. Cependant cette problématique rejoint les techniques de *graph embedding* dans la prise en compte des réseaux multi-relationnel et multi-couches [De Bacco et al., 2017].

À un niveau plus appliqué, et contrairement aux outils existants, comme Zeppelin [1] ou Jupyter [2], qui ont pour cible les *data-scientists*, notre système a pour but d'être utilisé par des *community managers* experts métier, à partir de vues ou d'extractions de données réalisées par des experts des données. Afin de supporter une méthodologie d'analyse itérative et incrémentale, et de guider les *community managers* dans leurs analyses, il est nécessaire que le workflow d'analyse de plateforme permette : 1) d'extraire des données et de les sauvegarder, tout en leur associant un contexte sous la forme de liens avec une ontologie, que ce soit au niveau de la donnée elle-même pour un enrichissement sémantique ou au moyen de métadonnées pour faciliter leur réutilisation, 2) d'exécuter des algorithmes sur les données, éventuellement sur proposition de la plateforme en fonction du type d'analyse souhaitée et des caractéristiques des données ; 3) de sauvegarder les résultats des algorithmes, de les commenter, de les comparer à la connaissance du domaine, par exemple pour détecter des singularités (section 5.7.3) ou encore pour les réutiliser comme des données pour d'autres analyses.

1. https://zeppelin.apache.org/
2. http://jupyter.org/

BIBLIOGRAPHIE

[Abdi, 2007] Abdi, H. (2007). **Singular value decomposition (svd) and generalized singular value decomposition**. *Encyclopedia of measurement and statistics. Thousand Oaks (CA) : Sage*, pages 907–12.

[Abel et al., 2011a] Abel, F., Araújo, S., Gao, Q., et Houben, G.-J. (2011a). **Analyzing cross-system user modeling on the social web**. Dans *Web Engineering*, pages 28–43. Springer.

[Abel et al., 2011b] Abel, F., Gao, Q., Houben, G.-J., et Tao, K. (2011b). **Semantic enrichment of twitter posts for user profile construction on the social web**. Dans *The Semanic Web : Research and Applications*, pages 375–389. Springer.

[Ahn et al., 2010] Ahn, Y.-Y., Bagrow, J. P., et Lehmann, S. (2010). **Link communities reveal multiscale complexity in networks**. *Nature*, 466(7307) :761–764.

[Ajmera et al., 2013] Ajmera, J., Ahn, H.-i., Nagarajan, M., Verma, A., Contractor, D., Dill, S., et Denesuk, M. (2013). **A crm system for social media : challenges and experiences**. Dans *Proc. of the 22nd international conference on World Wide Web*, pages 49–58. International World Wide Web Conferences Steering Committee.

[Albert et al., 2002] Albert, R., et Barabási, A.-L. (2002). **Statistical mechanics of complex networks**. *Reviews of modern physics*, 74(1) :47.

[Atefeh et al., 2013] Atefeh, F., et Khreich, W. (2013). **A survey of techniques for event detection in twitter**. *Computational Intelligence*, 31(1) :132–164.

[Aynaud et al., 2010] Aynaud, T., et Guillaume, J.-L. (2010). **Static community detection algorithms for evolving networks**. Dans *Modeling and optimization in mobile, ad hoc and wireless networks (WiOpt)*, pages 513–519. IEEE.

[Bai et al., 2003] Bai, J., et Perron, P. (2003). **Computation and analysis of multiple structural change models**. *Journal of applied econometrics*, 18(1) :1–22.

[Barabási et al., 1999] Barabási, A.-L., et Albert, R. (1999). **Emergence of scaling in random networks**. *Science*, 286(5439) :509–512.

[Barrat, 2013] Barrat, A. (2013). **La notion de réseau complexe : du réseau comme abstraction et outil à la masse de données des réseaux sociaux en ligne**. *Communication & Organisation*, (1) :15–24.

[Basaille-Gahitte et al., 2013a] Basaille-Gahitte, I., Abrouk, L., Cullot, N., et Leclercq, E. (2013a). **Apports des réseaux sociaux dans les si une application à la gestion de la relation client**. Dans *Inforsid*, pages 300–308.

[Basaille-Gahitte et al., 2013b] Basaille-Gahitte, I., Abrouk, L., Cullot, N., et Leclercq, E. (2013b). **Using social networks to enhance customer relationship management**. Dans *Proceedings of the Fifth International Conference on Management of Emergent Digital EcoSystems*, pages 169–176. ACM.

[Basaille-Gahitte et al., 2014] Basaille-Gahitte, I., Abrouk, L., Cullot, N., et Leclercq, E. (2014). **Apports des réseaux sociaux pour la gestion de la relation client**. *Revue des Sciences et Technologies de l'Information-Série ISI : Ingénierie des Systèmes d'Information*, 19(2) :85–109.

[Basaille-Gahitte et al., 2016] Basaille-Gahitte, I., Kirgizov, S., Leclercq, É., Savonnet, M., et Cullot, N. (2016). **Towards a twitter observatory : A multi-paradigm framework for collecting, storing and analysing tweets**. Dans *Research Challenges in Information Science (RCIS), 2016 IEEE Tenth International Conference on*, pages 1–10. IEEE.

[Basaille-Gahitte et al., 2017] Basaille-Gahitte, I., et Leclercq, E. (2017). **Un observatoire pour la modélisation et l'analyse des réseaux multi-relationnels : une application à l'étude du discours politique sur twitter (à paraître)**. *Le document numérique*, 20(1) :1–30.

[Blondel et al., 2008] Blondel, V. D., Guillaume, J.-L., Lambiotte, R., et Lefebvre, E. (2008). **Fast unfolding of communities in large networks**. *Journal of Statistical Mechanics : Theory and Experiment*, 2008(10) :P10008.

[Bondiombouy et al., 2015] Bondiombouy, C., Kolev, B., Levchenko, O., et Valduriez, P. (2015). **Integrating big data and relational data with a functional sql-like query language**. Dans *Database and Expert Systems Applications*, pages 170–185. Springer.

[Borgatti et al., 1992] Borgatti, S. P., et Everett, M. G. (1992). **Notions of position in social network analysis**. *Sociological methodology*, pages 1–35.

[Brandes et al., 2005] Brandes, U., et Erlebach, T. (2005). **Network analysis : methodological foundations**, volume 3418. Springer Science & Business Media.

[Bringay et al., 2011] Bringay, S., Béchet, N., Bouillot, F., Poncelet, P., Roche, M., et Teisseire, M. (2011). **Towards an on-line analysis of tweets processing**. Dans *Database and Expert Systems Applications*, pages 154–161. Springer.

[Burnap et al., 2014] Burnap, P., Rana, O., Williams, M., Housley, W., Edwards, A., Morgan, J., Sloan, L., et Conejero, J. (2014). **COSMOS : Towards an integrated and scalable service for analysing social media on demand**. *Int. Journal of Parallel, Emergent and Distributed Systems*, 30(2) :80–100.

[Cattuto et al., 2008] Cattuto, C., Baldassarri, A., Servedio, V., et Loreto, V. (2008). **Emergent community structure in social tagging systems**. *Advances in Complex Systems*, 11(04) :597–608.

[Cordina et al., 2013] Cordina, P., et Fayon, D. (2013). **Community management : fédérer des communautés sur les médias sociaux**. Pearson Education France.

[Costa et al., 2016] Costa, A., Kushnarev, S., Liberti, L., et Sun, Z. (2016). **Divisive heuristic for modularity density maximization**. *Computers & Operations Research*, 71 :100–109.

[Costa et al., 2012] Costa, P., Souza, F. F., Times, V. C., et Benevenuto, F. (2012). **Towards integrating online social networks and business intelligence**. Dans *IADIS international conference on Web based communities and social media*, volume 2012.

[Cuzzocrea et al., 2016] Cuzzocrea, A., De Maio, C., Fenza, G., Loia, V., et Parente, M. (2016). **Olap analysis of multidimensional tweet streams for supporting advanced analytics**. Dans *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 992–999. ACM.

[Danisch, 2015] Danisch, M. (2015). **Mesures de proximité appliquées à la détection de communautés dans les grands graphes de terrain**. PhD thesis, Paris 6.

[Danisch et al., 2013] Danisch, M., Guillaume, J.-L., et Le Grand, B. (2013). **Towards multi-ego-centred communities : a node similarity approach**. *International Journal of Web Based Communities*, 9(3) :299–322.

[Davies et al., 1979] Davies, D. L., et Bouldin, D. W. (1979). **A cluster separation measure**. *IEEE transactions on pattern analysis and machine intelligence*, (2) :224–227.

[de Amorim et al., 2015] de Amorim, R. C., et Hennig, C. (2015). **Recovering the number of clusters in data sets with noise features using feature rescaling factors**. *Information Sciences*, 324 :126–145.

[De Bacco et al., 2017] De Bacco, C., Power, E. A., Larremore, D. B., et Moore, C. (2017). **Community detection, link prediction, and layer interdependence in multilayer networks**. *Physical Review E*, 95(4) :042317.

[DeSolla Price, 1965] DeSolla Price, D. (1965). **Networks of scientific papers**. *Science*, pages 510–515.

[Dey et al., 2001] Dey, A. K., Abowd, G. D., et Salber, D. (2001). **A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications**. *Hum.-Comput. Interact.*, 16(2) :97–166.

[Dhillon et al., 2007] Dhillon, I. S., Guan, Y., et Kulis, B. (2007). **Weighted graph cuts without eigenvectors a multilevel approach**. *IEEE transactions on pattern analysis and machine intelligence*, 29(11).

[Diday, 1971] Diday, E. (1971). **Une nouvelle méthode en classification automatique et reconnaissance des formes la méthode des nuées dynamiques**. *Revue de statistique appliquée*, 19(2) :19–33.

[Ducruet, 2010] Ducruet, C. (2010). **Les mesures globales d'un réseau**. *Synthèse du groupe FMR*, page 9.

[Ducruet, 2012] Ducruet, C. (2012). **Multigraphes, multiplexes, et réseaux couplés**.

[Dunn, 1973] Dunn, J. C. (1973). **A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters**. Taylor & Francis.

[Elmore et al., 2015] Elmore, A., Duggan, J., Stonebraker, M., Balazinska, M., Cetintemel, U., Gadepally, V., Heer, J., Howe, B., Kepner, J., Kraska, T., et others (2015). **A demonstration of the bigdawg polystore system**. *Proceedings of the VLDB Endowment*, 8(12) :1908–1911.

[Erdos et al., 1961] Erdos, P., et Rényi, A. (1961). **On the evolution of random graphs**. *Bull. Inst. Internat. Statist*, 38(4) :343–347.

[Firan et al., 2007] Firan, C. S., Nejdl, W., et Paiu, R. (2007). **The benefit of using tag-based profiles**. Dans *Web Conference, 2007. LA-WEB 2007. Latin American*, pages 32–41. IEEE.

[Forgy, 1965] Forgy, E. W. (1965). **Cluster analysis of multivariate data : Efficiency vs. interpretability of classifications**. *biometrics*, 21 :768–769.

[Fortunato, 2010] Fortunato, S. (2010). **Community detection in graphs**. *Physics Reports*, 486(3) :75–174.

[Fortunato et al., 2007] Fortunato, S., et Barthelemy, M. (2007). **Resolution limit in community detection**. *Proceedings of the National Academy of Sciences*, 104(1) :36–41.

[Fortunato et al., 2016] Fortunato, S., et Hric, D. (2016). **Community detection in networks : A user guide**. *Physics Reports*, 659 :1–44.

[Freeman, 1977] Freeman, L. C. (1977). **A set of measures of centrality based on betweenness**. *Sociometry*, pages 35–41.

[Gallinucci et al., 2013] Gallinucci, E., Golfarelli, M., et Rizzi, S. (2013). **Meta-stars : multidimensional modeling for social business intelligence**. Dans *Proceedings of the sixteenth international workshop on Data warehousing and OLAP*, pages 11–18. ACM.

[Gan et al., 2013] Gan, M., Dou, X., et Jiang, R. (2013). **From ontology to semantic similarity : calculation of ontology-based semantic similarity**. *The Scientific World Journal*, 2013 :1–11.

[Girvan et al., 2002] Girvan, M., et Newman, M. E. (2002). **Community structure in social and biological networks**. *Proceedings of the national academy of sciences*, 99(12) :7821–7826.

[Golbeck, 2009] Golbeck, J. (2009). **Trust and nuanced profile similarity in online social networks**. *ACM Transactions on the Web (TWEB)*, 3(4) :12.

[Good et al., 2010] Good, B. H., de Montjoye, Y.-A., et Clauset, A. (2010). **Performance of modularity maximization in practical contexts**. *Physical Review E*, 81(4) :046106.

[Goyal et al., 2017] Goyal, P., et Ferrara, E. (2017). **Graph embedding techniques, applications, and performance : A survey**. *arXiv preprint arXiv :1705.02801*.

[Guarino, 1997] Guarino, N. (1997). **Understanding, building and using ontologies**. *International Journal of Human-Computer Studies*, 46(2-3) :293–310.

[Guille et al., 2014] Guille, A., et Favre, C. (2014). **Mention-anomaly-based event detection and tracking in twitter**. Dans *Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM Int. Conf. on*, pages 375–382. IEEE.

[Gusfield, 1978] Gusfield, J. R. (1978). **Community : A critical response**. HarperCollins.

[Gutman, 2004] Gutman, R. J. (2004). **Reach-based routing : A new approach to shortest path algorithms optimized for road networks.** *ALENEX/ANALC*, 4 :100–111.

[Harrigan et al., 2014] Harrigan, P., et Miles, M. (2014). **From e-crm to s-crm. critical factors underpinning the social crm activities of smes**. *Small Enterprise Research*, 21(1) :99–116.

[Héon et al., 2013] Héon, M., et Nkambou, R. (2013). **G-owl : Vers un langage de modélisation graphique, polymorphique et typé pour la construction d'une ontologie dans la notation owl**. Dans *IC-24èmes Journées francophones d'Ingénierie des Connaissances*.

[Hogben, 2013] Hogben, L. (2013). **Handbook of linear algebra**. Chapman and Hall/-CRC.

[Hung et al., 2008] Hung, C.-C., Huang, Y.-C., Hsu, J. Y.-j., et Wu, D. K.-C. (2008). **Tag-based user profiling for social media recommendation**. Dans *Workshop on Intelligent Techniques for Web Personalization and Recommender Systems at AAAI*.

[Jaccard, 1912] Jaccard, P. (1912). **The distribution of the flora in the alpine zone.** *New phytologist*, 11(2) :37–50.

[Jain, 2010] Jain, A. K. (2010). **Data clustering : 50 years beyond k-means**. *Pattern recognition letters*, 31(8) :651–666.

[James et al., 2014a] James, N. A., Kejariwal, A., et Matteson, D. S. (2014a). **Leveraging cloud data to mitigate user experience from" breaking bad"**. *arXiv preprint arXiv :1411.7955*.

[James et al., 2014b] James, N. A., Kejariwal, A., et Matteson, D. S. (2014b). **Leveraging cloud data to mitigate user experience from "breaking bad"**. *arXiv preprint arXiv :1411.7955*.

[Kanawati, 2011] Kanawati, R. (2011). **Licod : Leaders identification for community detection in complex networks**. Dans *Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on*, pages 577–582. IEEE.

[Kanawati, 2013] Kanawati, R. (2013). **Détection de communautés dans les grands graphes d'interactions (multiplexes) : état de l'art**. https://hal.archives-ouvertes.fr/hal-00881668v1.

[Karaboga, 2005] Karaboga, D. (2005). **An idea based on honey bee swarm for numerical optimization**. https://pdfs.semanticscholar.org/cf20/e34a1402a115523910d2a4243929f6704db1.pdf.

[Karaboga et al., 2007] Karaboga, D., et Basturk, B. (2007). **A powerful and efficient algorithm for numerical function optimization : artificial bee colony (ABC) algorithm**. *Journal of global optimization*, 39(3) :459–471.

[Katz, 1953] Katz, L. (1953). **A new status index derived from sociometric analysis**. *Psychometrika*, 18(1) :39–43.

[Kepner et al., 2011] Kepner, J., et Gilbert, J. (2011). **Graph algorithms in the language of linear algebra**. SIAM.

[Kernighan et al., 1970] Kernighan, B. W., et Lin, S. (1970). **An efficient heuristic procedure for partitioning graphs**. *The Bell system technical journal*, 49(2) :291–307.

[Khazaei et al., 2016] Khazaei, H., Fokaefs, M., Zareian, S., Beigi-Mohammadi, N., Ramprasad, B., Shtern, M., Gaikwad, P., et Litoiu, M. (2016). **How do I choose the right NoSQL solution ? a comprehensive theoretical and experimental survey**. *Big Data and Information Analytics (BDIA)*, 2 :1–32.

[Killick et al., 2012a] Killick, R., Fearnhead, P., et Eckley, I. (2012a). **Optimal detection of changepoints with a linear computational cost**. *Journal of the American Statistical Association*, 107(500) :1590–1598.

[Killick et al., 2012b] Killick, R., Fearnhead, P., et Eckley, I. A. (2012b). **Optimal detection of changepoints with a linear computational cost**. *Journal of the American Statistical Association*, 107(500) :1590–1598.

[Kivelä et al., 2014] Kivelä, M., Arenas, A., Barthelemy, M., Gleeson, J. P., Moreno, Y., et Porter, M. A. (2014). **Multilayer networks**. *Journal of complex networks*, 2(3) :203–271.

[Kleinberg, 1999] Kleinberg, J. (1999). **Authoritative sources in a hyperlinked environment**. *Journal of the ACM (JACM)*, 46(5) :604–632.

[Kleinfeld, 2002] Kleinfeld, J. (2002). **Could it be a big world after all ? the six degrees of separation myth**. *Society, April*, 12 :5–2.

[Knox et al., 2007] Knox, S., Payne, A., Ryals, L., Maklan, S., et Peppard, J. (2007). **Customer relationship management**. Routledge.

[Kondrashova et al., 2016] Kondrashova, T., Frame, A., et Kirgizov, S. (2016). **(re)constuire la temporalité d'un événement médiatique sur twitter : une étude contrastive**. XXe congrès de la SFSIC : Temps, temporalités et information-communication.

[Kraiem et al., 2015] Kraiem, M. B., Feki, J., Khrouf, K., Ravat, F., et Teste, O. (2015). **Modeling and olaping social media : the case of twitter**. *Social Network Analysis and Mining*, 5(1) :47.

[Kumar, 2010] Kumar, V. (2010). **Customer relationship management**. Wiley Online Library.

[Langville et al., 2005] Langville, A. N., et Meyer, C. D. (2005). **A survey of eigenvector methods for web information retrieval**. *SIAM review*, 47(1) :135–161.

[Leavitt et al., 2009] Leavitt, A., Burchard, E., Fisher, D., et Gilbert, S. (2009). **The Influentials : New Approaches for Analyzing Influence on Twitter**. *Webecology Project*, 4(2) :1–18.

[Leclercq et al., 2016] Leclercq, E., Savonnet, M., Grison, T., Kirgizov, S., et Basaille-Gahitte, I. (2016). **SNFreezer : a Platform for Harvesting and Storing Tweets in a Big Data Context**, chapitre 1, pages 19–33. Peter Lang.

[Lee et al., 2014] Lee, S., Kim, N., et Kim, J. (2014). **A multi-dimensional analysis and data cube for unstructured text and social media**. Dans *Big Data and Cloud Computing (BdCloud), 2014 IEEE Fourth International Conference on*, pages 761–764. IEEE.

[Leskovec et al., 2014] Leskovec, J., Rajaraman, A., et Ullman, J. D. (2014). **Mining of massive datasets**. Cambridge University Press.

[Li et al., 2011] Li, J., Wang, G. A., et Chen, H. (2011). **Identity matching using personal and social identity features**. *Information Systems Frontiers*, 13(1) :101–113.

[Li et al., 2012] Li, R., Lei, K. H., Khadiwala, R., et Chang, K. K.-C. (2012). **TEDAS : A Twitter-based event detection and analysis system**. Dans *International Conference on Data engineering (ICDE)*, pages 1273–1276. IEEE.

[Lim et al., 2013] Lim, H., Han, Y., et Babu, S. (2013). **How to fit when no one size fits.** Dans *CIDR*, volume 4, page 35.

[MacQueen, 1967] MacQueen, J. (1967). **Some methods for classification and analysis of multivariate observations**. Dans *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA.

[Malek et al., 2016] Malek, M., et Attal, J.-P. (2016). **Un nouvel algorithme de propagation de labels avec barrages**. *Revue d'Intelligence Artificielle*, 30(4) :pp.393–418.

[Manning et al., 2008] Manning, C. D., Raghavan, P., Schütze, H., et others (2008). **Introduction to information retrieval**, volume 1. Cambridge university press Cambridge.

[Mansmann, 2008] Mansmann, S. (2008). **Extending the OLAP technology to handle non-conventional and complex data**. PhD thesis.

[McMillan et al., 1986] McMillan, D. W., et Chavis, D. M. (1986). **Sense of community : A definition and theory**. *Journal of community psychology*, 14(1) :6–23.

[Melville et al., 2009] Melville, P., Sindhwani, V., et Lawrence, R. D. (2009). **Social media analytics : Channeling the power of the blogosphere for marketing insight**.

[Mohan et al., 2008] Mohan, S., Choi, E., et Min, D. (2008). **Conceptual modeling of enterprise application system using social networking and web 2.0 ?social crm system ?** Dans *Convergence and Hybrid Information Technology, 2008. ICHIT'08. Int. Conference on*, pages 237–244. IEEE.

[Motwani et al., 2010] Motwani, R., et Raghavan, P. (2010). **Randomized algorithms**. Chapman & Hall/CRC.

[Newman, 2006] Newman, M. E. (2006). **Modularity and community structure in networks**. *Proceedings of the national academy of sciences*, 103(23) :8577–8582.

[Newman et al., 2004] Newman, M. E., et Girvan, M. (2004). **Finding and evaluating community structure in networks**. *Physical review E*, 69(2) :026113.

[Norblin, 2014] Norblin, J. (2014). **Réalisation d'une plateforme pour la collecte, le stockage, l'analyse et la visualisation de données de type graphe. Application au projet Twitter aux Élections Européennes 2014**. Rapport technique, Rapport de stage de Master 2 Professionnel, Université de Bourgogne.

[Page et al., 1999] Page, L., Brin, S., Motwani, R., et Winograd, T. (1999). **The PageRank citation ranking : Bringing order to the Web**. Rapport technique, Stanford InfoLab.

[Papadopoulos et al., 2012] Papadopoulos, S., Kompatsiaris, Y., Vakali, A., et Spyridonos, P. (2012). **Community detection in social media**. *Data Mining and Knowledge Discovery*, 24(3) :515–554.

[Parthasarathy et al., 2011] Parthasarathy, S., Ruan, Y., et Satuluri, V. (2011). **Community discovery in social networks : Applications, methods and emerging trends**. Dans *Social network data analytics*, pages 79–113. Springer.

[Parzen, 1962] Parzen, E. (1962). **On estimation of a probability density function and mode**. *The annals of mathematical statistics*, pages 1065–1076.

[Payne et al., 2005] Payne, A., et Frow, P. (2005). **A strategic framework for customer relationship management**. *Journal of marketing*, 69(4) :167–176.

[Pelleg et al., 2000] Pelleg, D., Moore, A. W., et others (2000). **X-means : Extending k-means with efficient estimation of the number of clusters.** Dans *ICML*, volume 1, pages 727–734.

[Perrin, 2011] Perrin, C. (2011). **Dynamique identitaire et partitions sociales : le cas de l'identité'raciale'des noirs en france**. PhD thesis, Université de Bourgogne.

[Plantié et al., 2013] Plantié, M., et Crampes, M. (2013). **Survey on social community detection**. Dans *Social media retrieval*, pages 65–85. Springer.

[Pons et al., 2005] Pons, P., et Latapy, M. (2005). **Computing communities in large networks using random walks**. Dans *International Symposium on Computer and Information Sciences*, pages 284–293. Springer.

[Porter et al., 2009] Porter, M. A., Onnela, J.-P., et Mucha, P. J. (2009). **Communities in networks**. *Notices of the AMS*, 56(9) :1082–1097.

[Quan, 2011] Quan, H. (2011). **Online Social Networks & Social Network Services : A Technical Survey**. CRC Press.

[Raghavan et al., 2007] Raghavan, U. N., Albert, R., et Kumara, S. (2007). **Near linear time algorithm to detect community structures in large-scale networks**. *Physical review E*, 76(3) :036106.

[Rand, 1971] Rand, W. M. (1971). **Objective criteria for the evaluation of clustering methods**. *Journal of the American Statistical association*, 66(336) :846–850.

[Rehman, 2015] Rehman, N. U. (2015). **Extending the OLAP Technology for Social Media Analysis**. PhD thesis.

[Rehman et al., 2013] Rehman, N. U., Weiler, A., et Scholl, M. H. (2013). **Olaping social media : the case of twitter**. Dans *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 1139–1146. ACM.

[Rignault et al., 2012] Rignault, L., et Bonneton, L. (2012). **Manuel Du Social Media Marketing**. BoD-Books on Demand France.

[Rogers et al., 1960] Rogers, D. J., Tanimoto, T. T., et others (1960). **A computer program for classifying plants.** *Science (Washington)*, 132 :1115–18.

[Rosenblatt et al., 1956] Rosenblatt, M., et others (1956). **Remarks on some nonparametric estimates of a density function**. *The Annals of Mathematical Statistics*, 27(3) :832–837.

[Rousseeuw, 1987] Rousseeuw, P. J. (1987). **Silhouettes : a graphical aid to the interpretation and validation of cluster analysis**. *Journal of computational and applied mathematics*, 20 :53–65.

[Ryals et al., 2000] Ryals, L., Knox, S., et Maklan, S. (2000). **Customer relationship management (CRM) : Building the business case**. Cranfield University.

[Sadalage et al., 2012] Sadalage, P. J., et Fowler, M. (2012). **NoSQL distilled : a brief guide to the emerging world of polyglot persistence**. Pearson Education.

[Seifi, 2012] Seifi, M. (2012). **Cœurs stables de communautés dans les graphes de terrain**. PhD thesis.

[Shannon, 1948] Shannon, C. (1948). **A mathematical theory of communication, bell system technical journal 27 : 379-423 and 623–656**. *Mathematical Reviews (MathSciNet) : MR10, 133e*.

[Shi et al., 2000] Shi, J., et Malik, J. (2000). **Normalized cuts and image segmentation**. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8) :888–905.

[Shimbel, 1953] Shimbel, A. (1953). **Structural parameters of communication networks**. *The bulletin of mathematical biophysics*, 15(4) :501–507.

[Snyder et al., 1979] Snyder, D., et Kick, E. L. (1979). **Structural position in the world system and economic growth, 1955-1970 : A multiple-network analysis of transnational interactions**. *American journal of Sociology*, 84(5) :1096–1126.

[Soltani et al., 2016] Soltani, Z., et Navimipour, N. J. (2016). **Customer relationship management mechanisms : A systematic review of the state of the art literature and recommendations for future research**. *Computers in Human Behavior*, 61 :667–688.

[Swift, 2001] Swift, R. S. (2001). **Accelerating customer relationships : Using CRM and relationship technologies**. Prentice Hall Professional.

[Travers et al., 1967] Travers, J., et Milgram, S. (1967). **The small world problem**. *Phychology Today*, 1 :61–67.

[Van Rijsbergen, 1979] Van Rijsbergen, C. J. (1979). **Information retrieval**.

[Vinh et al., 2009] Vinh, N. X., Epps, J., et Bailey, J. (2009). **Information theoretic measures for clusterings comparison : is a correction for chance necessary ?** Dans *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1073–1080. ACM.

[Vinh et al., 2010] Vinh, N. X., Epps, J., et Bailey, J. (2010). **Information theoretic measures for clusterings comparison : Variants, properties, normalization and correction for chance**. *Journal of Machine Learning Research*, 11(Oct) :2837–2854.

[Von Luxburg, 2007] Von Luxburg, U. (2007). **A tutorial on spectral clustering**. *Statistics and computing*, 17(4) :395–416.

[Ward Jr, 1963] Ward Jr, J. H. (1963). **Hierarchical grouping to optimize an objective function**. *Journal of the American statistical association*, 58(301) :236–244.

[Watts et al., 1998] Watts, D. J., et Strogatz, S. H. (1998). **Collective dynamics of 'small-world'networks**. *nature*, 393(6684) :440–442.

[Winer, 2001] Winer, R. S. (2001). **A framework for customer relationship management**. *California management review*, 43(4) :89–105.

[Witten et al., 2005] Witten, I. H., et Frank, E. (2005). **Data Mining : Practical Machine Learning Tools and Techniques**. Morgan Kaufmann, 2 édition. (Ercument-2011-11-01).

[Wu et al., 2009] Wu, B., Ye, Q., Yang, S., et Wang, B. (2009). **Group crm : a new telecom crm framework from social network perspective**. Dans *Proc. of the 1st ACM international workshop on Complex networks meet information &#38 ; knowledge management*, CNIKM '09, pages 3–10, New York, NY, USA. ACM.

[Yang et al., 2012] Yang, J., et Leskovec, J. (2012). **Defining and evaluating network communities based on ground-truth**. Dans *2012 IEEE 12th International Conference on Data Mining (ICDM)*, pages 745–754. IEEE.

[Zhang et al., 2010] Zhang, C., Ouyang, D., et Ning, J. (2010). **An artificial bee colony approach for clustering**. *Expert Systems with Applications*, 37(7) :4761–4767.

[Zhang, 2013] Zhang, P. (2013). **Handbook of graph theory**. Chapman and Hall/CRC.

[Zhao et al., 2011] Zhao, P., Li, X., Xin, D., et Han, J. (2011). **Graph cube : on warehousing and olap multidimensional networks**. Dans *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pages 853–864. ACM.

7

ANNEXES

Tags associés à chaque ressource

R1 Repas, Viande, Graisse, Obésité
R2 Nutriments
R3 Contamination des aliments, Viande, Fruit
R4 Conservation des aliments, Viande, Fruit
R5 Sucre, Fruit, Soda, Jus de fruits
R6 Carence alimentaire
R7 Repas, Boissons alcoolisées, Facteur allergène
R8 Légumes, Maladie de la nutrition
R9 Hémopathie
R10 Facteur allergène
R11 Sucre, Soda, Obésité
R12 Grippe
R13 Facteur allergène, Rubéole
R14 Conservation des aliments, Fruit, Bactérie
R15 Contamination des aliments, Viande, Parasite
R16 Nutriments, Légumes
R17 Graisse, Facteur Allergène
R18 Légumes, Grippe
R19 Repas, Parasite
R20 Nutriments, Hémopathie
R21 Contamination des aliments, Maladie
R22 Alimentation, Grippe
R23 Aliments
R24 Boissons, Immunopathologie
R25 Fruit, Légumes, Carence alimentaire
R26 Aliments, Infection
R27 Virus
R28 Allergie
R29 Repas, Maladie
R30 Sucre, Boissons non alcoolisées
R31 Graisse, Obésité, Hémopathie
R32 Aliments, Infection
R33 Conservation des aliments, Bactérie
R34 Conservation des aliments, Parasite
R35 Carence alimentaire, Rubéole
R36 Contamination des aliments, Conservation des aliments, Boissons alcoolisées
R37 Contamination des aliments, Allergie
R38 Sucre, Jus de fruits, Immunopathologie
R39 Viande, Virus
R40 Boissons, Obésité
R41 Maladie de la nutrition
R42 Alimentation, Facteur allergène
R43 Rubéole
R44 Maladie de la nutrition, Infection
R45 Bactérie, Virus
R46 Contamination des aliments, Soda
R47 Contamination des aliments, Boissons alcoolisées
R48 Sucre, Viande, Graisse
R49 Contamination des aliments, Maladie de la nutrition, Grippe
R50 Aliments, Allergie

Liste des contacts entre utilisateurs

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15	U16	U17	U18	U19	U20
U1			1							1										
U2								1			1								1	
U3	1																			
U4						1														
U5															1					
U6				1																
U7														1		1				
U8		1																		
U9																				1
U10	1																			
U11		1																		
U12													1							
U13												1								
U14							1										1			
U15					1													1		
U16							1													
U17														1						
U18															1					
U19		1																		
U20									1											

Profils des utilisateurs

U1 : Viande, Graisse, Obésité, Fruit, Sucre, Contamination, Conservation / semi-spécialisé
U2 : Infection, Bactérie, Parasite, Virus, Grippe, Rubéole / spécialisé
U3 : Repas, Nutriments, Fruit, Jus de fruits, Carence alimentaire / semi-spécialisé
U4 : Boissons, Boissons non alcoolisées, Soda, Jus de fruits,
Boissons alcoolisées / spécialisé
U5 : Alimentation, Repas, Nutriments, Maladie de la nutrition / niveau haut du thésaurus
U6 : touche à tout, plus Contamination, Conservation, Infection
U7 : Hémopathie, Immunopathologie, Allergie, Facteur allergène / spécialisé
U8 : Maladie de la nutrition, Carence alimentaire, Obésité, Nutriments, Conservation,
Contamination /semi-spécialisé
U9 : Infection, Bactérie, Parasite, Virus, Grippe, Rubéole, Hémopathie,
Contamination / spécialisé
U10 : Maladie / touche à tout
U11 : Alimentation / touche à tout,
U12 : Aliments, Sucre, Viande, Graisse, Obésité / spécialisé
U13 : Alimentation, Repas, Hémopathie, Immonupathologie / niveau haut du thésaurus
U14 : Fruits, Légumes, Jus de fruits, Carence alimentaire / semi-spécialisé
U15 : touche à tout, plus Immunopathologie, Infection
U16 : touche à tout, plus Maladie de la nutrition, Nutriments
U17 : Maladie, Maladie de la nutrition, Hémopathie, Infection,
Immonupathologie / niveau haut du thésaurus
U18 : Alimentation, Aliments, Nutriments, Immunopathologie,
Conservation / niveau haut du thésaurus
U19 : Soda, Jus de fruits, Fruit, Carence alimentaire, Obésité / semi-spécialisé
U20 : Boissons, Boissons non alcoolisées, Soda, Jus de fruits, Conservation / spécialisé

Thésaurus et ressources associées

```
Alimentation R22 R42
----Repas R1 R7 R19 R29
----Nutriments R2 R16 R20
----Contamination des aliments R3 R15 R21 R36 R37 R46 R47 R49
----Conservation des aliments R4 R14 R33 R34 R36
----Aliments R23 R26 R32 R50
---------------Sucre R5 R11 R30 R38 R48
---------------Viande R1 R3 R4 R15 R39 R48
---------------Graisse R1 R17 R31 R48
---------------Fruit R3 R4 R5 R14 R25
---------------Légumes R8 R16 R18 R25
---------------Boissons R24 R40
---------------------Boissons non alcolisées R30
----------------------------------Soda R5 R11 R46
----------------------------------Jus de fruits R5 R38
---------------------Boissons alcolisées R7 R36 R47
Maladie R21 R29
----Maladie de la nutrition R8 R41 R44 R49
-----------Carence alimentaire R6 R25 R35
-----------Obésite R1 R11 R31 R40
----Hémopathie R9 R20 R31
----Immunopathologie R24 R38
-----------Allergie R28 R37 R50
-------------------Facteur allergène R7 R10 R13 R17 R42
----Infection R26 R32 R44
---------Bactérie  R14 R33 R45
---------Parasite R15 R19 R34
---------Virus R27 R39 R45
---------------Grippe R12 R18 R22 R49
---------------Rubéole R13 R35 R43
```

Notes attribuées par les utilisateurs aux ressources

	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23	R24	R25	R26	R27	R28	R29	R30	R31	R32	R33	R34	R35	R36	R37	R38	R39	R40	R41	R42	R43	R44	R45	R46	R47	R48	R49	R50
U1	4	0	5	4	5	0	0	2	0	0	3	1	0	0	2	0	4	0	0	1	0	0	0	0	3	0	0	0	0	3	5	0	0	0	3	0	0	3	3	4	0	0	1	2	0	3	0	4	0	0
U2	0	0	3	0	0	0	0	0	0	0	0	4	4	4	3	0	0	4	5	0	2	3	0	0	0	3	5	0	0	1	0	4	3	3	4	0	0	0	4	0	0	0	5	3	5	0	0	0	4	0
U3	5	4	4	3	4	5	5	0	0	0	0	0	0	3	0	4	0	0	4	1	0	0	0	0	5	0	0	1	3	0	0	0	0	0	4	0	0	5	0	0	0	1	0	0	0	0	1	0	0	1
U4	0	0	0	0	5	0	5	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	4	0	0	0	0	0	5	0	5	0	4	0	0	0	0	0	5	4	0	0	0
U5	5	5	0	0	0	0	0	4	0	0	0	0	0	0	0	4	0	0	5	4	2	0	3	0	0	2	0	0	2	0	0	2	0	0	0	0	0	0	0	4	3	0	5	0	0	0	0	4	0
U6	3	0	5	5	1	1	2	3	1	2	1	0	2	5	4	2	1	3	2	1	4	1	0	0	0	4	0	0	1	3	0	5	4	4	0	5	4	0	0	1	1	2	0	1	2	5	4	2	5	1
U7	0	0	0	0	0	0	4	0	5	5	0	0	4	0	0	0	5	0	0	4	0	0	0	5	0	0	0	5	0	0	3	0	0	0	0	0	4	3	0	0	0	4	0	0	0	0	0	0	0	5
U8	3	5	4	3	0	5	0	4	0	0	3	0	0	4	4	4	0	0	0	3	5	0	2	0	5	0	1	1	0	0	4	0	4	5	3	5	3	0	0	4	3	2	0	4	0	4	3	0	5	0
U9	0	0	5	0	0	0	0	0	0	0	0	5	4	5	5	0	0	3	3	0	5	3	0	0	0	3	2	0	0	1	0	4	3	3	4	4	5	0	2	0	0	0	3	4	3	4	4	0	4	0
U10	1	0	0	0	0	3	3	4	3	3	2	2	3	5	5	0	3	3	4	2	4	3	0	3	4	2	5	5	3	0	4	2	5	4	5	0	4	4	5	3	3	3	3	5	4	0	0	0	5	5
U11	5	3	4	5	3	0	5	2	0	0	4	0	0	4	3	3	0	3	3	4	2	3	1	5	3	2	0	0	4	3	3	3	4	3	0	4	5	4	2	5	0	4	0	0	0	5	4	5	1	5
U12	3	0	5	4	5	0	0	0	0	0	4	0	0	0	5	0	4	0	0	0	0	0	4	0	0	5	0	0	0	5	4	3	0	0	0	0	0	3	4	2	0	0	0	0	0	0	0	5	0	5
U13	5	0	0	0	0	0	5	0	4	0	0	0	1	1	0	0	2	0	4	5	0	4	0	5	0	1	0	0	3	0	5	0	0	1	0	1	0	3	0	0	0	5	0	1	0	0	0	0	0	1
U14	0	0	4	5	5	4	0	4	0	0	3	0	0	5	0	4	0	3	0	2	0	0	0	0	5	0	0	0	0	3	2	0	0	3	4	3	0	5	0	1	1	0	0	2	0	0	0	0	3	0
U15	1	4	3	5	2	1	3	0	4	4	0	1	5	4	3	0	4	2	3	1	0	4	2	4	0	3	2	5	0	1	1	3	4	4	3	0	4	5	2	0	1	4	4	3	5	0	1	2	3	4
U16	3	4	0	1	2	4	0	4	1	3	4	0	3	1	2	5	0	4	2	4	0	2	1	5	3	0	3	1	4	0	5	0	3	0	5	0	2	0	0	3	4	0	0	5	0	3	2	0	5	0
U17	0	0	0	0	0	0	0	3	1	0	0	0	0	0	0	0	0	0	0	5	4	0	0	4	0	1	0	0	5	0	4	3	0	0	0	0	0	5	0	0	3	0	0	3	0	0	0	0	4	0
U18	0	3	0	4	0	0	0	0	0	0	0	0	0	5	0	5	0	0	0	4	0	4	4	4	0	2	0	0	0	0	1	4	2	0	5	0	4	0	0	0	5	0	0	0	0	0	0	0	0	3
U19	5	0	3	2	5	4	0	0	0	0	4	0	0	5	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	3	0	0	0	1	0	0	3	0	4	0	0	0	0	0	4	0	0	0	0
U20	0	0	2	4	3	0	0	0	0	0	3	0	0	5	2	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	4	0	0	5	4	0	3	0	5	0	5	0	0	0	0	0	3	2	0	0	0

Historique de consultation

	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23	R24	R25	R26	R27	R28	R29	R30	R31	R32	R33	R34	R35	R36	R37	R38	R39	R40	R41	R42	R43	R44	R45	R46	R47	R48	R49	R50
U1	1	0	2	1	0	0	0	1	1	0	1	3	0	0	1	0	2	0	0	0	0	0	0	0	1	0	0	0	0	2	2	0	0	1	1	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0
U2	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	2	0	1	1	1	0	0	2	1	0	0	0	0	0	0	1	1	2	0	0	0	2	0	0	0	2	2	3	0	0	0	2	0
U3	2	1	1	0	0	0	3	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	2	0	0	0	0	1	1	0	0	0	1	0	0	1	0	0	0	1	1	0	0	0	1	0	0	1
U4	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
U5	2	3	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	3	0	0	0	0	1	0
U6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

n_{ij}

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15	U16	U17	U18	U19	U20
T1	0	0	5	0	0	4	10	5	0	9	0	0	4	4	9	8	1	0	4	0
T2	11	0	18	5	14	11	10	17	0	14	13	8	9	13	16	21	4	3	14	3
T3	2	0	10	5	4	8	9	9	0	13	7	0	5	8	8	11	3	0	4	0
T4	8	0	12	0	10	8	0	11	0	1	13	7	5	5	10	8	0	7	7	4
T5	7	0	18	10	9	9	9	14	0	13	13	5	5	13	14	17	3	3	9	3
T6	5	3	18	5	5	10	9	14	5	9	12	5	5	8	15	11	0	3	7	2
T7	16	3	20	5	9	18	10	21	5	13	17	14	4	22	23	19	4	7	14	9
T8	9	3	14	0	5	9	0	12	5	4	9	8	5	8	5	7	0	0	12	2
T9	7	0	4	5	4	5	5	4	0	7	5	5	4	9	6	7	4	0	5	3
T10	11	0	12	10	4	14	14	7	0	13	15	9	9	14	18	11	4	4	7	7
T11	9	0	16	10	5	8	4	8	0	3	16	9	5	10	14	7	0	7	7	7
T12	7	3	9	0	4	12	10	13	5	13	6	5	4	12	12	12	4	0	7	2
T13	16	3	22	10	14	15	9	16	5	11	22	13	14	13	17	14	4	3	13	5
T14	5	3	13	5	5	10	14	9	5	9	12	5	9	4	18	8	1	3	3	2
T15	10	0	8	0	9	12	5	10	0	8	12	7	9	9	10	9	4	4	7	4
T16	7	0	13	5	9	5	0	14	0	7	8	5	0	13	7	14	3	3	9	3
T17	4	0	17	5	5	11	14	13	0	12	13	4	9	9	21	13	1	7	6	4
T18	13	3	17	0	5	15	5	15	5	7	14	12	9	13	14	9	1	4	14	6
T19	7	3	9	0	4	9	0	13	5	7	6	5	0	12	4	8	3	0	7	2
T20	10	3	13	10	0	8	4	4	5	3	12	10	5	9	8	2	0	0	8	5
T21	15	3	17	0	9	20	10	19	5	14	16	12	9	17	18	16	4	4	14	6
T22	14	3	25	10	5	16	9	17	5	9	20	14	5	18	22	14	0	7	14	9
T23	11	3	13	0	14	14	10	16	5	11	14	8	9	8	16	15	4	3	8	2
T24	4	0	9	0	10	3	0	8	0	1	8	3	5	0	5	7	0	3	5	0
T25	13	3	26	5	10	19	14	20	5	13	22	12	14	13	25	16	1	7	14	6

m_{ij}

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15	U16	U17	U18	U19	U20
T1	0	0	0,125	0	0	0,1	0,2	0,111111111	0	0,2	0	0	0,1	0,1	0,257142857	0,16	0,022222222	0	0,088888889	0
T2	0,275	0	0,45	0,1	0,4	0,275	0,2	0,377777778	0	0,311111111	0,325	0,177777778	0,225	0,325	0,457142857	0,42	0,088888889	0,066666667	0,311111111	0,085714286
T3	0,05	0	0,25	0,1	0,114285714	0,2	0,18	0,2	0	0,288888889	0,175	0	0,125	0,2	0,228571429	0,22	0,066666667	0	0,088888889	0
T4	0,2	0	0,3	0	0,285714286	0,2	0	0,244444444	0	0,022222222	0,325	0,155555556	0,125	0,125	0,285714286	0,16	0	0,155555556	0,155555556	0,114285714
T5	0,175	0	0,45	0,2	0,257142857	0,225	0,18	0,311111111	0	0,288888889	0,325	0,111111111	0,125	0,325	0,4	0,34	0,066666667	0,066666667	0,2	0,085714286
T6	0,125	0,075	0,45	0,1	0,142857143	0,25	0,18	0,311111111	0,111111111	0,2	0,3	0,111111111	0,125	0,2	0,428571429	0,22	0	0,066666667	0,155555556	0,057142857
T7	0,4	0,075	0,5	0,1	0,257142857	0,45	0,2	0,466666667	0,111111111	0,288888889	0,425	0,311111111	0,1	0,55	0,657142857	0,38	0,088888889	0,155555556	0,311111111	0,257142857
T8	0,225	0,075	0,35	0	0,142857143	0,225	0	0,266666667	0,111111111	0,088888889	0,225	0,177777778	0,125	0,2	0,142857143	0,14	0	0	0,266666667	0,057142857
T9	0,175	0	0,1	0,1	0,114285714	0,125	0,1	0,088888889	0	0,155555556	0,125	0,111111111	0,1	0,225	0,17142857	0,14	0,088888889	0	0,111111111	0,085714286
T10	0,275	0	0,3	0,2	0,114285714	0,35	0,28	0,155555556	0	0,288888889	0,375	0,2	0,225	0,35	0,514285714	0,22	0,088888889	0,088888889	0,155555556	0,2
T11	0,225	0	0,4	0,2	0,142857143	0,2	0,08	0,177777778	0	0,066666667	0,4	0,2	0,125	0,25	0,4	0,14	0	0,155555556	0,155555556	0,2
T12	0,175	0,075	0,225	0	0,114285714	0,3	0,2	0,288888889	0,111111111	0,288888889	0,15	0,111111111	0,1	0,3	0,342857143	0,24	0,088888889	0	0,155555556	0,057142857
T13	0,4	0,075	0,55	0,2	0,4	0,375	0,18	0,355555556	0,111111111	0,244444444	0,55	0,288888889	0,35	0,325	0,485714286	0,28	0,088888889	0,066666667	0,288888889	0,142857143
T14	0,125	0,075	0,325	0,1	0,142857143	0,25	0,28	0,2	0,111111111	0,2	0,3	0,111111111	0,225	0,1	0,514285714	0,16	0,022222222	0,066666667	0,066666667	0,057142857
T15	0,25	0	0,2	0	0,257142857	0,3	0,1	0,222222222	0	0,177777778	0,3	0,155555556	0,225	0,225	0,285714286	0,18	0,088888889	0,088888889	0,155555556	0,114285714
T16	0,175	0	0,325	0,1	0,257142857	0,125	0	0,311111111	0	0,155555556	0,2	0,111111111	0	0,325	0,2	0,28	0,066666667	0,066666667	0,2	0,085714286
T17	0,1	0	0,425	0,1	0,142857143	0,275	0,28	0,288888889	0	0,266666667	0,325	0,088888889	0,225	0,225	0,6	0,26	0,022222222	0,155555556	0,133333333	0,114285714
T18	0,325	0,075	0,425	0	0,142857143	0,375	0,1	0,333333333	0,111111111	0,155555556	0,35	0,266666667	0,225	0,325	0,4	0,18	0,022222222	0,088888889	0,311111111	0,171428571
T19	0,175	0,075	0,225	0	0,114285714	0,225	0	0,288888889	0,111111111	0,155555556	0,15	0,111111111	0	0,3	0,114285714	0,16	0,066666667	0	0,155555556	0,057142857
T20	0,25	0,075	0,325	0,2	0	0,2	0,08	0,088888889	0,111111111	0,066666667	0,3	0,222222222	0,125	0,225	0,228571429	0,04	0	0	0,177777778	0,142857143
T21	0,375	0,075	0,425	0	0,257142857	0,5	0,2	0,422222222	0,111111111	0,311111111	0,4	0,266666667	0,225	0,425	0,514285714	0,32	0,088888889	0,088888889	0,311111111	0,171428571
T22	0,35	0,075	0,625	0,2	0,142857143	0,4	0,18	0,377777778	0,111111111	0,2	0,5	0,311111111	0,125	0,45	0,628571429	0,28	0	0,155555556	0,311111111	0,257142857
T23	0,275	0,075	0,325	0	0,4	0,35	0,2	0,355555556	0,111111111	0,244444444	0,35	0,177777778	0,225	0,2	0,457142857	0,3	0,088888889	0,066666667	0,177777778	0,057142857
T24	0,1	0	0,225	0	0,285714286	0,075	0	0,177777778	0	0,022222222	0,2	0,066666667	0,125	0	0,142857143	0,14	0	0,066666667	0,111111111	0
T25	0,325	0,075	0,65	0,1	0,285714286	0,475	0,28	0,444444444	0,111111111	0,288888889	0,55	0,266666667	0,35	0,325	0,714285714	0,32	0,022222222	0,155555556	0,311111111	0,171428571

d_{ij}

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15	U16	U17	U18	U19	U20
T1	0	0	0,025	0	0	0,02	0,04	0,022222222	0	0,04	0	0	0,02	0,02	0,051428571	0,032	0,004444444	0	0,017777778	0
T2	0,1375	0	0,09	0,02	0,08	0,055	0,04	0,075555556	0	0,062222222	0,065	0,035555556	0,045	0,065	0,091428571	0,084	0,017777778	0,013333333	0,062222222	0,017142857
T3	0,01	0	0,05	0,02	0,022857143	0,04	0,036	0,04	0	0,057777778	0,035	0	0,025	0,04	0,045714286	0,044	0,013333333	0	0,017777778	0
T4	0,04	0	0,06	0	0,057142857	0,04	0	0,048888889	0	0,004444444	0,065	0,031111111	0,025	0,025	0,057142857	0,032	0	0,031111111	0,031111111	0,022857143
T5	0,035	0	0,09	0,04	0,051428571	0,045	0,036	0,062222222	0	0,057777778	0,065	0,022222222	0,025	0,065	0,08	0,068	0,013333333	0,013333333	0,04	0,017142857
T6	0,025	0,015	0,09	0,02	0,028571429	0,05	0,036	0,062222222	0,022222222	0,04	0,06	0,022222222	0,025	0,04	0,085714286	0,044	0	0,013333333	0,031111111	0,011428571
T7	0,08	0,015	0,1	0,02	0,051428571	0,09	0,04	0,093333333	0,022222222	0,057777778	0,085	0,062222222	0,02	0,11	0,131428571	0,076	0,017777778	0,031111111	0,062222222	0,051428571
T8	0,045	0,015	0,07	0	0,028571429	0,045	0	0,053333333	0,022222222	0,017777778	0,045	0,035555556	0,025	0,04	0,028571429	0,028	0	0	0,053333333	0,011428571
T9	0,035	0	0,02	0,02	0,022857143	0,025	0,02	0,017777778	0	0,031111111	0,025	0,022222222	0,02	0,045	0,034285714	0,028	0,017777778	0	0,022222222	0,017142857
T10	0,055	0	0,06	0,04	0,022857143	0,07	0,056	0,031111111	0	0,057777778	0,075	0,04	0,045	0,07	0,102857143	0,044	0,017777778	0,017777778	0,031111111	0,04
T11	0,045	0	0,08	0,04	0,028571429	0,04	0,016	0,035555556	0	0,013333333	0,08	0,04	0,025	0,05	0,08	0,028	0	0,031111111	0,031111111	0,04
T12	0,035	0,015	0,045	0	0,022857143	0,06	0,04	0,057777778	0,022222222	0,057777778	0,03	0,022222222	0,02	0,06	0,068571429	0,048	0,017777778	0	0,031111111	0,011428571
T13	0,08	0,015	0,11	0,04	0,08	0,075	0,036	0,071111111	0,022222222	0,048888889	0,11	0,057777778	0,07	0,065	0,097142857	0,056	0,017777778	0,013333333	0,057777778	0,028571429
T14	0,025	0,015	0,065	0,02	0,028571429	0,05	0,056	0,04	0,022222222	0,04	0,06	0,022222222	0,045	0,02	0,102857143	0,032	0,004444444	0,013333333	0,013333333	0,011428571
T15	0,05	0	0,04	0	0,051428571	0,06	0,02	0,044444444	0	0,035555556	0,06	0,031111111	0,045	0,045	0,057142857	0,036	0,017777778	0,017777778	0,031111111	0,022857143
T16	0,035	0	0,065	0,02	0,051428571	0,025	0	0,062222222	0	0,031111111	0,04	0,022222222	0	0,065	0,04	0,056	0,013333333	0,013333333	0,04	0,017142857
T17	0,02	0	0,085	0,02	0,028571429	0,055	0,056	0,057777778	0	0,053333333	0,065	0,017777778	0,045	0,045	0,12	0,052	0,004444444	0,031111111	0,026666667	0,022857143
T18	0,065	0,015	0,085	0	0,028571429	0,075	0,02	0,066666667	0,022222222	0,031111111	0,07	0,053333333	0,045	0,065	0,08	0,036	0,004444444	0,017777778	0,062222222	0,034285714
T19	0,035	0,015	0,045	0	0,022857143	0,045	0	0,057777778	0,022222222	0,031111111	0,03	0,022222222	0	0,06	0,022857143	0,032	0,013333333	0	0,031111111	0,011428571
T20	0,05	0,015	0,065	0,04	0	0,04	0,016	0,017777778	0,022222222	0,013333333	0,06	0,044444444	0,025	0,045	0,045714286	0,008	0	0	0,035555556	0,028571429
T21	0,075	0,015	0,085	0	0,051428571	0,1	0,04	0,084444444	0,022222222	0,062222222	0,08	0,053333333	0,045	0,085	0,102857143	0,064	0,017777778	0,017777778	0,062222222	0,034285714
T22	0,07	0,015	0,125	0,04	0,028571429	0,08	0,036	0,075555556	0,022222222	0,04	0,1	0,062222222	0,025	0,09	0,125714286	0,056	0	0,031111111	0,062222222	0,051428571
T23	0,055	0,015	0,065	0	0,08	0,07	0,04	0,071111111	0,022222222	0,048888889	0,07	0,035555556	0,045	0,04	0,091428571	0,06	0,017777778	0,013333333	0,035555556	0,011428571
T24	0,02	0	0,045	0	0,057142857	0,015	0	0,035555556	0	0,004444444	0,04	0,013333333	0,025	0	0,028571429	0,028	0	0,013333333	0,022222222	0
T25	0,065	0,015	0,13	0,02	0,057142857	0,095	0,056	0,088888889	0,022222222	0,057777778	0,11	0,053333333	0,07	0,065	0,142857143	0,064	0,004444444	0,031111111	0,062222222	0,034285714

Document réalisé avec LaTeX et :
le style LaTeX pour Thèse de Doctorat créé par S. Galland — http://www.multiagent.fr/ThesisStyle
la collection de paquets `tex-upmethodology` — http://www.arakhne.org/tex-upmethodology/

Résumé :

La gestion de la relation client (GRC, ou CRM en anglais), est un terme ayant émergé au milieu des années 1990, souvent utilisé pour décrire des outils informatisés offrant des prestations de services à des consommateurs avant, pendant et après un achat. Ces consommateurs ont suivit la transformation du Web qui a lieu depuis quelques années, où tout utilisateur devient fournisseur de contenu au moyen d'outils comme, notamment, les réseaux sociaux avec le partage de ressources, de contenu et les mécanismes d'annotation. Les réseaux sociaux grand public, tels que Facebook ou Twitter, sont maintenant utilisés quotidiennement par un très grand nombre d'utilisateurs. Les entreprises doivent suivre cette évolution et intégrer les réseaux sociaux comme nouveau canal de communication afin de pouvoir interagir avec leurs clients et prospects, et réciproquement. Ainsi, le Social CRM ou s-CRM, constitue l'évolution du CRM liée à l'intégration des médias sociaux dans le cadre d'outils de gestion de la relation client. Cette évolution doit permettre de prendre en compte la richesse des données présentes sur les réseaux sociaux et les valoriser. Les données produites par les les réseaux sociaux en ligne ont un potentiel de valorisation très important dans le cadre du CRM. De nombreux outils algorithmiques ont été développés pour analyser ces données. Cependant, afin de prendre en compte cette diversité, aussi bien en matière de données, d'algorithmes, que d'usages, il est nécessaire d'élaborer une plateforme générique utilisable par des spécialistes du domaine métier. Cette plateforme devra permettre de gérer la collecte, le stockage ainsi que l'analyse et la visualisation des données issues des réseaux sociaux. Afin de répondre aux multiples besoins de l'analyse des données sociales (analyse de communautés, identification d'utilisateur influents, détections d'événements, etc.), et de réduire le temps de mise en forme des données pour des analyses en temps réel, tout en accédant rapidement à de grandes quantités de données, plusieurs paradigmes de modélisation des données (en vue de leur stockage) doivent être envisagés. La sémantique, abordée dans notre cas comme la contextualisation des données ou des résultats des algorithmes par rapport à la connaissance du domaine, ainsi que la modélisation des données issues des réseaux sociaux par des graphes ou des profils utilisateurs, constituent les deux fils conducteurs de nos recherches. Nous proposons une méthode de détection et de caractérisation d'événements ne nécessitant pas de paramètre spécifique, hormis une fenêtre temporelle pour détecter des événements courts ou longs. La caractérisation des événements détectés est réalisée par une contextualisation au moyen des hashtags les plus représentatifs sur la période détectée. Nous proposons également une méthode détection de l'influence s'appuyant sur un graphe orienté et pondéré, exploitant l'algorithme HITS. Les différents algorithmes sont validés avec des données réelles. Un proof of concept de plateforme, appelé SNFreezer, a été développé et a permis de valider plusieurs des fonctionnalités au travers de différents projets. Nous avons validé la scalabilité de la plateforme au niveau de la collecte et du stockage des données. Cette plateforme a servi de point de départ pour une implémentation plus industrielle. Cette dernière est liée au développement du projet DisCoCRM, qui s'inscrit dans la démarche des sociétés eb-Lab et Teletech International d'intégrer les réseaux sociaux dans leur offre d'outils CRM, initiée avec le début de ma thèse.

Mots-clés : Médias sociaux, Relation client, Communautés

Abstract:

Customer relationship management (CRM) is a term that emerged in the middle of the 1990's, and that is often used to describe computerized tools that provide services to consumers before, during and after a sale. These consumers have followed the transformation of the Web that has happened in the last few years, where each user becomes a supplier of content using tools like, amongst others, social networks by sharing resources, content, and annotating. General use social networks, such as Facebook or Twitter, are now used daily by a very large number of users. Companies have to follow this evolution and include social networks as a new communication channel in order to interact with their clients and prospects, and vice versa. Social CRM, or s-CRM, is the CRM evolution linked to the integration of social networks in the tools used to customer relationship management. This evolution has to take into account the richness of the social networks data and take advantage of them. Social network data have a great potential of adding value to CRM. Several tools have been developed to analyse them. However, in order to take into account the diversity, in terms of data, algorithms and use, a generic platform that can be used by experts has to be developed. This platform will have to manage the collection, storing and analysis and visualisation of social network data. In order to answer the multiple needs related to the social network data analysis (community analysis, influential users identification, event detection, etc.), and to reduce the data formatting time for real time analysis, while quickly accessing large amount of data, several data modelling paradigms (so as to store them) have to be considered. Semantics, which we see as the contextualization of data or the results of the algorithms in relation to the domain knowledge, and also social network data modelling with graphs or user profiles, are the two central themes of our research. We submit a community detection and event characterization that doesn't use any specific parameter, except for a time window used to detect long or short events. The events are characterized by contextualizing with the most representative hashtags on the detected period. We also submit an influence detection method using weighted and directed graph and the HITS algorithm. The various algorithms are validated with real world data. A proof of concept of platform, called SNFreezer, has been developed and used to validate several features through various projects. We validated the data collection and storing scalability of the platform. This platform was the starting of a more industrial implementation. This was linked to the DisCoCRM project that is part of the will, initiated with the beginning of my thesis, of eb-Lab and Teletech International to integrate social networks into their CRM tools.

Keywords: Social networks, Customer relationship management, Communities

SPIM

■ École doctorale SPIM - Université de Bourgogne/UFR ST BP 47870 F - 21078 Dijon cedex

■ tél. +33 (0)3 80 39 59 10 ■ ed-spim@univ-fcomte.fr ■ www.ed-spim.univ-fcomte.fr

UNIVERSITÉ DE BOURGOGNE